DB2 10 for z/OS

Installation and Migration Guide

GC19-2974-08

DB2 10 for z/OS

Installation and Migration Guide

GC19-2974-08

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” at the end of this information.

Ninth edition (August 2012)

This edition applies to DB2 10 for z/OS (product number 5605-DB2), DB2 10 for z/OS Value Unit Edition (product number 5697-P31), and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Specific changes are indicated by a vertical bar to the left of a change. A vertical bar to the left of a figure caption indicates that the figure has changed. Editorial changes that have no technical significance are not noted.

© Copyright IBM Corporation 1982, 2012.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this information . . . xvii
Who should read this information . . . xvii
Migration information . . . xvii
DB2 Utilities Suite . . . xviii
Terminology and citations . . . xviii
Accessibility features for DB2 10 for z/OS . . . xviii
How to send your comments . . . xix
How to read syntax diagrams . . . xix

Chapter 1. Introduction to installation and migration . . . 1
Installation and migration tools . . . 2
Introduction to installation. . . . 3
Summary of installation steps . . . 3
Introduction to migration from DB2 Version 8 . . . 4
Migration modes for DB2 8 to DB2 10 . . . 5
Migration to conversion mode from Version 8: Summary . . . 7
Falling back to Version 8: Summary. . . . 9
Remigration from Version 8: Summary . . . 10
Migration to enabling-new-function mode from Version 8: Summary . . . 10
Migration from enabling-new-function mode from Version 8 to new-function mode: Summary . . . 11
Reversion to conversion mode* from Version 8: Summary . . . 12
Reversion to enabling-new-function mode* from Version 8: Summary . . . 12
Migration from conversion mode* from Version 8 to enabling-new-function mode from Version 8: Summary . . . 12
Migration from conversion mode* from Version 8 to new-function mode: Summary . . . 13
Migration from enabling-new-function mode* from Version 8 to new-function mode: Summary . . . 13
Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8 . . . 14
Checklist for migration to DB2 Version 10 conversion mode from Version 8 . . . 15
Introduction to migration from DB2 Version 9.1 . . . 16
Migration modes for DB2 9 to DB2 10 . . . 16
Migration to conversion mode from Version 9.1: Summary . . . 18
Falling back to Version 9.1: Summary. . . . 20
Remigration from Version 9.1: Summary. . . . 21
Migration to enabling-new-function mode from Version 9.1: Summary. . . . 21
Migration from enabling-new-function mode from Version 9.1 to new-function mode: Summary . . . 22
Reversion to conversion mode* from Version 9.1: Summary . . . 22
Reversion to enabling-new-function mode* from Version 9.1: Summary . . . 23
Migration from conversion mode* from Version 9.1 to enabling-new-function mode from Version 9.1: Summary 23
Migration from conversion mode* from Version 9.1 to new-function mode: Summary . . . 24
Migration from enabling-new-function mode* from Version 9.1 to new-function mode: Summary. . . . 24
Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1 . . . 24
Checklist for migration to DB2 Version 10 conversion mode from Version 9.1 . . . 25

Chapter 2. Preparing your system to install or migrate DB2 . . . 27
Hardware and software requirements. . . . 27
Required maintenance . . . 27
Planning storage for DB2 . . . 28
DB2 subsystem storage requirements . . . 29
Virtual storage requirements for address spaces . . . 46
Virtual storage requirements for storage pools and working storage . . . 51
Calculating real storage requirements. . . . 62
Planning for performance of DB2 . . . 64
Converting EXPLAIN tables (before migration) . . . 64
Changes that might affect your migration from Version 8 . . . 65
Compatibility mode is now called conversion mode (from Version 8 or Version 9.1) . . . 65
Availability of new functions in conversion mode from Version 8 and new-function mode . . . 65

© Copyright IBM Corp. 1982, 2012

Application and SQL release incompatibilities for migration from Version 8 . . . 66
Utility release incompatibilities for migration from Version 8 . . . 92
Command release incompatibilities for migration from Version 8 . . . 94
Security release incompatibilities for migration from Version 8 . . . 95
Storage release incompatibilities for migration from Version 8 . . . 96
Other release incompatibilities for migration from Version 8 . . . 98
Functions that are deprecated for systems migrating from Version 8 . . . 105
Functions that are no longer supported for systems migrating from Version 8 . . . 107
Changes that might affect your migration from Version 9.1 . . . 114
Availability of new functions in conversion mode from Version 9.1 and new-function mode . . . 114
Application and SQL release incompatibilities for migration from Version 9.1 . . . 114
Utility release incompatibilities for migration from Version 9.1 . . . 130
Command release incompatibilities for migration from Version 9.1 . . . 132
Security release incompatibilities for migration from Version 9.1 . . . 132
Other release incompatibilities for migration from Version 9.1 . . . 133
Functions that are deprecated for systems migrating from Version 9.1 . . . 139
Functions that are no longer supported for systems migrating from Version 9.1 . . . 141
Preparing for DB2 data sharing . . . 146
DB2 data sharing in a Parallel Sysplex . . . 147
Data sharing naming conventions . . . 153
Planning for availability in a data sharing environment . . . 166
Storage estimates for data sharing environments . . . 179
Before you enable DB2 data sharing . . . 194
Application design planning . . . 197
Release coexistence considerations . . . 202

Chapter 3. Loading the DB2 libraries . . . 207
Editing the SMP/E jobs . . . 207
Creating JOB statements . . . 207
Choosing link list options . . . 208
Accessing the correct DB2 program library . . . 210
Performance implications of installation libraries . . . 211
Naming conventions for DB2 library names . . . 212
Specifying SMP/E data set options . . . 213
Specifying a new SMP/E prefix for IRLM . . . 214
DB2 utilities packaging . . . 214
SMP/E jobs for DB2 utility products . . . 215
Operation of DB2 utilities in a mixed-release data sharing environment . . . 215
What you produce . . . 216

Chapter 4. Tailoring DB2 jobs to your environment using the installation CLIST . . . 219
Running the installation CLIST . . . 219
Making the DB2 ISPF libraries available to TSO . . . 219
Updating the CLIST defaults input member: job DSNTIJXZ . . . 220
Multiple procedure libraries . . . 223
System affinity for installation jobs . . . 224
Invoking the CLIST . . . 224
General instructions for using the installation CLIST . . . 225
Directory of panel field names. . . . 229
Directory of subsystem parameters and application default values. . . . 242
Subsystem parameters that are not on installation panels . . . 253
Subsystem parameter module and application defaults module values . . . 265
Main panel: DSNTIPA1 . . . 265
INSTALL TYPE field . . . 267
DATA SHARING field (DSHARE subsystem parameter) . . . 268
FROM RELEASE field . . . 269
DATA SET(MEMBER) NAME field . . . 269
LIBRARY NAME PREFIX field . . . 270
LIBRARY NAME SUFFIX field . . . 271
DATA SET NAME PREFIX field . . . 271

DATA SET NAME SUFFIX field . . . 271
INPUT MEMBER NAME field. . . . 272
OUTPUT MEMBER NAME field . . . 274
OTC license usage panel: DSNTIPO1 . . . 274
Notice and acceptance of OTC license panel: DSNTIPO2 . . . 275
LICENSE TERMS ACCEPTED field (OTC_LICENSE subsystem parameter). . . . 275
DB2 catalog and directory panel: DSNTIPA2 . . . 275
CATALOG ALIAS field (CATALOG subsystem parameter) . . . 276
DEFINE CATALOG field . . . 279
DIRECTORY AND CATALOG DATA field (CATDDACL, CATDMGCL, and CATDSTCL subsystem parameters) . . . 279
DIRECTORY AND CATALOG INDEXES field (CATXDACL, CATXMGCL, and CATXSTCL subsystem parameters) . . . 280
COMPRESS SPT01 field (COMPRESS_SPT01 subsystem parameter) . . . 280
SPT01 INLINE LENGTH field (SPT01_INLINE_LENGTH subsystem parameter) . . . 281
Data parameters panel: DSNTIPA3 . . . 281
PERMANENT UNIT NAME field . . . 282
TEMPORARY UNIT NAME field. . . . 283
CLIST ALLOCATION field . . . 283
NON-VSAM DATA field . . . 283
VSAM CATALOG, DEFAULT AND WORK FILE DATABASE field . . . 284
LOG COPY 1, BSDS 2 field . . . 285
LOG COPY 2, BSDS 1 field . . . 285
Define group or member panel: DSNTIPK. . . . 286
GROUP NAME field (GRPNAME subsystem parameter) . . . 287
MEMBER NAME field (MEMBNAME subsystem parameter) . . . 287
WORK FILE DB field. . . . 288
GROUP ATTACH field (SSID DECP value) . . . 288
SUBGRP ATTACH field . . . 288
COORDINATOR field (COORDNTR subsystem parameter) . . . 289
ASSISTANT field (ASSIST subsystem parameter) . . . 289
RANDOM ATTACH field (RANDOMATT subsystem parameter) . . . 289
DEL CF STRUCTS field (DEL_CFSTRUCTS_ON_RESTART subsystem parameter) . . . 290
System resource data set names panel: DSNTIPH . . . 290
COPY 1 NAME field . . . 291
COPY 2 NAME field . . . 292
NUMBER OF COPIES field (TWOACTV subsystem parameter) . . . 292
COPY 1 PREFIX field. . . . 292
COPY 2 PREFIX field. . . . 293
NUMBER OF COPIES field (TWOARCH subsystem parameter) . . . 293
COPY 1 PREFIX field (ARCPFX1 subsystem parameter) . . . 293
COPY 2 PREFIX field (ARCPFX2 subsystem parameter) . . . 294
TIMESTAMP ARCHIVES field (TSTAMP subsystem parameter) . . . 294
Data set names panel 1: DSNTIPT . . . 295
TEMP CLIST LIBRARY field . . . 296
SAMPLE LIBRARY field. . . . 296
CLIST LIBRARY field. . . . 297
APPLICATION DBRM field . . . 297
APPLICATION LOAD field . . . 297
DECLARATION LIBRARY field . . . 298
LINK LIST LIBRARY field . . . 298
LOAD LIBRARY field . . . 298
MACRO LIBRARY field . . . 298
LOAD DISTRIBUTION field . . . 299
EXIT LIBRARY field . . . 299
DBRM LIBRARY field . . . 299
IRLM LOAD LIBRARY field . . . 299
IVP DATA LIBRARY field . . . 300
INCLUDE LIBRARY field . . . 300
Data set names panel 2: DSNTIPU . . . 300
IBM LE RUNTIME LIBRARY field . . . 304

IBM LE LINK EDIT LIB field . . . 304
IBM LE PRELINK MSG LIB field . . . 305
HIGH LEVEL ASSEMBLER LIB field . . . 305
C/CPP COMPILER MODULE field . . . 305
C/CPP COMPILER LIBRARY field . . . 306
C/CPP HEADER LIBRARY field . . . 306
C/370 COMPILER MESSAGES field . . . 307
CPP CLASS LIB HEADERS field . . . 307
CPP AUTO CALL LIBRARY field . . . 307
CPP CLASS LIBRARY field . . . 308
CPP PROCEDURE LIBRARY field . . . 308
COBOL COMPILER LIBRARY field . . . 308
FORTRAN COMPILER LIBRARY field . . . 308
FORTRAN LINK EDIT LIB field . . . 309
PL/I COMPILER LIBRARY field . . . 309
Data set names panel 3: DSNTIPW . . . 309
SYSTEM MACLIB field . . . 310
SYSTEM PROCEDURES field . . . 310
SORT LIBRARY field . . . 310
IMS RESLIB field . . . 311
ISPF ISPLINK MODULE field . . . 312
GDDM MACLIB field . . . 312
GDDM LOAD MODULES field . . . 312
CICS LOAD LIBRARY field . . . 312
CICS MACRO LIBRARY field . . . 313
CICS COBOL LIBRARY field . . . 313
CICS PL/I LIBRARY field . . . 313
CICS EXCI LIBRARY field . . . 313
Sizes panel 1: DSNTIPD . . . 313
DATABASES field for panel DSNTIPD . . . 314
TABLES field . . . 315
COLUMNS field . . . 315
VIEWS field . . . 315
TABLE SPACES field . . . 315
PLANS field . . . 315
PLAN STATEMENTS field . . . 316
PACKAGES field . . . 316
PACKAGE STATEMENTS field . . . 316
PACKAGE LISTS field . . . 316
EXECUTED STMTS field . . . 316
TABLES IN STMT field . . . 317
LOB INLINE LENGTH field (LOB_INLINE_LENGTH subsystem parameter) . . . 317
USER LOB VALUE STG field (LOBVALA subsystem parameter) . . . 317
SYSTEM LOB VAL STG field (LOBVALS subsystem parameter) . . . 318
USER XML VALUE STG field (XMLVALA subsystem parameter) . . . 318
SYSTEM XML VALUE STG field (XMLVALS subsystem parameter) . . . 318
Sizes panel 2: DSNTIP7 . . . 318
DEFINE DATA SETS field (IMPDSDEF subsystem parameter) . . . 319
USE DATA COMPRESSION field (IMPTSCMP subsystem parameter) . . . 320
TABLE SPACE ALLOCATION field (TSQTY subsystem parameter) . . . 320
INDEX SPACE ALLOCATION field (IXQTY subsystem parameter) . . . 320
VARY DS CONTROL INTERVAL field (DSVCI subsystem parameter) . . . 321
OPTIMIZE EXTENT SIZING field (MGEXTSZ subsystem parameter) . . . 321
REORDERED ROW FORMAT field (RRF subsystem parameter) . . . 322
DEFAULT PARTITION SEGSIZE field (DPSEGSZ subsystem parameter) . . . 322
Work file database panel: DSNTIP9 . . . 323
4K SORT PRIMARY SPACE field . . . 325
4K SORT WORK TS'S field . . . 326
4K SORT WORK SEG SIZE field . . . 326
32K SORT PRIMARY SPACE field . . . 326
32K SORT WORK TS'S field . . . 327

32K SORT WORK SEG SIZE field . . . 327
4K TEMP PRIMARY SPACE field. . . . 328
4K TEMP WORK TS'S field . . . 329
4K TEMP WORK SEG SIZE field . . . 329
32K TEMP PRIMARY SPACE field . . . 329
32K TEMP WORK TS'S field . . . 330
32K TEMP WORK SEG SIZE field . . . 331
MAX TEMP STG/AGENT field (MAXTEMPS subsystem parameter) . . . 331
SEPARATE WORK FILES field (WFDBSEP subsystem parameter) . . . 331
MAX TEMP RID field (MAXTEMPS_RID subsystem parameter) . . . 332
Thread management panel: DSNTIPE . . . 333
DATABASES field for panel DSNTIPE . . . 333
MAX USERS field (CTHREAD subsystem parameter) . . . 334
MAX REMOTE ACTIVE field (MAXDBAT subsystem parameter) . . . 334
MAX REMOTE CONNECTED field (CONDBAT subsystem parameter) . . . 335
MAX TSO CONNECT field (IDFORE subsystem parameter). . . . 336
MAX BATCH CONNECT field (IDBACK subsystem parameter) . . . 336
MAX KEPT DYN STMTS field (MAXKEEPD subsystem parameter) . . . 337
CONTRACT THREAD STG field (CONTSTOR subsystem parameter) . . . 337
MANAGE THREAD STORAGE field (MINSTOR subsystem parameter). . . . 338
LONG-RUNNING READER field (LRDRTHLD subsystem parameter) . . . 338
PAD INDEXES BY DEFAULT field (PADIX subsystem parameter) . . . 338
MAX OPEN FILE REFS field (MAXOFILR subsystem parameter) . . . 339
Buffer pool sizes panel 1: DSNTIP1 . . . 339
DEFAULT 4-KB BUFFER POOL FOR USER DATA field (TBSBPOOL subsystem parameter) . . . 340
DEFAULT 8-KB BUFFER POOL FOR USER DATA field (TBSBP8K subsystem parameter) . . . 340
DEFAULT 16-KB BUFFER POOL FOR USER DATA field (TBSBP16K subsystem parameter) . . . 340
DEFAULT 32-KB BUFFER POOL FOR USER DATA field (TBSBP32K subsystem parameter) . . . 341
DEFAULT BUFFER POOL FOR USER LOB DATA field (TBSBPLOB subsystem parameter) . . . 341
DEFAULT BUFFER POOL FOR USER XML DATA field (TBSBPXML subsystem parameter) . . . 341
DEFAULT BUFFER POOL FOR USER INDEXES field (IDXBPOOL subsystem parameter) . . . 342
BUFFERPOOL fields BP0 - BP29 . . . 342
Buffer pool sizes panel 2: DSNTIP2 . . . 342
BUFFERPOOL fields BP30 - BP32K9. . . . 343
Tracing parameters panel: DSNTIPN . . . 343
AUDIT TRACE field (AUDITST subsystem parameter) . . . 344
TRACE AUTO START field (TRACSTR subsystem parameter) . . . 345
TRACE SIZE field (TRACTBL subsystem parameter) . . . 345
SMF ACCOUNTING field (SMFACCT subsystem parameter) . . . 346
SMF STATISTICS field (SMFSTAT subsystem parameter) . . . 346
STATISTICS TIME field (STATIME subsystem parameter) . . . 347
STATISTICS SYNC field (SYNCVAL subsystem parameter) . . . 347
DATASET STATS TIME field (DSSTIME subsystem parameter) . . . 348
MONITOR TRACE field (MON subsystem parameter) . . . 348
MONITOR SIZE field (MONSIZE subsystem parameter) . . . 348
UNICODE IFCIDS field (UIFCIDS subsystem parameter). . . . 349
DDF/RRSAF ACCUM field (ACCUMACC subsystem parameter) . . . 349
AGGREGATION FIELDS field (ACCUMUID subsystem parameter) . . . 350
COMPRESS SMF RECS field (SMFCOMP subsystem parameter) . . . 351
Operator functions panel: DSNTIPO. . . . 351
WTO ROUTE CODES field (ROUTCDE subsystem parameter) . . . 352
RECALL DATABASE field (RECALL subsystem parameter) . . . 352
RECALL DELAY field (RECALLD subsystem parameter). . . . 352
RLF AUTO START field (RLF subsystem parameter) . . . 353
RLST NAME SUFFIX field (RLFTBL subsystem parameter) . . . 353
RLST ACCESS ERROR field (RLFERR subsystem parameter) . . . 353
AUTO BIND field (ABIND subsystem parameter) . . . 354
EXPLAIN PROCESSING field (ABEXP subsystem parameter) . . . 355
DPROP SUPPORT field (EDPROP and CHGDC subsystem parameters) . . . 356
SITE TYPE field (SITETYP subsystem parameter) . . . 357
TRACKER SITE field (TRKRSITE subsystem parameter) . . . 357

READ COPY2 ARCHIVE field (ARC2FRST subsystem parameter) . . . 357
REAL TIME STATS field (STATSINT subsystem parameter) . . . 358
Default startup modules panel: DSNTIPO3 . . . 358
PARAMETER MODULE field . . . 358
APPL DEFAULTS field . . . 359
ACCESS CONTROL field (ACCESS_CNTL_MODULE subsystem parameter) . . . 360
IDENTIFY/AUTH field (IDAUTH_MODULE subsystem parameter) . . . 360
SIGNON field (SIGNON_MODULE subsystem parameter) . . . 361
Application programming defaults panel 1: DSNTIPF . . . 362
LANGUAGE DEFAULT field (DEFLANG DECP value) . . . 364
DECIMAL POINT IS field (DECIMAL DECP value) . . . 364
STRING DELIMITER field (DELIM DECP value) . . . 365
SQL STRING DELIMITER field (SQLDELI DECP value) . . . 365
DIST SQL STR DELIMTR field (DSQLDELI DECP value) . . . 366
MIXED DATA field (MIXED DECP value) . . . 367
EBCDIC CCSID field (SCCSID, MCCSID, and GCCSID DECP values) . . . 367
ASCII CCSID field (ASCCSID, AMCCSID, AGCCSID DECP values) . . . 369
UNICODE CCSID field (USCCSID, UMCCSID, and UGCCSID DECP values) . . . 369
DEF ENCODING SCHEME field (ENSCHEME DECP value) . . . 369
APPLICATION ENCODING field (APPENSCH DECP value) . . . 370
LOCALE LC_CTYPE field (LC_TYPE DECP value) . . . 370
DECFLOAT ROUNDING MODE field (DEF_DECFLOAT_ROUND_MODE DECP value) . . . 371
Application programming defaults panel 2: DSNTIP4 . . . 372
MINIMUM DIVIDE SCALE field (DECDIV3 subsystem parameter) . . . 373
DECIMAL ARITHMETIC field (DECARTH DECP value) . . . 373
USE FOR DYNAMICRULES field (DYNRULS DECP value) . . . 374
DESCRIBE FOR STATIC field (DESCSTAT subsystem parameter) . . . 374
DATE FORMAT field (DATE DECP value) . . . 375
TIME FORMAT field (TIME DECP value) . . . 376
LOCAL DATE LENGTH field (DATELEN DECP value) . . . 377
LOCAL TIME LENGTH field (TIMELEN DECP value) . . . 377
IMPLICIT TIME ZONE field (IMPLICIT_TIMEZONE DECP value) . . . 377
STD SQL LANGUAGE field (STDSQL DECP value) . . . 378
PAD NUL-TERMINATED field (PADNTSTR DECP value) . . . 378
Performance and optimization panel: DSNTIP8 . . . 379
CURRENT DEGREE field (CDSSRDEF subsystem parameter) . . . 379
CACHE DYNAMIC SQL field (CACHEDYN subsystem parameter) . . . 380
OPTIMIZATION HINTS field (OPTHINTS subsystem parameter) . . . 380
MAX DEGREE field (PARAMDEG subsystem parameter) . . . 381
PARALLELISM EFFICIENCY field (PARA_EFF subsystem parameter) . . . 381
IMMEDIATE WRITE field (IMMEDWRI subsystem parameter) . . . 382
EVALUATE UNCOMMITTED field (EVALUNC subsystem parameter) . . . 382
SKIP UNCOMM INSERTS field (SKIPUNCI subsystem parameter) . . . 383
CURRENT REFRESH AGE field (REFSHAGE subsystem parameter) . . . 383
CURRENT MAINT TYPES field (MAINTYPE subsystem parameter) . . . 383
STAR JOIN QUERIES field (STARJOIN subsystem parameter) . . . 384
MAX DATA CACHING field (MXDTCACH subsystem parameter) . . . 384
PLAN MANAGEMENT field (PLANMGMT subsystem parameter) . . . 385
PLAN MANAGEMENT SCOPE field (PLANMGMTSCOPE subsystem parameter) . . . 385
RANDOMIZE XML DOCID field (XML_RANDOMIZE_DOCID subsystem parameter) . . . 385
IRLM panel 1: DSNTIPI . . . 386
INSTALL IRLM field . . . 386
SUBSYSTEM NAME field (IRLMSID subsystem parameter) . . . 387
RESOURCE TIMEOUT field (IRLMRWT subsystem parameter) . . . 387
AUTO START field (IRLMAUT subsystem parameter) . . . 388
PROC NAME field (IRLMPRC subsystem parameter) . . . 388
TIME TO AUTOSTART field (IRLMSWT subsystem parameter) . . . 389
U LOCK FOR RR/RS field (RRULOCK subsystem parameter) . . . 389
X LOCK FOR SEARCHED U/D field (XLKUPDLT subsystem parameter) . . . 389
START IRLM CTRACE field . . . 390
IMS BMP TIMEOUT field (BMPTOUT subsystem parameter) . . . 390

DL/I BATCH TIMEOUT field (DLITOUT subsystem parameter) . . . 391
RETAINED LOCK TIMEOUT field (RETLWAIT subsystem parameter) . . . 391
IRLM panel 2: DSNTIPJ . . . 391
PAGE PROTECT field . . . 392
MAX STORAGE FOR LOCKS field . . . 393
LOCKS PER TABLE(SPACE) field (NUMLKTS subsystem parameter) . . . 393
LOCKS PER USER field (NUMLKUS subsystem parameter) . . . 394
DEADLOCK TIME field . . . 394
DEADLOCK CYCLE field . . . 395
MEMBER IDENTIFIER field . . . 395
IRLM XCF GROUP NAME field . . . 396
LOCK ENTRY SIZE field . . . 396
NUMBER OF LOCK ENTRIES field . . . 396
DISCONNECT IRLM field . . . 397
DB2 utilities parameters panel 1: DSNTIP6 . . . 397
SYSTEM-LEVEL BACKUPS field (SYSTEM_LEVEL_BACKUPS subsystem parameter) . . . 398
RESTORE/RECOVER field (RESTORE_ RECOVER_FROMDUMP subsystem parameter) . . . 398
DUMP CLASS NAME field (UTILS_DUMP_CLASS_NAME subsystem parameter) . . . 399
MAXIMUM TAPE UNITS field (RESTORE_TAPEUNITS subsystem parameter) . . . 399
FAST REPLICATION field (CHECK_FASTREPLICATION subsystem parameter) . . . 400
FAST RESTORE field (REC_FASTREPLICATION subsystem parameter) . . . 400
FLASHCOPY PPRC field (FLASHCOPY_PPRC subsystem parameter) . . . 401
DEFAULT TEMPLATE field (FCCOPYDDN subsystem parameter) . . . 401
COPY field (FLASHCOPY_COPY subsystem parameter) . . . 402
LOAD field (FLASHCOPY_LOAD subsystem parameter). . . . 402
REORG TABLESPACE field (FLASHCOPY_REORG_TS subsystem parameter). . . . 403
REBUILD INDEX field (FLASHCOPY_REBUILD_INDEX subsystem parameter) . . . 403
REORG INDEX field (FLASHCOPY_REORG_INDEX subsystem parameter) . . . 404
DB2 utilities parameters panel 2: DSNTIP61 . . . 404
TEMP DS UNIT NAME field (VOLTDEVT subsystem parameter) . . . 405
UTIL TEMP STORCLAS field (UTIL_TEMP_STORCLAS subsystem parameter) . . . 405
STATISTICS HISTORY field (STATHIST subsystem parameter) . . . 405
STATISTICS ROLLUP field (STATROLL subsystem parameter) . . . 406
UTILITY TIMEOUT field (UTIMOUT subsystem parameter). . . . 406
UT SORT DATA SET ALLOCATION field (UTSORTAL subsystem parameter). . . . 407
IGNORE SORTNUM STAT field (IGNSORTN subsystem parameter) . . . 408
SET CHECK PENDING field (CHECK_SETCHKP subsystem parameter) . . . 409
UT DB2 SORT USE field (DB2SORT subsystem parameter) . . . 409
Protection panel: DSNTIPP . . . 410
ARCHIVE LOG RACF field (PROTECT subsystem parameter) . . . 410
USE PROTECTION field (AUTH subsystem parameter) . . . 410
PLAN AUTH CACHE field (AUTHCACH subsystem parameter) . . . 411
PACKAGE AUTH CACHE field (CACHEPAC subsystem parameter). . . . 411
ROUTINE AUTH CACHE field (CACHERAC subsystem parameter). . . . 411
AUTH EXIT LIMIT field (AEXITLIM subsystem parameter). . . . 412
Protection panel 2: DSNTIPP1 . . . 412
SYSTEM ADMIN 1 field (SYSADM subsystem parameter) . . . 413
SYSTEM ADMIN 2 field (SYSADM2 subsystem parameter) . . . 413
SYSTEM OPERATOR 1 field (SYSOPR1 subsystem parameter) . . . 414
SYSTEM OPERATOR 2 field (SYSOPR2 subsystem parameter) . . . 414
SECURITY ADMIN 1 field (SECADM1 subsystem parameter) . . . 415
SEC ADMIN 1 TYPE field (SECADM1_TYPE subsystem parameter) . . . 416
SECURITY ADMIN 2 field (SECADM2 subsystem parameter) . . . 416
SEC ADMIN 2 TYPE field (SECADM2_TYPE subsystem parameter) . . . 417
SEPARATE SECURITY field (SEPARATE_SECURITY subsystem parameter) . . . 417
UNKNOWN AUTHID field (DEFLTID subsystem parameter) . . . 418
RESOURCE AUTHID field (RLFAUTH subsystem parameter) . . . 418
BIND NEW PACKAGE field (BINDNV subsystem parameter) . . . 419
DBADM CREATE AUTH field (DBACRVW subsystem parameter) . . . 419
REVOKE DEP PRIV field (REVOKE_DEP_PRIVILEGES subsystem parameter) . . . 420
ROUTINES CREATOR field . . . 420

SEC DEF CREATOR field . . . 421
MVS PARMLIB updates panel: DSNTIPM . . . 422
SUBSYSTEM NAME field (SSID DECP value) . . . 422
COMMAND PREFIX field . . . 423
SUBSYSTEM MEMBER field . . . 424
SUBSYSTEM SEQUENCE field . . . 424
AUTH MEMBER field . . . 424
AUTH SEQUENCE field . . . 425
LINK LIST ENTRY field . . . 425
LINK LIST SEQUENCE field . . . 425
COMMAND SCOPE field . . . 425
SUPPRESS SOFT ERRORS field (SUPERRS subsystem parameter) . . . 426
Active log data set parameters: DSNTIPL . . . 426
NUMBER OF LOGS field . . . 426
OUTPUT BUFFER field (OUTBUFF subsystem parameter) . . . 427
ARCHIVE LOG FREQ field . . . 427
UPDATE RATE field . . . 427
Checkpoint parameters: DSNTIPL1 . . . 428
CHECKPOINT TYPE field (CHKTYPE subsystem parameter) . . . 428
RECORDS/CHECKPOINT field (CHKFREQ and CHKLOGR subsystem parameters) . . . 429
MINUTES/CHECKPOINT field (CHKFREQ and CHKMINS subsystem parameters) . . . 430
UR CHECK FREQ field (URCHKTH subsystem parameter) . . . 430
UR LOG WRITE CHECK field (URLGWTH subsystem parameter) . . . 431
LIMIT BACKOUT field (LBACKOUT subsystem parameter) . . . 431
BACKOUT DURATION field (BACKODUR subsystem parameter) . . . 432
RO SWITCH CHKPTS field (PCLOSEN subsystem parameter) . . . 432
RO SWITCH TIME field (PCLOSET subsystem parameter) . . . 433
LEVELID UPDATE FREQ field (DLDFREQ subsystem parameter) . . . 433
Archive log data set parameters panel: DSNTIPA . . . 434
ALLOCATION UNITS field (ALCUNIT subsystem parameter) . . . 434
PRIMARY QUANTITY field (PRIQTY subsystem parameter) . . . 434
SECONDARY QTY field (SECQTY subsystem parameter) . . . 435
CATALOG DATA field (CATALOG subsystem parameter) . . . 435
DEVICE TYPE 1 field (UNIT subsystem parameter) . . . 436
DEVICE TYPE 2 field (UNIT2 subsystem parameter) . . . 436
BLOCK SIZE field (BLKSIZE subsystem parameter) . . . 437
READ TAPE UNITS field (MAXRTU subsystem parameter) . . . 437
DEALLOC PERIOD field (DEALLCT subsystem parameter) . . . 437
RECORDING MAX field (MAXARCH subsystem parameter) . . . 438
WRITE TO OPER field (ARCWTOR subsystem parameter) . . . 438
WTOR ROUTE CODE field (ARCWRTC subsystem parameter) . . . 439
RETENTION PERIOD field (ARCRETN subsystem parameter) . . . 439
QUIESCE PERIOD field (QUIESCE subsystem parameter) . . . 440
COMPACT DATA field (COMPACT subsystem parameter) . . . 440
SINGLE VOLUME field (SVOLARC subsystem parameter) . . . 441
Databases and spaces to start automatically panel: DSNTIPS . . . 441
RESTART OR DEFER field (RESTART subsystem parameter) . . . 441
START NAMES field (ALL subsystem parameter) . . . 442
Distributed data facility panel 1: DSNTIPR . . . 442
DDF STARTUP OPTION field (DDF subsystem parameter) . . . 443
DB2 LOCATION NAME field . . . 444
DB2 NETWORK LUNAME field . . . 444
DB2 NETWORK PASSWORD field . . . 444
RLST ACCESS ERROR field (RLFERRD subsystem parameter) . . . 444
RESYNC INTERVAL field (RESYNC subsystem parameter) . . . 445
DDF THREADS field (CMTSTAT subsystem parameter) . . . 445
MAX INACTIVE DBATS field (MAXTYPE1 subsystem parameter) . . . 446
DB2 GENERIC LUNAME field . . . 447
IDLE THREAD TIMEOUT field (IDTHTOIN subsystem parameter) . . . 447
EXTENDED SECURITY field (EXTSEC subsystem parameter) . . . 448
Distributed data facility panel 2: DSNTIP5 . . . 448

DRDA PORT field . . . 449
SECURE PORT field . . . 449
RESYNC PORT field . . . 450
TCP/IP ALREADY VERIFIED field (TCPALVER subsystem parameter) . . . 450
EXTRA BLOCKS REQ field (EXTRAREQ subsystem parameter) . . . 451
EXTRA BLOCKS SRV field (EXTRASRV subsystem parameter) . . . 451
TCP/IP KEEPALIVE field (TCPKPALV subsystem parameter) . . . 451
POOL THREAD TIMEOUT field (POOLINAC subsystem parameter) . . . 452
Routine parameters panel: DSNTIPX . . . 453
WLM PROC NAME field . . . 453
NUMBER OF TCBS field . . . 453
MAX ABEND COUNT field (STORMXAB subsystem parameter) . . . 454
TIMEOUT VALUE field (STORTIME subsystem parameter) . . . 454
WLM ENVIRONMENT field (WLMENV subsystem parameter) . . . 455
MAX OPEN CURSORS field (MAX_NUM_CUR subsystem parameter) . . . 455
MAX STORED PROCS field (MAX_ST_PROC subsystem parameter) . . . 455
MAXIMUM LE TOKENS field (LEMAX subsystem parameter) . . . 455
BIF COMPATIBILITY field (BIF_COMPATIBILITY subsystem parameter) . . . 456
ADMIN SCHEDULER field (ADMTPROC subsystem parameter) . . . 458
Parameters for DB2-supplied routines panel: DSNTIPR1 . . . 458
FOR MOST DB2 ROUTINES field . . . 459
FOR ROUTINES IDENTIFIED TO PROGRAM CONTROL field . . . 460
FOR SERIALIZED ROUTINES field . . . 460
FOR DB2 UTILITIES ROUTINES field . . . 460
FOR DB2 XML PROCESSING ROUTINES field . . . 460
FOR MOST DB2 JAVA ROUTINES field . . . 461
FOR DB2 REXX ROUTINES field . . . 461
FOR UNIFIED DEBUGGER ROUTINES field . . . 461
FOR THE DB2 CICS ROUTINE field . . . 461
FOR DB2 MQSERIES ROUTINES field . . . 462
FOR DB2 WEB SERVICES ROUTINES field . . . 462
ADVANCED CONFIGURATION OPTIONS field . . . 462
DB2 routine selection panel: DSNTIPRA . . . 462
Data definition control support panel: DSNTIPZ . . . 485
INSTALL DD CONTROL SUPT field (RGFINSTL subsystem parameter) . . . 486
CONTROL ALL APPLICATIONS field (RGFDEDPL subsystem parameter) . . . 486
REQUIRE FULL NAMES field (RGFFULLQ subsystem parameter) . . . 486
UNREGISTERED DDL DEFAULT field (RGFDEFLT subsystem parameter) . . . 486
ART/ORT ESCAPE CHARACTER field (RGFESCP subsystem parameter) . . . 487
REGISTRATION OWNER field (RGFCOLID subsystem parameter) . . . 487
REGISTRATION DATABASE field (RGFDBNAM subsystem parameter) . . . 487
APPL REGISTRATION TABLE field (RGFNMPRT subsystem parameter) . . . 488
OBJT REGISTRATION TABLE field (RGFNMORT subsystem parameter) . . . 488
Job editing panel: DSNTIPY . . . 488
REMOTE LOCATION field . . . 489
Job card information field . . . 489
CLIST calculations panel 1: DSNTIPC . . . 490
DSMAX field (DSMAX subsystem parameter) . . . 491
EDM STATEMENT CACHE field (EDMSTMTC subsystem parameter) . . . 492
EDM DBD CACHE field (EDMDBDC subsystem parameter) . . . 492
EDM SKELETON POOL SIZE field (EDM_SKELETON_POOL subsystem parameter) . . . 493
EDM LIMIT BELOW THE BAR (EDMPOOL subsystem parameter) . . . 493
BUFFER POOL SIZE field . . . 494
SORT POOL SIZE field (SRTPOOL subsystem parameter) . . . 494
RID POOL SIZE field (MAXRBLK subsystem parameter) . . . 494
CLIST messages fields for panel DSNTIPC . . . 495
Storage messages fields . . . 495
CLIST calculations panel 2: DSNTIPC1 . . . 495
CLIST messages fields for panel DSNTIPC1 . . . 496
Completing the CLIST processing . . . 496
CLIST messages . . . 496

Installation jobs that the CLIST tailors . . . 497
Tailoring the installation jobs . . . 499
Job DSNTIJUZ and the subsystem parameter load module, application defaults load module, and DSNHMCID. . . . 500
The update process . . . 501
Updating parameters through the Update selection menu panel: DSNTIPB . . . 502
Updating other parameters . . . 505

Chapter 5. Installing or migrating DB2. . . . 507
Installing DB2 . . . 507
Installation step 1: Define the SMS environment for DB2 catalog and directory data sets: DSNTIJSS . . . 507
Installation step 2: Define DB2 to z/OS: DSNTIJMV . . . 508
Installation step 3: Define the ICF catalog and alias: DSNTIJCA (optional) . . . 513
Installation step 4: Define system data sets: DSNTIJIN. . . . 514
Installation step 5: Define DB2 initialization parameters: DSNTIJUZ . . . 516
Installation step 6: Initialize system data sets: DSNTIJID, DSNTIJIE, and DSNTIJIF . . . 517
Installation step 7: Define user authorization exit routines: DSNTIJEX (optional) . . . 518
Installation step 8: Record DB2 data to SMF (optional) . . . 519
Installation step 9: Establish subsystem security (optional) . . . 520
Installation step 10: Connect DB2 to TSO . . . 521
Installation step 11: Connect DB2 to IMS (optional). . . . 529
Installation step 12: Connect CICS to DB2 (optional) . . . 529
Installation step 13: Define DB2 to z/OS . . . 529
Installation step 14: Start the DB2 subsystem . . . 530
Installation step 15: Tailor the DB2 catalog: DSNTIJTC. . . . 531
Installation step 16: Create default storage group, define temporary work files, and bind DB2 REXX Language Support: DSNTIJTM . . . 532
Installation step 17: Define and bind DB2 objects: DSNTIJSG . . . 533
Installation step 18: Populate the user-maintained databases (optional) . . . 535
Installation step 19: Configure DB2 for running stored procedures and user-defined functions . . . 535
Installation step 20: Set up DB2-supplied routines . . . 547
Installation step 21: Back up the DB2 directory and catalog: DSNTIJIC . . . 556
Installation step 22: Set up the administrative task scheduler . . . 557
Installation step 23: Verify a successful installation . . . 560
Installation step 24: Enable additional capabilities for DB2 . . . 560
Migrating DB2 . . . 570
Migrating your DB2 subsystem to conversion mode (from Version 8 or Version 9.1) . . . 570
Falling back . . . 638
Remigrating . . . 647
Migrating your DB2 subsystem to new-function mode. . . . 649
Reverting to conversion mode* (from Version 8 or Version 9.1) or enabling-new-function mode* (from Version 8 or Version 9.1) . . . 665
Returning to enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode . . . 667

Chapter 6. Installing, migrating, and enabling DB2 data sharing . . . 669
Strategy for creating a data sharing group . . . 669
Subsystem parameters and application defaults in a data sharing environment . . . 670
Specifying a member's subsystem parameter load module . . . 671
The scope and uniqueness of DB2 subsystem parameters. . . . 671
Updating subsystem parameters for a member . . . 678
Application defaults parameters . . . 678
Installing a new data sharing group . . . 678
Enabling DB2 data sharing . . . 679
Running the CLIST for enabling data sharing. . . . 680
Stopping DB2 activity . . . 681
Identifying DB2 to z/OS . . . 682
Altering the current active log data sets . . . 683
Defining DB2 initialization parameters . . . 684
Recording DB2 data to SMF (optional) . . . 684
Connecting DB2 to TSO . . . 686

Defining DB2 to z/OS . . . 686
Starting the DB2 subsystem . . . 687
Creating the default storage group, defining temporary work files, and binding DB2 REXX Language Support 688
Creating an image copy of the DB2 directory and catalog (optional) . . . 690
Verifying that enabling data sharing was successful (optional) . . . 690
Adding a new member . . . 691
Running the CLIST for adding a new member . . . 691
Identifying DB2 to z/OS . . . 692
Defining system data sets . . . 694
Defining DB2 initialization parameters . . . 694
Initializing system data sets . . . 695
Defining user authorization exit routines (optional). . . . 695
Recording DB2 data to SMF (optional) . . . 695
Establishing subsystem security (optional). . . . 697
Connecting DB2 to IMS (optional) . . . 698
Connecting CICS to DB2 (optional) . . . 698
Defining DB2 to z/OS . . . 698
Starting the DB2 subsystem . . . 699
Creating the default storage group, defining temporary work files, and binding DB2 REXX Language Support 700
Creating an image copy of the DB2 directory and catalog . . . 702
Verifying that adding a data sharing member was successful . . . 703
Merging existing DB2 data into the group . . . 703
Merging subsystems . . . 703
Merging data . . . 704
Migrating an existing data sharing group to a new release . . . 711
Migrating the first member of a group to Version 10 . . . 711
Migrating subsequent members of a group to Version 10 . . . 712
Migrating a data sharing group to new-function mode . . . 722
Mixed releases in a data sharing group. . . . 723
Special considerations for WLM-managed stored procedures in coexistence . . . 723
Determining the release of a data sharing group in coexistence. . . . 724
Determining the function level of an IRLM group in coexistence . . . 725
Call attachment and TSO attachment in coexistence . . . 726
Automatic rebinds in coexistence. . . . 726
Recommendations for BIND in coexistence . . . 728
Recommendations for utilities in coexistence . . . 728
Load module names for running purchased utilities in coexistence . . . 730
Recommendation for group restart in coexistence . . . 731
Recommendation for SPUFI in coexistence . . . 731
Testing the data sharing group . . . 731
Testing group buffer pool caching . . . 731
Testing global lock serialization . . . 732
Testing concurrency . . . 732
Testing Sysplex query parallelism . . . 733
Falling back and remigrating in a data sharing environment. . . . 734
Falling back in a data sharing environment . . . 734
Remigrating in a data sharing environment . . . 735
Disabling and re-enabling DB2 data sharing . . . 736
Disabling data sharing . . . 737
Data recovery after disabling DB2 data sharing . . . 739
Re-enabling DB2 data sharing . . . 740
Renaming a member by performing an IPL . . . 740
Renaming a member by stopping DB2 . . . 741

Chapter 7. Verifying that installation or migration was successful with the sample applications . . . 743
Installation verification phases and programs. . . . 744
Planning for verification. . . . 748
COBOL options to use with the installation verification procedures . . . 749
C and C++ language options to use with the installation verification procedures . . . 750
PL/I options to use with the installation verification procedures . . . 751

Phase 0: Deleting the sample objects (DSNTEJ0) . . . 751
Phase 1: Creating and loading sample tables . . . 752
Job DSNTEJ1 . . . 752
Job DSNTEJ1L . . . 755
Job DSNTEJ1P . . . 756
Job DSNTEJ1U . . . 757
Phase 2: Testing the batch environment . . . 758
Job DSNTEJ2A . . . 758
Job DSNTEJ2C . . . 758
Job DSNTEJ2D . . . 759
Job DSNTEJ2E . . . 760
Job DSNTEJ2F . . . 760
Job DSNTEJ2H . . . 761
Job DSNTEJ2P . . . 761
Job DSNTEJ2U . . . 762
Phase 3: Testing SPUFI, DRDA access, dynamic SQL, and TSO . . . 764
Testing SPUFI . . . 765
Dynamic SQL and the ISPF/CAF application . . . 766
Jobs DSNTEJ3C and DSNTEJ3P . . . 766
Job DSNTEJ3M . . . 768
Starting an application in an ISPF/TSO environment . . . 768
Phase 4: Testing the IMS environment . . . 769
Jobs DSNTEJ4C and DSNTEJ4P . . . 769
Starting an application in an IMS environment . . . 772
Using the phone application in IMS . . . 773
Phase 5: Testing the CICS environment . . . 773
Job DSNTEJ5A . . . 774
Jobs DSNTEJ5C and DSNTEJ5P . . . 774
Starting an application in a CICS environment . . . 776
Using the phone application in CICS . . . 777
Using CICS storage-handling facilities . . . 778
Phase 6: Accessing data at a remote site . . . 778
DRDA access sample . . . 780
Job DSNTEJ6 . . . 781
Starting an application in an ISPF/TSO environment in phase 6 . . . 781
Stored procedure samples . . . 782
Phase 7: Accessing LOB data . . . 798
Job DSNTEJ7 . . . 799
Job DSNTEJ71 (optional) . . . 799
Job DSNTEJ73 . . . 800
Job DSNTEJ75 . . . 801
Job DSNTEJ76 . . . 801
Job DSNTEJ77 . . . 802
Job DSNTEJ78 . . . 802
Starting an application in an ISPF/TSO environment in phase 7 . . . 803
The sample applications . . . 803
Printing options for the sample application listings . . . 804
The sample application panels . . . 804
Working with scenarios . . . 807
Working with the project application scenario . . . 808
Working with the organization application scenario . . . 810
Working with the phone application scenario . . . 814
Working with the distributed organization application scenario . . . 818
Working with the employee resume and photo scenarios . . . 826
Edit exit routine . . . 830
Huffman compression exit routine . . . 830
Sample field procedure . . . 830
Dynamic SQL statements: DSNTESA, DSNTESQ . . . 830
DSNTESA . . . 830
DSNTESQ . . . 832
Dynamic SQL programs: DSNTIAD, DSNTEP2, DSNTIAUL . . . 832

Chapter 8. Working with additional capabilities for DB2 . . . 833
DB2-supplied stored procedures and user-defined functions . . . 833
WLM management of stored procedures . . . 839
Core WLM environments for DB2-supplied routines . . . 840
DSNTWLMB tool . . . 847
Sample SQL statements for installing DB2-supplied routines. . . . 849
Configuration control statements for DB2-supplied routines . . . 849
Disabling DB2-supplied routines for validation . . . 851
Setting up support for external SQL procedures . . . 852
Additional steps for enabling the stored procedures and objects for XML schema support . . . 853
Additional steps for enabling the administrative task scheduler and administrative enablement routines . . . 854
Additional steps for enabling the CICS transaction invocation procedure . . . 856
Additional steps for enabling the DSNAEXP EXPLAIN stored procedure . . . 857
Additional steps for enabling the common SQL API stored procedures . . . 857
Additional steps for enabling WebSphere MQ user-defined functions . . . 859
Enabling DB2 web services . . . 863
The IBM Data Server Driver for JDBC and SQLJ . . . 869
Connecting the IMS attachment facility . . . 869
Making DB2 load modules available to IMS . . . 869
Defining DB2 to IMS . . . 870
IMS attachment facility macro (DSNMAPN) . . . 875
Connecting distributed database systems . . . 877
The communications protocols . . . 877
The role of the communications database (CDB) . . . 878
The DRDA database protocol . . . 878
DDF installation and implications for DB2. . . . 882
Connecting systems with VTAM . . . 883
Customize VTAM for DB2 . . . 885
Choosing names and a password . . . 885
Defining the DB2 subsystem to VTAM . . . 888
Populating the communications database . . . 895
Starting VTAM to use DB2 . . . 899
Tuning the VTAM system . . . 899
Sample VTAM definitions to connect two DB2 subsystems . . . 911
Using the change log inventory utility to update the BSDS . . . 916
Connecting systems with TCP/IP . . . 917
TCP/IP terminology . . . 918
Enabling TCP/IP communication. . . . 919
TCP/IP limitations . . . 932
Initializing a TCP stack for use with a VIPA . . . 932
Initializing a TCP stack for use with both IPv4 and IPv6 addresses . . . 932
Two-phase commit for TCP/IP clients . . . 933
Multiple TCP/IP stacks and DB2 . . . 934
Specifying security requirements . . . 935
Configuring the DB2 command line processor . . . 935
Command line processor prerequisites . . . 935
Setting up your system to use the DB2 command line processor . . . 935
IBM Text Search for DB2 for z/OS . . . 936
Creating tables for use by an accelerator product . . . 936

Chapter 9. Working with international data . . . 939
Character conversion concepts. . . . 939
System-coded character set identifiers . . . 940
Unicode support in DB2. . . . 940
Customizing support for Unicode . . . 941
EBCDIC and ASCII support . . . 945
Implications of character conversions . . . 948
Euro symbol support . . . 948
How an entry in SYSIBM.SYSSTRINGS works with character conversion . . . 950
Conversion-related changes that require remote packages to be rebound . . . 952
Locales for uppercase and lowercase conversion of EBCDIC data . . . 952

Setting up z/OS to allow uppercase and lowercase conversion of Unicode and ASCII data 953

Information resources for DB2 for z/OS and related products 955

How to obtain DB2 information. 961

How to use the DB2 library 963

Notices 967
Programming Interface Information 968
Trademarks 969

Glossary 971

Index 1017

About this information

This information describes installation and migration of DB2® for z/OS® (DB2) in a data sharing and non-data-sharing environment.

This information assumes that your DB2 subsystem is running in Version 10 new-function mode. Generally, new functions that are described, including changes to existing functions, statements, and limits, are available only in new-function mode, unless explicitly stated otherwise. Exceptions to this general statement include optimization and virtual storage enhancements, which are also available in conversion mode unless stated otherwise. In Versions 8 and 9, most utility functions were available in conversion mode. However, for Version 10, most utility functions work only in new-function mode.

Who should read this information

This information is primarily intended for those people who are responsible for installing DB2 or setting up DB2 for distributed communications. This information is intended for those people who plan to install DB2 from the enterprise server using the installation CLIST.

This information assumes that you are familiar with:

- The basic concepts and facilities of DB2 in the z/OS environment
- The z/OS Time Sharing Option (TSO) and the z/OS Interactive System Productivity Facility (ISPF)
- The basic concepts of Structured Query Language (SQL)
- The basic concepts of Customer Information Control System (CICS®)
- The basic concepts of Information Management System (IMS™)
- How to define and allocate z/OS data sets by using z/OS job control language (JCL)
- How to use IBM® System Modification Program/Extended (SMP/E) to install IBM licensed programs

To set up DB2 for distributed communications, knowledge of Virtual Telecommunications Access Method (VTAM®) or Transmission Control Protocol/Internet Protocol (TCP/IP) is also needed.

Migration information

This information is intended to provide guidance for migrating to DB2 Version 10 from Version 8 or Version 9.1. Most of the information applies to migration from either version.

Information that is specific to migration from only Version 8 to Version 10 is identified where it occurs by the following markings:

From V8 Information about migrating from Version 8... From V8

Information that is specific to migration from only Version 9.1 to Version 10 is identified where it occurs by the following markings:

© Copyright IBM Corp. 1982, 2012

DB2 Utilities Suite

Important: In this version of DB2 for z/OS, the DB2 Utilities Suite is available as an optional product. You must separately order and purchase a license to such utilities, and discussion of those utility functions in this publication is not intended to otherwise imply that you have a license to them.

The DB2 Utilities Suite can work with DB2 Sort and the DFSORT program, which you are licensed to use in support of the DB2 utilities even if you do not otherwise license DFSORT for general use. If your primary sort product is not DFSORT, consider the following informational APARs mandatory reading:

- II14047/II14213: USE OF DFSORT BY DB2 UTILITIES
- II13495: HOW DFSORT TAKES ADVANTAGE OF 64-BIT REAL ARCHITECTURE

These informational APARs are periodically updated.

Related information

DB2 utilities packaging (Utility Guide)

Terminology and citations

When referring to a DB2 product other than DB2 for z/OS, this information uses the product's full name to avoid ambiguity.

The following terms are used as indicated:

DB2 Represents either the DB2 licensed program or a particular DB2 subsystem.

OMEGAMON®
Refers to any of the following products:
- IBM Tivoli® OMEGAMON XE for DB2 Performance Expert on z/OS
- IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS
- IBM DB2 Performance Expert for Multiplatforms and Workgroups
- IBM DB2 Buffer Pool Analyzer for z/OS

C, C++, and C language
Represent the C or C++ programming language.

CICS Represents CICS Transaction Server for z/OS.

IMS Represents the IMS Database Manager or IMS Transaction Manager.

MVS™ Represents the MVS element of the z/OS operating system, which is equivalent to the Base Control Program (BCP) component of the z/OS operating system.

RACF®
Represents the functions that are provided by the RACF component of the z/OS Security Server.

Accessibility features for DB2 10 for z/OS

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in z/OS products, including DB2 10 for z/OS. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers and screen magnifiers.
- Customization of display attributes such as color, contrast, and font size

Tip: The Information Management Software for z/OS Solutions Information Center (which includes information for DB2 10 for z/OS) and its related publications are accessibility-enabled for the IBM Home Page Reader. You can operate all features using the keyboard instead of the mouse.

Keyboard navigation

You can access DB2 10 for z/OS ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the DB2 10 for z/OS ISPF panels using TSO/E or ISPF, refer to the *z/OS TSO/E Primer*, the *z/OS TSO/E User's Guide*, and the *z/OS ISPF User's Guide*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Related accessibility information

Online documentation for DB2 10 for z/OS is available in the Information Management Software for z/OS Solutions Information Center, which is available at the following website: http://publib.boulder.ibm.com/infocenter/imzic

IBM and accessibility

See the *IBM Accessibility Center* at http://www.ibm.com/able for more information about the commitment that IBM has to accessibility.

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 for z/OS documentation. You can use the following methods to provide comments:

- Send your comments by email to db2zinfo@us.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title or a help topic title).
- You can also send comments by using the **Feedback** link at the footer of each page in the Information Management Software for z/OS Solutions Information Center at http://publib.boulder.ibm.com/infocenter/imzic.

How to read syntax diagrams

Certain conventions apply to the syntax diagrams that are used in IBM documentation.

Apply the following rules when reading the syntax diagrams that are used in DB2 for z/OS documentation:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The ►►── symbol indicates the beginning of a statement.

  The ──► symbol indicates that the statement syntax is continued on the next line.

  The ►── symbol indicates that a statement is continued from the previous line.

  The ──►◄ symbol indicates the end of a statement.
- Required items appear on the horizontal line (the main path).

```
►►─required_item──────────────────────────────────────────────────────────►◄
```

- Optional items appear below the main path.

```
►►─required_item──┬──────────────┬──────────────────────────────────────►◄
                  └─optional_item─┘
```

  If an optional item appears above the main path, that item has no effect on the execution of the statement and is used only for readability.

```
                  ┌─optional_item─┐
►►─required_item──┴───────────────┴──────────────────────────────────────►◄
```

- If you can choose from two or more items, they appear vertically, in a stack.

  If you *must* choose one of the items, one item of the stack appears on the main path.

```
►►─required_item──┬─required_choice1─┬──────────────────────────────────►◄
                  └─required_choice2─┘
```

  If choosing one of the items is optional, the entire stack appears below the main path.

```
►►─required_item──┬──────────────────┬──────────────────────────────────►◄
                  ├─optional_choice1─┤
                  └─optional_choice2─┘
```

  If one of the items is the default, it appears above the main path and the remaining choices are shown below.

```
                  ┌─default_choice──┐
►►─required_item──┴─────────────────┴──────────────────────────────────►◄
                  ├─optional_choice─┤
                  └─optional_choice─┘
```

- An arrow returning to the left, above the main line, indicates an item that can be repeated.

```
                  ┌──────────────────┐
                  ▼                  │
►►─required_item──┴─repeatable_item──┴──────────────────────────────────►◄
```

  If the repeat arrow contains a comma, you must separate repeated items with a comma.

```
                  ┌─,──────────────┐
                  ▼                │
►►─required_item──┴─repeatable_item┴──────────────────────────────────►◄
```

A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.

```
►►─required_item─┤ fragment-name ├──────────────────────────────────►◄
```

fragment-name:

```
├─required_item──┬──────────────┬──────────────────────────────────┤
                 └─optional_name┘
```

- With the exception of XPath keywords, keywords appear in uppercase (for example, `FROM`). Keywords must be spelled exactly as shown. XPath keywords are defined as lowercase names, and must be spelled exactly as shown. Variables appear in all lowercase letters (for example, *column-name*). They represent user-supplied names or values.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Chapter 1. Introduction to installation and migration

The essential steps are similar for both installing DB2 Version 10 and migrating to DB2 Version 10 conversion mode (from Version 8 or Version 9.1). From conversion mode (from Version 8 or Version 9.1), you must complete additional steps to migrate DB2 to Version 10 new-function mode.

For installation and migration, you need to complete the following procedures:

1. Estimate storage needs.
2. Determine which new functions you need.
3. If you are using distributed data, install VTAM and, optionally, TCP/IP.
4. Set up a Parallel Sysplex® if you plan to use data sharing.
5. Load the DB2 libraries (do the SMP/E steps).

   If you plan to use the callable SQL interface of DB2, see Configuring DB2 ODBC and running sample applications for the additional installation jobs that you need to run.

   If you plan to use DB2 for z/OS Java Edition, see Installing the IBM Data Server Driver for JDBC and SQLJ for additional installation jobs that you need to run.
6. If you are migrating, install needed service on the prior release. See the DB2 Program Directory for information about needed service.
7. If you are migrating, check release incompatibilities, and make the necessary changes in your applications.
8. Provide an SMS environment for managing DB2 catalog and directory data sets. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.
9. Tailor the installation or migration jobs.
10. Install DB2 Version 10 or migrate DB2 to Version 10 conversion mode (from Version 8 or Version 9.1).
11. Connect the DB2 attachment facilities.
12. Prepare DB2 for use.
13. Verify the Version 10 installation or migration to Version 10 conversion mode (from Version 8 or Version 9.1).

For more information about installing other features of DB2, see the DB2 Program Directory.

After you migrate to conversion mode (from Version 8 or Version 9.1), complete the following procedures to migrate to Version 10 new-function mode:

1. Verify that DB2 is stable in conversion mode (from Version 8 or Version 9.1). For data sharing, verify that all members are stable.
2. Tailor the enabling-new-function jobs.
3. Change to enabling-new-function mode (from Version 8 or Version 9.1) to convert the catalog.
4. Start new-function mode.

© Copyright IBM Corp. 1982, 2012

If you have problems during or after migration to Version 10 conversion mode (from Version 8 or Version 9.1), you can complete the following procedures:

1. Fall back to your previous version (Version 8 or Version 9.1).
2. Remigrate to DB2 Version 10 conversion mode (from Version 8 or Version 9.1).

If you have problems after migrating to enabling-new-function mode (from Version 8 or Version 9.1), you can revert to Version 10 conversion mode* (from Version 8 or Version 9.1). If you have problems after migrating to new-function mode, you can revert to either Version 10 conversion mode* (from Version 8 or Version 9.1) or Version 10 enabling-new-function mode* (from Version 8 or Version 9.1).

Related concepts:

Chapter 6, "Installing, migrating, and enabling DB2 data sharing," on page 669

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Installing the RACF access control module (RACF Access Control Module Guide)

Installing OmniFind Text Search Server (IBM OmniFind Text Search Server Installation, Administration, and Reference)

Related reference:

z/OS Parallel Sysplex Configuration Overview

Related information:

Getting started with IBM Spatial Support for DB2 for z/OS (IBM Spatial Support User's Guide and Reference)

Installation and migration tools

DB2 includes several tools and capabilities that help you complete the steps that are involved in installing or migrating to Version 10.

Installation and migration tools: DB2 provides a set of tools that automate the process of installing or migrating. These tools include:

- Most of the job control language (JCL) that is needed to install and migrate a release of DB2

  This JCL constitutes the installation and migration jobs. Each of these jobs helps you perform an installation or migration task.
- The installation CLIST (command list) to help tailor the installation and migration jobs

  This CLIST is also called the *migration CLIST*, or simply the *CLIST*. It contains the necessary code for tailoring the jobs to suit your needs.

- A series of ISPF panels that you can use to pass information to the CLIST

  With Interactive Systems Productivity Facility (ISPF) and Interactive Systems Productivity Facility/Program Development Facility (ISPF/PDF), you can use a series of ISPF panels to pass parameter values to the CLIST. The CLIST uses these values to tailor the installation and migration jobs. This process is called the ISPF *tailoring session*.
- Sample applications to help determine if you installed or migrated DB2 correctly

  DB2 provides a set of sample programs and procedures that help you determine if DB2 is operating correctly.

All references to SYS1.PARMLIB also imply the logical PARMLIB data set that is used for DB2.

Minimal assemblies: Because it is distributed as object code, DB2 requires few assemblies. You must complete an assembly to specify DB2 initialization parameters, but this assembly requires only a few seconds.

Ability to defer decisions about DB2 characteristics: DB2 allows you to specify many subsystem characteristics during DB2 operation. You can modify initialization parameters, authorize users, define databases and tables, and tune DB2 while DB2 is running. Therefore, you can defer many decisions until after you finish installing or migrating DB2.

Ability to update installation and migration options: During the process of installing and migrating, DB2 uses ISPF panels to prompt you for many options. DB2 allows you to update most of these options without requiring you to reinstall or remigrate. You can accept the default values for certain options and, after acquiring experience with DB2, tailor them to suit your needs.

Related tasks:

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Related reference:

"Directory of subsystem parameters and application default values" on page 242

Introduction to installation

Installation is the process of preparing DB2 to operate as a z/OS subsystem.

Summary of installation steps

After you complete the SMP/E steps and run the installation CLIST, you can edit and run the jobs that install your DB2 Version 10 subsystem. A new installation starts in new-function mode.

The following table identifies the steps that you perform to install DB2 Version 10.

Table 1. Overview of steps for installing DB2 Version 10

Step	Description	Job
1	Define the SMS environment for DB2 catalog and directory data sets.	DSNTIJSS
2	Define DB2 Version 10 to z/OS, and build cataloged procedures.	DSNTIJMV

Table 1. Overview of steps for installing DB2 Version 10 (continued)

Step	Description	Job
3	Optionally, define a new integrated catalog facility (ICF) catalog and alias.	DSNTIJCA
4	Define DB2 data sets.	DSNTIJIN
5	Define DB2 initialization parameters.	DSNTIJUZ
6	Initialize DB2 catalog and directory data sets.	DSNTIJID, DSNTIJIE, and DSNTIJIF
7	Optionally, prepare authorization exit routines.	DSNTIJEX
8	Optionally, prepare for SMF recording.	(none)
9	Optionally, establish subsystem security.	(none)
10	Establish the TSO environment for DB2.	DSNTIJVC
11	Optionally, connect DB2 to IMS.	(none)
12	Optionally, connect CICS to DB2.	(none)
13	IPL z/OS.	(none)
14	Start DB2 Version 10.	(none)
15	Tailor the DB2 catalog.	DSNTIJTC
16	Define temporary work file table spaces and initial buffer pool sizes. Bind the packages for DB2 REXX Language Support.	DSNTIJTM
17	Define and bind DB2 objects and user-maintained databases.	DSNTIJSG
18	Optionally, populate the user-maintained databases, and, if you are using DDF, populate the communications database (within the DB2 catalog).	(none)
19	Configure DB2 for running stored procedures and user-defined functions.	(none)
20	Set up DB2-supplied routines.	DSNTIJRW, DSNTIJRT, and DSNTIJRV
21	Take an image copy of the DB2 catalog and directory.	DSNTIJIC
22	Set up the administrative task scheduler.	(none)
23	Run the installation verification procedure.	DSNTEJ*xx*

Related concepts:

Chapter 3, “Loading the DB2 libraries,” on page 207

Related tasks:

Chapter 2, “Preparing your system to install or migrate DB2,” on page 27

Chapter 4, “Tailoring DB2 jobs to your environment using the installation CLIST,” on page 219

“Installing DB2” on page 507

Introduction to migration from DB2 Version 8

Migration is the process of upgrading from one DB2 release to a later release. Migration also refers to the process of changing from one DB2 mode to the next mode.

You can migrate a DB2 Version 8 subsystem that is in new-function mode to Version 10 without starting the subsystem in Version 9.1.

Important restrictions:

- A data sharing group that is migrated from Version 8 to Version 10 cannot have any Version 9.1 members.
- If you start a Version 8 subsystem migration to Version 10, you can fall back to only Version 8.
- If you start a Version 8 subsystem migration to Version 10 and then fall back to Version 8, you cannot subsequently migrate the subsystem to Version 9.1.

Related concepts:

"Availability of new functions in conversion mode from Version 8 and new-function mode" on page 65

Migration modes for DB2 8 to DB2 10

Migration from Version 8 to Version 10 new-function mode includes five DB2 modes. Four of these modes are differentiated from the modes that are part of migration from Version 9.1 to Version 10.

The names of the modes during migration from Version 8 are:

- Conversion mode from Version 8
- Conversion mode* from Version 8
- Enabling-new-function mode from Version 8
- Enabling-new-function mode* from Version 8
- New-function mode

New-function mode is not differentiated by the release from which you are migrating.

Each of the DB2 modes has a corresponding short name that is used in the DB2 output. The short names are also used in some topics to improve readability where multiple mode names are mentioned. The following table shows the short names of the DB2 modes for migration from Version 8 and the corresponding long names.

Table 2. Long and short names for migration modes

Short name for migration mode	Long name for migration mode
CM8	Conversion mode from Version 8
CM8*	Conversion mode* from Version 8
ENFM8	Enabling-new-function mode from Version 8
ENFM8*	Enabling-new-function mode* from Version 8
NFM	New-function mode

To see the current DB2 mode of a subsystem, issue a DISPLAY GROUP command. The output shows the short name for the migration mode. The definition of each mode is as follows:

Conversion mode from Version 8
The state of the catalog after the Version 10 migration process is complete. In a DB2 data sharing group, members in CM8 can coexist with members

that are still in Version 8. New Version 10 functions are not available for use in CM8. A subsystem that is in CM8 can fall back to Version 8, but not to Version 9.1.

Conversion mode* from Version 8
A state that is similar to CM8, but the asterisk (*) indicates that, at one time, the DB2 subsystem or data sharing group was in ENFM8, ENFM8*, or NFM. You can still access objects that were created in ENFM8 or NFM. Data sharing groups that have a member in this mode cannot have any Version 8 members. A subsystem that is in CM8* cannot fall back to Version 8.

Enabling-new-function mode from Version 8
A transitional state that indicates that the DB2 subsystem or data sharing group is in the process of enabling new function. New Version 10 functions are not available in ENFM8. A subsystem that is in ENFM8 cannot return to CM8, but it can revert to CM8*. A subsystem that is in ENFM8 cannot fall back to Version 8 or coexist with a Version 8 subsystem.

Enabling-new-function mode* from Version 8
A transitional state that is similar to ENFM8, but the asterisk (*) indicates that, at one time, the DB2 subsystem or data sharing group was in NFM. In ENFM8*, you can still access objects that were created in NFM, but you cannot create new objects. A subsystem that is in ENFM8* can revert to CM8*, but it cannot fall back to Version 8 or coexist with a Version 8 subsystem.

New-function mode
The state of the catalog after enabling-new-function mode processing is complete and job DSNTIJNF runs successfully. The catalog has been marked as being in new-function mode. All new Version 10 and Version 9.1 functions are available in NFM. A subsystem that was migrated to Version 10 NFM from CM8 can revert to CM8* or ENFM8*. However, it cannot fall back to Version 8 or coexist with a Version 8 subsystem.

The process of migrating from DB2 Version 8 to Version 10 begins with migration to CM8. From there, you can migrate to ENFM8 and then to NFM.

The flow from one mode to another during migration is illustrated in the following figure:

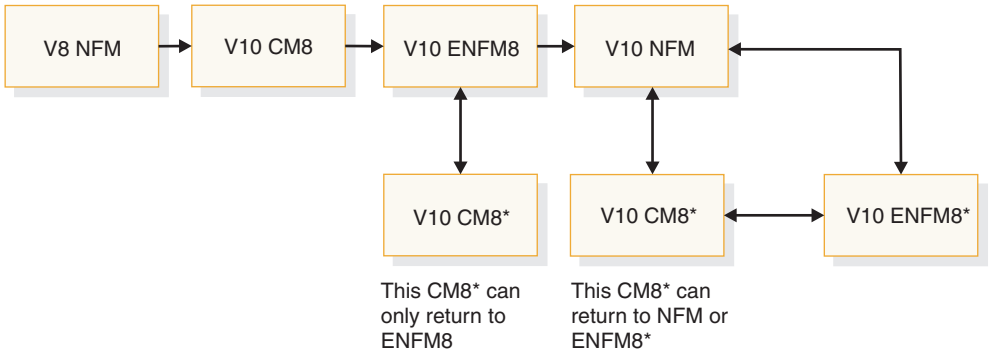


Figure 1. The flow from one mode to another during migration

CM8* and ENFM8* are optional modes that you can revert to if you encounter problems in another mode or if you want to temporarily prevent the use of new function.

Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

Related tasks:

"Define the SMS environment for the DB2 catalog and directory data sets (DSNTIJSS)" on page 576

Related reference:

-DISPLAY GROUP (DB2) (DB2 Commands)

Migration to conversion mode from Version 8: Summary

After you load the DB2 libraries and run the installation CLIST, you can migrate your DB2 Version 8 subsystem to Version 10. When you finish migration, your subsystem is in Version 10 conversion mode from Version 8.

Migration to Version 10 conversion mode from Version 8 includes the steps in the following table.

Table 3. Overview of steps for migrating to DB2 Version 10 conversion mode from Version 8

Step	Description	Job
1	Check for changes to DB2 that might impact your migration to Version 10 conversion mode from Version 8. Make adjustments for release incompatibilities. Perform premigration activities to determine unsupported objects.	DSNTIJPM
2	Optionally, run DSN1COPY with the CHECK option on the catalog table spaces, and invoke DSN1CHKR to check for broken links on your Version 8 subsystem.	none
3	Optionally, determine which packages will be invalid after migration.	none
4	Optionally, check for consistency between catalog tables.	none
5	Take an image copy of your Version 8 catalog.	DSNTIJIC
6	Establish the TSO environment for DB2.	DSNTIJVC
7	Optionally, connect DB2 to IMS.	none
8	Optionally, connect CICS to DB2.	none
9	Stop your DB2 Version 8 subsystem.	none
10	Optionally, back up your Version 8 volumes.	none
11	Define DB2 initialization parameters.	DSNTIJUZ
12	Optionally, establish subsystem security.	none
13	Define DB2 Version 10 to z/OS, and build cataloged procedures.	DSNTIJMV
14	Define system data sets.	DSNTIJIN
15	Define authorization exit routines.	DSNTIJEX

Table 3. Overview of steps for migrating to DB2 Version 10 conversion mode from Version 8 (continued)

Step	Description	Job
16	IPL z/OS.[1]	none
17	Start DB2 Version 10.	none
18	Tailor the DB2 Version 10 catalog.	DSNTIJTC
19	Optionally, invoke DSN1CHKR to check for broken links on Version 10. Optionally, run job DSNTIJCX to check indexes in the catalog and directory.	none
20	Optionally, rebuild indexes that are put in the informational rebuild-pending state during migration to DB2 Version 10.	DSNTIJRI
21	Enable change data capture.	none
22	Prepare the dynamic SQL program and bind the packages for DB2 REXX Language Support. Optionally, create additional table spaces in the work file database.	DSNTIJTM
23	Set up DB2-supplied applications and user-maintained databases.	DSNTIJSG
24	Migrate EXPLAIN tables to Version 10 format and Unicode encoding.	DSNTIJXA, DSNTIJXB, and DSNTIJXC
25	Configure the DB2 environment for running stored procedures and user-defined functions.	none
26	Set up DB2-supplied routines.	DSNTIJRW, DSNTIJRT, and DSNTIJRV
27	Verify views.	none
28	Take an image copy of the Version 10 catalog.	DSNTIJIC
29	Set up the administrative task scheduler.	none
30	Optionally, run select Version 8 verification jobs.	DSNTEJ*xx*
31	Optionally, run RUNSTATS to collect improved statistics.	none

Note:

1. Optional if no PARMLIB updates exist and if early code is at the correct level.

Related concepts:

Chapter 3, "Loading the DB2 libraries," on page 207

"Migration to enabling-new-function mode from Version 8: Summary" on page 10

Related tasks:

"Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14

"Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15

Chapter 2, "Preparing your system to install or migrate DB2," on page 27

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

"Migrating your DB2 subsystem to conversion mode (from Version 8 or Version 9.1)" on page 570

Falling back to Version 8: Summary

Fallback is the process of returning to your previous DB2 release after successfully completing the catalog migration to Version 10 conversion mode (from Version 8 or Version 9.1). A successful migration means that migration job DSNTIJTC completed successfully. You can fall back to Version 8 only if you migrated from Version 8.

You can fall back to Version 8 if a severe error occurs either during the subsequent migration steps or during operation of DB2 Version 10 in conversion mode from Version 8. You cannot fall back from any of the following Version 10 modes:

- conversion mode* from Version 8
- enabling-new-function mode from Version 8
- enabling-new-function mode* from Version 8
- new-function mode

To fall back to Version 8, complete the steps in the following table.

Table 4. Overview of steps to fall back to Version 8

Step	Description	Job
1	Stop DB2 Version 10.	(none)
2	Rename the cataloged procedures.	DSNTIJFV
3	Reconnect TSO, IMS, and CICS to DB2 Version 8.	(none)
4	Start Version 8.	(none)
5	Modify your DSNTIJIC job for use during fallback.	(none)
6	Run select Version 8 installation verification jobs. These jobs should be the jobs that you ran to verify your migration to conversion mode from Version 8.	DSNTEJ*xx*

Related concepts:

"Implications of falling back" on page 638

Related tasks:

"Fallback procedure" on page 641

"Falling back in a data sharing environment" on page 734

Remigration from Version 8: Summary

Remigration, which is migration to DB2 Version 10 after falling back to your previous version, is simpler than the initial migration process.

The basic steps for remigrating to Version 10 from Version 8 are as follows.

Table 5. Overview of steps for remigration to DB2 Version 10

Step	Description	Job
1	Run DSN1COPY with the CHECK option on the catalog table spaces. Invoke DSN1CHKR to check for broken links on your Version 8 system. Run the queries in DSNTESQ to check for consistency between catalog tables.	(none)
2	Take an image copy of your Version 8 catalog.	DSNTIJIC
3	Stop your Version 8 system.	(none)
4	Reconnect TSO, IMS, and CICS to DB2 Version 10.	(none)
5	Rename cataloged procedures.	DSNTIJFV
6	Start DB2 Version 10.	(none)
7	Optionally, take an image copy of the Version 10 catalog.	DSNTIJIC
8	Optionally, run select Version 8 verification jobs.	DSNTEJ*xx*

You must perform some additional tasks manually.

Related tasks:

"Remigrating" on page 647

Migration to enabling-new-function mode from Version 8: Summary

After you migrate to Version 10 conversion mode from Version 8, you must migrate to enabling-new-function mode from Version 8 before you can activate new-function mode.

Migration to Version 10 enabling-new-function mode from Version 8 includes the steps in the following table.

Table 6. Overview of steps for migrating to DB2 Version 10 enabling-new-function mode from Version 8

Step	Description	Job
1	Run installation CLIST using ENFM option.	(none)
2	Take an image copy of your Version 10 catalog.	DSNTIJIC

Table 6. Overview of steps for migrating to DB2 Version 10 enabling-new-function mode from Version 8 (continued)

Step	Description	Job
3	Convert the DB2 Version 10 catalog and directory. Optionally, run job DSNTIJNH to stop enabling-new-function mode processing before job DSNTIJEN completes.	DSNTIJEN

Related tasks:

"Migrating from conversion mode (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)" on page 653

"Completing enabling-new-function mode (from Version 8 or Version 9.1) processing" on page 654

Migration from enabling-new-function mode from Version 8 to new-function mode: Summary

After you migrate to Version 10 enabling-new-function mode from Version 8, you can migrate to Version 10 new-function mode.

Migration to Version 10 new-function mode includes the steps in the following table.

Table 7. Overview of steps for migrating to DB2 Version 10 new-function mode

Step	Description	Job
1	Enter new-function mode.	DSNTIJNF
2	Optionally, check for consistency between catalog tables.	DSNTESQ
3	Optionally, stop DB2 Version 10.	(none)
4	Create the application defaults load module, *dsnhdecp*, for new-function mode.	DSNTIJNG
5	If you stopped DB2 before running DSNTIJNG, start DB2 Version 10.	(none)
6	Create user-managed objects that require new-function mode.	DSNTIJNX
7	Install DB2-supplied routines that require new-function mode.	DSNTIJRT
8	Validate the installation of DB2-supplied routines.	DSNTIJRV
9	Convert remaining EXPLAIN tables to Version 10 format.	DSNTIJXA
10	Run Version 10 verification jobs.	DSNTEJ*xx*

Related concepts:

"Implications of migrating to Version 10 new-function mode" on page 650

Related tasks:

"Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode" on page 661

"Converting EXPLAIN tables for migration from Version 8" on page 664

Reversion to conversion mode* from Version 8: Summary

You can revert to conversion mode* from Version 8 if you encounter problems after migrating to certain Version 10 modes.

About this task

If you encounter problems in any of the following Version 10 modes, you can revert to conversion mode* from Version 8 until the problems are resolved:

- enabling-new-function mode from Version 8
- enabling-new-function mode* from Version 8
- new-function mode

Procedure

To revert from any of these modes to conversion mode* from Version 8:

Run job DSNTIJCS.

Related tasks:

"Reverting to conversion mode* (from Version 8 or Version 9.1)" on page 665

Reversion to enabling-new-function mode* from Version 8: Summary

If you want to temporarily prevent the use of new function after you migrate to Version 10 new-function mode, you can revert to enabling-new-function mode* from Version 8.

Procedure

To revert from Version 10 new-function mode to enabling-new-function mode* from Version 8:

Run job DSNTIJES.

Related tasks:

"Reverting to enabling-new-function mode* (from Version 8 or Version 9.1)" on page 666

Migration from conversion mode* from Version 8 to enabling-new-function mode from Version 8: Summary

After you revert to Version 10 conversion mode* from Version 8, you can migrate to enabling-new-function mode from Version 8.

About this task

The following table identifies the migration modes that are mentioned in this topic, along with the short names that are used to refer to those modes.

Table 8. Long and short names for migration modes

Short name for migration mode	Long name for migration mode
CM8*	Conversion mode* from Version 8
ENFM8	Enabling-new-function mode from Version 8
ENFM8*	Enabling-new-function mode* from Version 8
NFM	New-function mode

Procedure

To migrate to enabling-new-function mode from Version 8 from CM8*:

Run job DSNTIJEN.
Whether job DSNTIJEN changes DB2 to ENFM8* or ENFM8 depends on the state of DB2 before you reverted to CM8*. If you reverted to CM8* from NFM or ENFM8*, job DSNTIJEN changes DB2 to ENFM8*. If you reverted to CM8* from ENFM8, job DSNTIJEN changes DB2 to ENFM8.

Related tasks:

"Moving from conversion mode* (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)" on page 667

Migration from conversion mode* from Version 8 to new-function mode: Summary

If your subsystem is in Version 10 conversion mode* from Version 8, you can migrate to Version 10 new-function mode.

Procedure

To migrate from Version 10 conversion mode* from Version 8 to Version 10 new-function mode:

Run job DSNTIJNF. If enabling-new-function mode from Version 8 processing did not complete when DB2 was previously in enabling-new-function mode from Version 8, DSNTIJNF issues an error that indicates that you need to run DSNTIJEN.

Related tasks:

"Moving from conversion mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

Migration from enabling-new-function mode* from Version 8 to new-function mode: Summary

After you revert to Version 10 enabling-new-function mode* from Version 8, you can migrate to Version 10 new-function mode.

Procedure

To migrate to Version 10 new-function mode from enabling-new-function mode* from Version 8:

Run job DSNTIJNF.

Related tasks:

"Migrating from enabling-new-function mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8

Before you migrate to DB2 Version 10 conversion mode from Version 8, complete the premigration activities to prepare your subsystem.

Procedure

__ 1. Complete the following tasks in any order.
  - __ • Ensure that you meet the hardware and software requirements.
  - __ • Plan for storage requirements.
  - __ • Plan for performance.
  - __ • Convert EXPLAIN tables to Version 8 format and Unicode encoding.
  - __ • Make adjustments for the following changes in Version 10:
    - __ – Application and SQL release incompatibilities
    - __ – Utility release incompatibilities
    - __ – Command release incompatibilities
    - __ – Security release incompatibilities
    - __ – Storage release incompatibilities
    - __ – Other release incompatibilities
    - __ – Deprecated functions
    - __ – Functions that are no longer supported

__ 2. Use SMP/E to load the DB2 libraries.

__ 3. Complete the following tasks in any order.
  - __ • Tailor the installation jobs by running the installation CLIST in migration mode.
  - __ • Run premigration query DSNTIJPM.
  - __ • Ensure that the data sets for your DB2 catalog and directory reside on SMS-managed storage.
  - __ • Save your critical access paths.
  - __ • Examine new or changed DB2I default panel values to ensure that your customized values are still valid.
  - __ • Ensure that sample objects from your Version 8 release are still available.
  - __ • Make a backup copy of the BSDS.
  - __ • Change stored procedures that specify DSNWZPR as the external module name to specify DSNWZP.

__ 4. Immediately before starting the migration tasks, ensure that no utility jobs are running.

What to do next

See "Checklist for migration to DB2 Version 10 conversion mode from Version 8" for the list of migration tasks to complete after premigration.

Checklist for migration to DB2 Version 10 conversion mode from Version 8

After you complete the premigration tasks to prepare your Version 8 subsystem, you can continue with migrating to DB2 Version 10 conversion mode from Version 8.

About this task

Complete these steps in the order that they are listed.

Procedure

__ • Complete premigration activities.
__ • Optional: Run the link checker on DB2 table spaces.
__ • Optional: Determine which packages are invalid after migration.
__ • Optional: Check for consistency between catalog tables.
__ • Take image copies of the directory and catalog: job DSNTIJIC.
__ • Connect DB2 to TSO.
__ • Optional: Connect IMS to DB2.
__ • Optional: Connect CICS to DB2.
__ • Stop your DB2 Version 8 subsystem.
__ • Optional: Back up your Version 8 volumes.
__ • Define DB2 initialization parameters: job DSNTIJUZ.
__ • Optional: Establish subsystem security.
__ • Define DB2 Version 10 to z/OS, and build cataloged procedures: job DSNTIJMV.
__ • Define system data sets: job DSNTIJIN.
__ • Optional: Define authorization exit routines: job DSNTIJEX.

  You can skip this step if you did not run job DSNTIJEX for your Version 8 subsystem.

__ • Optional: IPL z/OS.

  You can skip this step if no PARMLIB updates exist and your early code is at the correct level.

__ • Start DB2 Version 10.
__ • Tailor the DB2 Version 10 catalog: job DSNTIJTC.
__ • Optional: Check the integrity of the catalog and directory.
__ • Optional: Rebuild indexes that are put in the informational rebuild-pending state during migration to DB2 Version 10: job DSNTIJRI.
__ • Enable change data capture.
__ • Prepare the dynamic SQL program and bind the packages for DB2 REXX Language Support. Optionally, create additional table spaces in the work file database: job DSNTIJTM.
__ • Rebind SPUFI packages and packages that are defined by IBM: job DSNTIJSG.
__ • Migrate EXPLAIN tables to the current format and encoding type.

__ • Configure DB2 for running stored procedures and user-defined functions.
__ • Set up DB2-supplied routines and supporting objects: jobs DSNTIJRW, DSNTIJRT and DSNTIJRV.
__ • Verify views.
__ • Take an image copy of the Version 10 catalog: job DSNTIJIC.
__ • Set up the administrative task scheduler.
__ • Optional: Run Version 8 verification jobs: jobs DSNTEJ*xx*.
__ • Optional: Collect improved statistics.

Introduction to migration from DB2 Version 9.1

Migration is the process of upgrading from one DB2 release to a later release. Migration also refers to the process of changing from one DB2 mode to the next mode.

Important restrictions:

- A data sharing group that is migrated from Version 9.1 to Version 10 cannot have any Version 8 members.
- If you start a Version 9.1 subsystem migration to Version 10, you can fall back to only Version 9.1.
- If you start a Version 8 subsystem migration to Version 10 and then fall back to Version 8, you cannot subsequently migrate the subsystem to Version 9.1.

Related concepts:

"Availability of new functions in conversion mode from Version 9.1 and new-function mode" on page 114

Migration modes for DB2 9 to DB2 10

Migration from Version 9.1 to Version 10 new-function mode includes five DB2 modes. Four of these modes are differentiated from the modes that are part of migration from Version 8 to Version 10.

The names of the modes during migration from Version 9.1 are:

- Conversion mode from Version 9.1
- Conversion mode* from Version 9.1
- Enabling-new-function mode from Version 9.1
- Enabling-new-function mode* from Version 9.1
- New-function mode

New-function mode is not differentiated by the release from which you are migrating.

Each of the DB2 modes has a corresponding short name that is used in the DB2 output. The short names are also used in some topics to improve readability where multiple mode names are mentioned. The following table shows the short names of the DB2 modes for migration from Version 9.1 and the corresponding long names.

Table 9. Long and short names for migration modes

Short name for migration mode	Long name for migration mode
CM9	Conversion mode from Version 9.1

Table 9. Long and short names for migration modes (continued)

Short name for migration mode	Long name for migration mode
CM9*	Conversion mode* from Version 9.1
ENFM9	Enabling-new-function mode from Version 9.1
ENFM9*	Enabling-new-function mode* from Version 9.1
NFM	New-function mode

To see the current mode of a DB2 subsystem, issue a DISPLAY GROUP command. The output shows the short name for the migration mode. The definition of each mode is as follows:

Conversion mode from Version 9.1
The state of the catalog after the Version 10 migration process is complete. In a DB2 data sharing group, members in CM9 can coexist with members that are still in Version 9.1. New Version 10 functions are not available for use in CM9. A subsystem that is in CM9 can fall back to only Version 9.1.

Conversion mode* from Version 9.1
A state that is similar to CM9, but the asterisk (*) indicates that, at one time, the DB2 subsystem or data sharing group was in ENFM9, ENFM9* or NFM. You can still access objects that were created in ENFM9 or NFM. Data sharing groups that have a member in this mode cannot have any Version 9.1 members. A subsystem that is in CM9* cannot fall back to Version 9.1.

Enabling-new-function mode from Version 9.1
A transitional state that indicates that the DB2 subsystem or data sharing group is in the process of enabling new function. New Version 10 functions are not available in ENFM9. A subsystem that is in ENFM9 cannot return to CM9, but it can revert to CM9*. A subsystem that is in ENFM9 also cannot fall back to Version 9.1 or coexist with a Version 9.1 subsystem.

Enabling-new-function mode* from Version 9.1
A transitional state that is similar to ENFM9, but the asterisk (*) indicates that, at one time, the DB2 subsystem or data sharing group was in NFM. In ENFM9*, you can still access objects that were created in NFM, but you cannot create new objects. A subsystem that is in ENFM9* can revert to CM9*, but it cannot fall back to Version 9.1 or coexist with a Version 9.1 subsystem.

New-function mode
The state of the catalog after enabling-new-function mode processing is complete and job DSNTIJNF runs successfully. The catalog has been marked as being in new-function mode. All new Version 10 functions are available in NFM. A subsystem that was migrated to Version 10 NFM from CM9 can revert to CM9* or ENFM9*. However, the subsystem cannot fall back to Version 9.1 or coexist with a Version 9.1 subsystem.

The process of migrating from DB2 Version 9.1 to Version 10 begins with migration to CM9. From there, you can migrate to ENFM9 and then to NFM.

The flow from one migration mode to another is illustrated in the following figure:

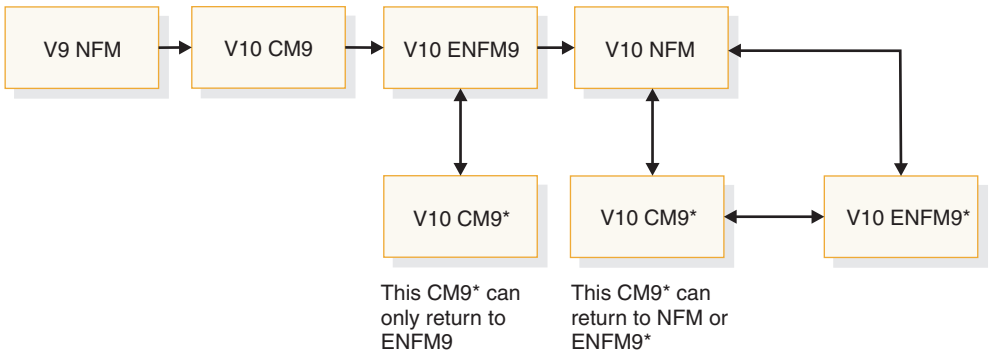


Figure 2. The flow from one mode to another during migration

CM9* and ENFM9* are optional modes that you can revert to if you encounter problems in another mode or if you want to temporarily prevent the use of new function.

Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

Related tasks:

"Define the SMS environment for the DB2 catalog and directory data sets (DSNTIJSS)" on page 576

Related reference:

-DISPLAY GROUP (DB2) (DB2 Commands)

Migration to conversion mode from Version 9.1: Summary

After you load the DB2 libraries and run the installation CLIST, you can migrate your DB2 Version 9.1 subsystem to Version 10. When you finish migration, your subsystem is in Version 10 conversion mode from Version 9.1.

Migration to Version 10 conversion mode from Version 9.1 includes the steps in the following table.

Table 10. Overview of steps for migrating to DB2 Version 10 conversion mode from Version 9.1

Step	Description	Job
1	Check for changes to DB2 that might impact your migration to Version 10 conversion mode. Make adjustments for release incompatibilities. Perform premigration activities to determine unsupported objects.	DSNTIJPM
2	Optionally, run DSN1COPY with the CHECK option on the catalog table spaces, and invoke DSN1CHKR to check for broken links on your Version 9.1 subsystem.	none

Table 10. Overview of steps for migrating to DB2 Version 10 conversion mode from Version 9.1 (continued)

Step	Description	Job
3	Optionally, determine which packages will be invalid after migration.	none
4	Optionally, check for consistency between catalog tables.	none
5	Take an image copy of your Version 9.1 catalog.	DSNTIJIC
6	Establish the TSO environment for DB2.	DSNTIJVC
7	Optionally, connect DB2 to IMS.	none
8	Optionally, connect CICS to DB2.	none
9	Stop your DB2 Version 9.1 subsystem.	none
10	Optionally, back up your Version 9.1 volumes.	none
11	Define DB2 initialization parameters.	DSNTIJUZ
12	Optionally, establish subsystem security.	none
13	Define DB2 Version 10 to z/OS, and build cataloged procedures.	DSNTIJMV
14	Define system data sets.	DSNTIJIN
15	Define authorization exit routines.	DSNTIJEX
16	IPL z/OS.[1]	none
17	Start DB2 Version 10.	none
18	Tailor the DB2 Version 10 catalog.	DSNTIJTC
19	Optionally, invoke DSN1CHKR to check for broken links on Version 10. Optionally, run job DSNTIJCX to check indexes in the catalog and directory.	none
20	Optionally, rebuild indexes that are put in the informational rebuild-pending state during migration to DB2 Version 10.	DSNTIJRI
21	Enable change data capture.	none
22	Prepare the dynamic SQL program and bind the packages for DB2 REXX Language Support. Optionally, create additional table spaces in the work file database.	DSNTIJTM
23	Set up DB2-supplied applications and user-maintained databases.	DSNTIJSG
24	Migrate EXPLAIN tables to Version 10 format and Unicode encoding.	DSNTIJXA, DSNTIJXB, and DSNTIJXC
25	Configure DB2 for running stored procedures and user-defined functions.	none
26	Set up DB2-supplied routines.	DSNTIJRW, DSNTIJRT, and DSNTIJRV
27	Verify views.	none
28	Take an image copy of the Version 10 catalog.	DSNTIJIC
29	Set up the administrative task scheduler.	none
30	Optionally, run select Version 9.1 verification jobs.	DSNTEJ*xx*

Table 10. Overview of steps for migrating to DB2 Version 10 conversion mode from Version 9.1 (continued)

Step	Description	Job
31	Optionally, run RUNSTATS to collect improved statistics. You can skip this step if your Version 9 RUNSTATS procedures included the KEYCARD option.	none

Note:

1. Optional if no PARMLIB updates exist and if early code is at the correct level.

Related concepts:

Chapter 3, "Loading the DB2 libraries," on page 207

"Migration to enabling-new-function mode from Version 9.1: Summary" on page 21

Related tasks:

"Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24

"Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25

Chapter 2, "Preparing your system to install or migrate DB2," on page 27

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

"Migrating your DB2 subsystem to conversion mode (from Version 8 or Version 9.1)" on page 570

Falling back to Version 9.1: Summary

Fallback is the process of returning to your previous DB2 release after successfully completing the catalog migration to Version 10 conversion mode (from Version 8 or Version 9.1). A successful migration means that migration job DSNTIJTC completed successfully. You can fall back to Version 9.1 only if you migrated from Version 9.1.

You can fall back to DB2 Version 9.1 if a severe error occurs either during the subsequent migration steps or during operation of Version 10 in conversion mode from Version 9.1. You cannot fall back from any of the following Version 10 modes:

- Conversion mode* from Version 9.1
- Enabling-new-function mode from Version 9.1
- Enabling-new-function mode* from Version 9.1
- New-function mode

To fall back to Version 9.1, complete the steps in the following table.

Table 11. Overview of steps to fall back to Version 9.1

Step	Description	Job
1	Stop DB2 Version 10.	(none)
2	Rename the cataloged procedures.	DSNTIJFV
3	Reconnect TSO, IMS, and CICS to DB2 Version 9.1.	(none)
4	Start Version 9.1.	(none)
5	Modify your DSNTIJIC job for use during fallback.	(none)

Table 11. Overview of steps to fall back to Version 9.1 (continued)

Step	Description	Job
6	Run select Version 9.1 installation verification jobs. These jobs should be the jobs that you ran to verify your migration to conversion mode from Version 9.1.	DSNTEJ*xx*

Related concepts:

"Implications of falling back" on page 638

Related tasks:

"Fallback procedure" on page 641

"Falling back in a data sharing environment" on page 734

Remigration from Version 9.1: Summary

Remigration, which is migration to DB2 Version 10 after falling back to your previous version, is simpler than the initial migration process.

The basic steps for remigrating to Version 10 from Version 9.1 are as follows.

Table 12. Overview of steps for remigration to DB2 Version 10

Step	Description	Job
1	Run DSN1COPY with the CHECK option on the catalog table spaces. Invoke DSN1CHKR to check for broken links on your Version 9.1 system. Run the queries in DSNTESQ to check for consistency between catalog tables.	(none)
2	Take an image copy of your Version 9.1 catalog.	DSNTIJIC
3	Stop your Version 9.1 system.	(none)
4	Reconnect TSO, IMS, and CICS to DB2 Version 10.	(none)
5	Rename cataloged procedures.	DSNTIJFV
6	Start DB2 Version 10.	(none)
7	Optionally, take an image copy of the Version 10 catalog.	DSNTIJIC
8	Optionally, run select Version 9.1 verification jobs.	DSNTEJ*xx*

You must perform some additional tasks manually.

Related tasks:

"Remigrating" on page 647

Migration to enabling-new-function mode from Version 9.1: Summary

After you migrate to Version 10 conversion mode from Version 9.1, you must migrate to enabling-new-function mode from Version 9.1 before you can activate new-function mode.

Migration to Version 10 enabling-new-function mode from Version 9.1 includes the steps in the following table.

Table 13. Overview of steps for migrating to DB2 Version 10 enabling-new-function mode from Version 9.1

Step	Description	Job
1	Run installation CLIST using ENFM option.	(none)
2	Take an image copy of your Version 10 catalog.	DSNTIJIC
3	Convert the DB2 Version 10 catalog and directory. Optionally, run job DSNTIJNH to stop enabling-new-function mode processing before job DSNTIJEN completes.	DSNTIJEN

Related tasks:

"Migrating from conversion mode (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)" on page 653

"Completing enabling-new-function mode (from Version 8 or Version 9.1) processing" on page 654

Migration from enabling-new-function mode from Version 9.1 to new-function mode: Summary

After you migrate to Version 10 enabling-new-function mode from Version 9.1, you can migrate to Version 10 new-function mode.

Table 14. Overview of steps for migrating to DB2 Version 10 new-function mode

Step	Description	Job
1	Enter new-function mode.	DSNTIJNF
2	Optionally, check for consistency between catalog tables.	DSNTESQ
3	Optionally, stop DB2 Version 10.	(none)
4	Create the application defaults load module, *dsnhdecp*, for new-function mode.	DSNTIJNG
5	If you stopped DB2 before running DSNTIJNG, start DB2 Version 10.	(none)
6	Install DB2-supplied routines that require new-function mode.	DSNTIJRT
7	Validate the installation of DB2-supplied routines.	DSNTIJRV
8	Run Version 10 verification jobs.	DSNTEJ*xx*

Related concepts:

"Implications of migrating to Version 10 new-function mode" on page 650

Related tasks:

"Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode" on page 661

Reversion to conversion mode* from Version 9.1: Summary

You can revert to conversion mode* from Version 9.1 if you encounter problems after migrating to certain Version 10 modes.

About this task

If you encounter problems in any of the following Version 10 modes, you can revert to conversion mode* from Version 9.1 until the problems are resolved:

- enabling-new-function mode from Version 9.1
- enabling-new-function mode* from Version 9.1
- new-function mode

Procedure

To revert from any of these modes to conversion mode* from Version 9.1:

Run job DSNTIJCS.

Related tasks:

"Reverting to conversion mode* (from Version 8 or Version 9.1)" on page 665

Reversion to enabling-new-function mode* from Version 9.1: Summary

If you want to temporarily prevent the use of new function after you migrate to Version 10 new-function mode, you can revert to enabling-new-function mode* from Version 9.1.

Procedure

To revert from Version 10 new-function mode to enabling-new-function mode* from Version 9.1:

Run job DSNTIJES.

Related tasks:

"Reverting to enabling-new-function mode* (from Version 8 or Version 9.1)" on page 666

Migration from conversion mode* from Version 9.1 to enabling-new-function mode from Version 9.1: Summary

After you revert to Version 10 conversion mode* from Version 9.1, you can migrate to Version 10 enabling-new-function mode from Version 9.1.

About this task

The following table identifies the migration modes that are mentioned in this topic, along with the short names that are used to refer to those modes.

Table 15. Long and short names for migration modes

Short name for migration mode	Long name for migration mode
CM9*	Conversion mode* from Version 9.1
ENFM9	Enabling-new-function mode from Version 9.1
ENFM9*	Enabling-new-function mode* from Version 9.1
NFM	New-function mode

Procedure

To migrate to Version 10 enabling-new-function mode from Version 9.1 from CM9*:

Run job DSNTIJEN.
Whether DSNTIJEN changes DB2 to ENFM9* or ENFM9 depends on the state of DB2 before you reverted to CM9*. If you reverted to CM9* from NFM or ENFM9*, job DSNTIJEN changes DB2 to ENFM9*. If you reverted to CM9* from ENFM9, job DSNTIJEN changes DB2 to ENFM9.

Related tasks:

"Moving from conversion mode* (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)" on page 667

Migration from conversion mode* from Version 9.1 to new-function mode: Summary

If your subsystem is in Version 10 conversion mode* from Version 9.1, you can migrate to Version 10 new-function mode.

Procedure

To migrate from conversion mode* from Version 9.1 to Version 10 new-function mode:

Run job DSNTIJNF. If enabling-new-function mode from Version 9.1 processing did not complete when DB2 was previously in enabling-new-function mode from Version 9.1, DSNTIJNF issues an error that indicates that you need to run DSNTIJEN.

Related tasks:

"Moving from conversion mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

Migration from enabling-new-function mode* from Version 9.1 to new-function mode: Summary

After you revert to Version 10 enabling-new-function mode* from Version 9.1, you can migrate to Version 10 new-function mode.

Procedure

To migrate to Version 10 new-function mode from enabling-new-function mode* from Version 9.1:

Run job DSNTIJNF.

Related tasks:

"Migrating from enabling-new-function mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1

Before you migrate to DB2 Version 10 conversion mode from Version 9.1, complete the premigration activities to prepare your subsystem.

Procedure

__ 1. Complete the following tasks in any order.
   - __ • Ensure that you meet the hardware and software requirements.
   - __ • Plan for storage requirements.
   - __ • Plan for performance.
   - __ • Convert EXPLAIN tables to Version 9.1 format and Unicode encoding.
   - __ • Make adjustments for the following changes in Version 10:
     - __ – Application and SQL release incompatibilities
     - __ – Utility release incompatibilities
     - __ – Command release incompatibilities
     - __ – Security release incompatibilities
     - __ – Other release incompatibilities
     - __ – Deprecated functions
     - __ – Functions that are no longer supported

__ 2. Use SMP/E to load the DB2 libraries.

__ 3. Complete the following tasks in any order.
   - __ • Tailor the installation jobs by running the installation CLIST in migration mode.
   - __ • Run premigration query DSNTIJPM.
   - __ • Ensure that the data sets for your DB2 catalog and directory reside on SMS-managed storage.
   - __ • Save your critical access paths.
   - __ • Examine new or changed DB2I default panel values to ensure that your customized values are still valid.
   - __ • Ensure that sample objects from your Version 9.1 release are still available.

__ 4. Immediately before starting the migration tasks, ensure that no utility jobs are running.

What to do next

See “Checklist for migration to DB2 Version 10 conversion mode from Version 9.1” for the list of migration tasks to complete after premigration.

Checklist for migration to DB2 Version 10 conversion mode from Version 9.1

After you complete the premigration tasks to prepare your Version 9.1 subsystem, you can continue with migrating to DB2 Version 10 conversion mode from Version 9.1.

About this task

Complete these steps in the order that they are listed.

Procedure

- __ • Complete premigration activities.
- __ • Optional: Run the link checker on DB2 table spaces.
- __ • Optional: Determine which packages are invalid after migration.
- __ • Optional: Check for consistency between catalog tables.

- __ • Take image copies of the directory and catalog: job DSNTIJIC.
- __ • Connect DB2 to TSO.
- __ • Optional: Connect DB2 to IMS.
- __ • Optional: Connect CICS to DB2.
- __ • Stop your DB2 Version 9.1 subsystem.
- __ • Optional: Back up your Version 9.1 volumes.
- __ • Define DB2 initialization parameters: job DSNTIJUZ.
- __ • Optional: Establish subsystem security.
- __ • Define DB2 Version 10 to z/OS, and build cataloged procedures: job DSNTIJMV.
- __ • Define system data sets: job DSNTIJIN.
- __ • Optional: Define authorization exit routines: job DSNTIJEX.

  You can skip this step if you did not run job DSNTIJEX for your Version 9.1 subsystem.
- __ • Optional: IPL z/OS.

  You can skip this step if no PARMLIB updates exist and your early code is at the correct level.
- __ • Start DB2 Version 10.
- __ • Tailor the DB2 Version 10 catalog: job DSNTIJTC.
- __ • Optional: Check the integrity of the catalog and directory.
- __ • Optional: Rebuild indexes that are put in the informational rebuild-pending state during migration to DB2 Version 10: job DSNTIJRI.
- __ • Enable change data capture.
- __ • Prepare the dynamic SQL program and bind the packages for DB2 REXX Language Support. Optionally, create additional table spaces in the work file database: job DSNTIJTM.
- __ • Rebind SPUFI packages and packages that are defined by IBM: job DSNTIJSG.
- __ • Migrate EXPLAIN tables to the current format and encoding type.
- __ • Configure DB2 for running stored procedures and user-defined functions.
- __ • Set up DB2-supplied routines and supported objects: jobs DSNTIJRW, DSNTIJRT and DSNTIJRV.
- __ • Verify views.
- __ • Take an image copy of the Version 10 catalog: job DSNTIJIC.
- __ • Set up the administrative task scheduler.
- __ • Optional: Run Version 9.1 verification jobs: jobs DSNTEJ*xx*.
- __ • Optional: Collect improved statistics.

  You can skip this step if your Version 9 RUNSTATS procedures included the KEYCARD option.

Chapter 2. Preparing your system to install or migrate DB2

The key to a successful installation or migration of DB2 is planning. Prepare your environment and system before you begin installing or migrating.

About this task

Plan the amount of direct-access storage and virtual storage that you need. Planning and coordinating with other DB2 subsystems is essential if you plan to install the distributed data facility (DDF). Review the values that are needed for the parameters on the installation and migration panels. By planning in advance, your task of filling in the parameters becomes easier.

Related tasks:

"Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15

"Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25

"Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14

"Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24

"Running the installation CLIST" on page 219

Controlling connections to remote systems (DB2 Administration Guide)

Hardware and software requirements

Before you can install or migrate DB2, you must fulfill the hardware and software requirements.

You must migrate to a z/OS Version 1 Release 10 or later environment before installing or migrating to DB2 Version 10. DB2 Version 10 for z/OS operates on z990, z890, z9®, z10™, and subsequent 64-bit z/Architecture® processors. See the DB2 Program Directory for information about system requirements.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Required maintenance

In addition to ensuring that you have required hardware and software products installed, you need to make sure that required service changes have been applied.

To help you identify required service, z/OS Version 1 Release 10 (or any subsequent release) introduces enhanced HOLDDATA, which provides several advantages over previous HOLDDATA. One advantage is the addition of fix categories (FIXCATs), which provide a simplified, automated method of identifying

© Copyright IBM Corp. 1982, 2012

and applying missing PTFs that are required for installation or migration or to support specific product functions. You can use this automated process instead of manually reviewing Preventive Service Planning (PSP) buckets, identifying service that is already installed, and applying missing PTFs. For more information about enhanced HOLDDATA and available FIXCATs, see Enhanced HOLDDATA for z/OS and OS/390.

Planning storage for DB2

When you install or migrate DB2, consider the size of your site and how it will be set up to determine your storage needs.

About this task

The parameters that you specify when you run the installation CLIST affect the sizes of some data sets and the required amount of virtual storage. All data sets are linear data sets, with the exception of the bootstrap data set, which is a key-sequenced data set.

You can use Data Facility Storage Management Subsystem (DFSMS) to manage DB2 data sets. It provides automatic backup and recovery features, which might require disk storage beyond what is estimated below. Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

You can estimate your storage needs based on models for a small site, a medium site, a large site, and an extra-large site. The definitions for each model are as follows.

- The **small** site supports a small number of DB2 users. The small site has about 100 plans, 50 application databases, and 1000 tables.
- The **medium** site supports more extensive use of DB2 databases. The medium site has about 200 plans, 200 application databases, and 4000 tables.
- The **large** site supports heavy use of DB2. The large site has about 400 plans, 400 application databases, and 8000 tables.
- The **extra-large** site supports very heavy use of DB2. The extra-large site has about 600 plans, 600 application databases, and 12,000 tables.

When you first install DB2, choose one of these models. Later, you can modify parameters to better suit your needs.

Related concepts:

"Storage estimates for data sharing environments" on page 179

z/OS DFSMS Storage Administration Reference

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Related information:

DB2 for z/OS Program Directories

DB2 subsystem storage requirements

You can estimate the storage requirements for your DB2 subsystem based on your site size.

To determine the estimated storage requirements, see the tables below. The space requirements do not include space for user databases, image copies, archive logs, or temporary data sets that you create during installation or migration.

These estimated space requirements are based on the assumption that, when running the installation CLIST, you accept the default values for the number of databases, tables, and application plans that are expected at your site. You specify these values on installation panel DSNTIPD.

If you do not accept the default values, you can calculate the storage that you need for the DB2 data sets by using the information in Active log data sets storage requirements. For other data sets, you can use the formulas in the CLIST. After calculating the required storage for each data set, you can calculate the total requirements.

Table 16. Estimated space requirements (in cylinders) for DB2 by site size

Site size	3390
Small	949
Medium	1499
Large	5142
Extra-large	8872

When you are estimating storage needs, you must also consider whether you are using extended address volumes. If you use extended address volumes, you need to add 10 cylinders for each object in the cylinder-managed space of an extended address volume. The following table shows which DB2 data sets can reside in the extended addressing space of an extended address volume for each z/OS release.

Table 17. Data sets that can reside in extended address spaces

	Data sets allowed in extended addressing space of z/OS 1.10?	Data sets allowed in extended addressing space of z/OS 1.11?	Data sets allowed in extended addressing space of z/OS 1.12?
Tables and indexes	Yes	Yes	Yes

Table 17. Data sets that can reside in extended address spaces (continued)

	Data sets allowed in extended addressing space of z/OS 1.10?	Data sets allowed in extended addressing space of z/OS 1.11?	Data sets allowed in extended addressing space of z/OS 1.12?
BSDS	Yes	Yes	Yes
Active logs	Yes	Yes	Yes
Archive logs	No	Yes (extended format sequential data sets)	Yes
Utilities sequential input and output data sets	No	Yes (extended format sequential data sets)	Yes
Utilities partitioned data sets and PDSEs	No	No	Yes
Sort work data sets	No	No	Yes
DB2(r) installation data sets (CLISTs, panels, samples, macros, and so on)	No	No	Yes
SDSNLINK, SDSNLOAD	No	No	Yes

The table below provides estimated storage requirements in megabytes (MB) for DB2 data sets. Individual values are rounded and may not add up to the total. Estimated space requirements do not significantly differ by device type. Although the DB2 libraries require a fixed amount of space, disk requirements for active logs and the DB2 catalog increase with the size of a site. You need additional space for archive logs, image copies, user databases, and other working data sets.

Table 18. Estimated space requirements (in megabytes) for DB2 data sets by site size

Site size	DB2 libraries	DB2 catalog	Directory	Active logs	BSDS	Work file database	Total
Small	316	199	61	102	15	24	712
Medium	316	342	198	204	15	24	1094
Large	316	546	385	2028	15	372	3657
Extra-large	316	747	572	4050	15	580	6275

Related reference:

"Sizes panel 1: DSNTIPD" on page 313

Related information:

DB2 for z/OS Program Directories

DB2 catalog storage requirements

Estimated storage requirements for the DB2 catalog include space for the entire set of DB2 catalog data sets and their indexes.

Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory

data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

Estimated storage requirements are shown in the table below.

Table 19. Estimated space requirements (in cylinders) for the DB2 catalog by site size

Site size	3390
Small	224
Medium	436
Large	727
Extra-large	1017

If you plan to use partitioned table spaces that are created with (or will grow to have a large number of) partitions, you might need to allocate more storage than what is suggested in the table above. Some catalog objects can grow substantially over time if a large number of partitions are created or added to table spaces.

Related tasks:

"Define the SMS environment for the DB2 catalog and directory data sets (DSNTIJSS)" on page 576

Changing catalog and directory size and location (DB2 Performance)

DB2 directory storage requirements

The amount of directory space required for the DB2 directory depends mainly on the number of user databases, application plans and packages, and tables in the DB2 subsystem.

Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

Storage requirements for the DB2 directory are shown in the table below.

Table 20. Estimated space requirements (in cylinders) for the DB2 directory by site size

Site size	3390
Small	84
Medium	278
Large	543
Extra-large	808

If you plan to use partitioned table spaces that are created with (or will grow to have) a large number of partitions, you might need to allocate more storage than what is suggested in the table above. Some directory objects can grow substantially over time if a large number of partitions are created or added to table spaces.

Related tasks:

"Define the SMS environment for the DB2 catalog and directory data sets (DSNTIJSS)" on page 576

Active log data sets storage requirements

Active log data sets record significant events and data changes. Active log data sets are periodically offloaded to archive log data sets. Storage requirements for active log data sets depend on the frequency of data updates and of offloads to archive log data sets.

If you change data frequently and offload it to the archive log infrequently, you need a large amount of disk space for the active log. If, under normal circumstances, offloading occurs once each day, the active log data sets can hold the log records that your subsystem produces during one day of processing.

These are the assumptions concerning each of the four site models:

- The **small site** changes data 1800 times per hour, and the active log is offloaded once each day.
- The **medium site** changes data 3600 times per hour, and the active log is offloaded once each day.
- The **large site** changes data 36,000 times per hour, and the active log is offloaded once each day.
- The **extra-large site** changes data 72,000 times per hour, and the active log is offloaded once each day.

Example: This is how the DSNTINST CLIST calculates the amount of disk space that a medium site needs:

1. During the ISPF tailoring session, assume that you specified:
   - An archive period estimate of 24 hours (ARCHIVE LOG FREQ parameter on installation panel DSNTIPL).
   - A data change rate estimate of 3600 changes per hour (UPDATE RATE parameter on installation panel DSNTIPL).
2. DB2 uses 400 bytes as the size of a typical row.
3. Other types of log records are comparatively small in size and are fixed in length. The length of each type depends on the information that it contains.
4. The size of the active log, including disk track overhead, is estimated as:
   Data set size
   = (data change log record size)
   * (data change rate per hour)
   * (hours in archive period)
   = 400 bytes * 3600 per hour * 24 hours
   + data set allocation overhead
   = 34 MB
   If you have dual logs, you will need 68 MB. Dual logs are recommended but not required.
   If you have dual logs with three data sets each, you will need 204 MB.

   Data set allocation overhead is the difference between the allocated space and the requested data size in 4 KB blocks. The change is caused by the difference between the space in 4 KB blocks and the track size, which includes rounding up to a cylinder boundary. In this example, space is requested on a 3390, and 48 cylinders are allocated per data set.

If a LOB table space is defined with LOG(YES), estimate 1.0 to 1.1 times the size of the LOB for inserts or updates. Only control information is logged for deletes and table spaces that are defined with LOG(NO). The formula for calculating LOB table space requirements is:

```
MAX (50 bytes, 1.0 * (size of LOB))
```

If you enabled data sharing, you generally need to have more disk space for the active log, and you need to archive the logs more frequently.

If you accept the defaults of using three active log data sets and dual logging, DB2 creates six active log data sets. For a typical production DB2 subsystem, you should have more than three active logs and you should use dual logs. Other choices can lead to degraded performance when you encounter an I/O error. You can avoid some outages by having an adequate number of active logs for several hours of batch update processing.

You get better performance with larger active logs for other common problems, such as long-running updates. However, using large active log data sets causes DB2 to archive less frequently. Infrequent archiving can result in increased data loss in the event of remote-site recovery or disaster recovery.

The table immediately below shows estimated storage requirements for active log data sets (assuming dual logging).

Table 21. Estimated space requirements (in MB) for active log data sets by site size

Site size	Archive period (hours)	Data change rate (per hour)	Space for each active log data set (MB)	Total space for six active log data sets (MB)
Small	24	1800	17	102
Medium	24	3600	34	204
Large	24	36,000	338	2028
Extra-large	24	72,000	675	4050

The following table shows the amount of space that is required for active log data sets on various devices. The estimates in both of these tables include track overhead.

Table 22. Estimated space requirements (in cylinders) for the active log by site size

Site size	3390
Small	144
Medium	288
Large	2880
Extra-large	5760

Some other considerations for the size of your active log data sets include:

- Tape utilization

  Recommendation: You should archive to disk, rather than tape, especially in a data sharing environment. Using tape archive logs can result in contention for tape volumes and delays in reading archive logs.

When you archive to a media type that is listed in the table below, the planning size numbers are suggested sizes for your active logs. If the size of an active log data set is small compared to the size of a tape, the tape utilization is fairly low.

The maximum size of a DB2 active log data set is 4 GB.After conversion of the BSDS, the maximum size of a DB2 active log data set is 4 GB. You can have 93 active log data sets, and up to 10,000 archive log data sets per log copy.

Table 23. Estimated active log planning size

Log media	Estimated planning size
6250 BPI tape	100 MB
3590 High-Performance Tape Subsystem	10 GB
3480 cartridge	200 MB or more
4mm cartridge (60 m)	1.3 GB
4mm cartridge (90 m)	2.0 GB
4mm cartridge (120 m)	4.0 GB

Using larger block sizes for archive logs can increase the estimated planning sizes by up to 40%. You specify block size for archive logs with the BLOCK SIZE field on installation panel DSNTIPA. Compression on the newer cartridge units can also substantially increase the estimated planning sizes. If you use compression on the tape units, you can have larger active logs and controls for long-running updates, and you can archive to disk with DFSMShsm migration to tape.

- Checkpoint frequency

  The CHECKPOINT TYPE field on panel DSNTIPL1 specifies whether the frequency of system checkpoints is controlled by number of consecutive log records, number of minutes, or both. The RECORDS/CHECKPOINT field and MINUTES/CHECKPOINT field on panel DSNTIPL1 determine the number of log records between checkpoints or the number of minutes per checkpoint. When you choose a value for the checkpoint frequency, you must consider the trade-offs between the overhead that is needed for frequent subsystem checkpoints and the time that is needed to restart a DB2 subsystem after a termination without a quiesce. If the checkpoint value is more than 1,000,000 log records, the time that is needed to restart DB2 after a termination without a quiesce can grow to over 15 minutes. The recommended values for the checkpoint frequency are in the range of 500,000 to 1,000,000 log records or 2 to 5 minutes.

- Number of tables that are defined for data capture

  When tables are defined with the DATA CAPTURE CHANGES option, the entire before-image of an updated row is captured on the log. This additional information can represent an increase in log data compared to tables that are not defined with the DATA CAPTURE CHANGES option, depending on whether the table contains fixed-length or variable-length rows.

Related concepts:

Logging environment for data sharing (DB2 Data Sharing Planning and Administration)

Related reference:

"Checkpoint parameters: DSNTIPL1" on page 428

Bootstrap data sets storage requirements

Each bootstrap data set (BSDS) requires 6 MB. During installation, DB2 automatically allocates two copies of the BSDS. During migration, Version 10 adopts the BSDS characteristics that you specified for your previous version.

That is, if you specified two copies of the BSDSs for your previous version, you will have two copies for Version 10. Keeping two copies of the BSDS is strongly recommended. The total space requirement is about 15 MB for both BSDSs. The BSDSs at any size site require about 10 cylinders of 3390 storage.

Related concepts:

Management of the bootstrap data set (DB2 Administration Guide)

Work file database storage requirements

The work file database is used as storage for processing SQL statements that require working storage.

From V8 As of DB2 Version 9.1, the TEMP database is no longer used. If you are migrating a non-coexisting Version 8 system, you can reclaim storage that is associated with a Version 8 TEMP database by dropping the TEMP database and redefining the storage to be used for the work file database. If storage space is not a problem, wait to drop the TEMP database until you are sure that you do not need to fall back to Version 8. If you fall back to Version 8 and you dropped the TEMP database during migration, you have to re-create it after fallback. From V8

The table below shows the disk requirement estimates for the temporary work files in the work file database. Other work file database storage requirements relate to migration.

If you are installing DB2, see *Storage requirements for installation*, below.

Table 24. Estimated space requirements (in cylinders) for the work file database by site size

Site size	3390
Small	29
Medium	29
Large	456
Extra-large	712

Storage requirements for migration

The migration job DSNTIJTC creates and updates indexes on catalog tables. These indexes are created and updated sequentially during migration. The work file database is used for the sort of each index; DB2 needs enough work file storage to sort the largest of the indexes that are listed in the table below. The migration fails if you do not have enough storage. Therefore, ensure that you have enough space before you begin.

From V8 PSPI For migration from DB2 Version 8, the following indexes are new and changed for existing catalog tables.

Table 25. Indexes that are added or updated sequentially using the work file database (for migration from Version 8)

Catalog table name	Index name	Column names
SYSIBM.SYSCOLAUTH	SYSIBM.DSNACX02	CREATOR, TNAME, TIMESTAMP
	SYSIBM.DSNACX03	GRANTOR, GRANTORTYPE, CREATOR, TNAME, TIMESTAMP
	SYSIBM.DSNACX04	GRANTEE, GRANTEETYPE, CREATOR, TNAME, TIMESTAMP
SYSIBM.SYSCOLUMNS	SYSIBM.DSNDCX05	TBCREATOR, TBNAME
SYSIBM.SYSDBAUTH	SYSIBM.DSNADH01	GRANTEE, NAME, GRANTEETYPE
	SYSIBM.DSNADH02	NAME
	SYSIBM.DSNADX01	GRANTOR, NAME, GRANTORTYPE
SYSIBM.SYSDBRM	SYSIBM.DSNDBX01	PLNAME
	SYSIBM.DSNDBX02	PLNAME, NAME
SYSIBM.SYSFIELDS	SYSIBM.DSNDFX01	TBCREATOR, TBNAME, NAME
SYSIBM.SYSINDEXES	SYSIBM.DSNDXX07	TBCREATOR, TBNAME
SYSIBM.SYSINDEXPART	SYSIBM.DSNDRX03	IXCREATOR, IXNAME
SYSIBM.SYSKEYS	SYSIBM.DSNDKX02	IXCREATOR, IXNAME
	SYSIBM.DSNDKX03	IXCREATOR, IXNAME, COLSEQ
SYSIBM.SYSPACKAUTH	SYSIBM.DSNKAX01	GRANTOR, LOCATION, COLLID, NAME, GRANTORTYPE
	SYSIBM.DSNKAX02	GRANTEE, LOCATION, COLLID, NAME, BINDAUTH, COPYAUTH, EXECUTEAUTH, GRANTEETYPE
SYSIBM.SYSPARMS	SYSIBM.DSNOPX01	SCHEMA, SPECIFICNAME, ROUTINETYPE, ROWTYPE, ORDINAL, VERSION
SYSIBM.SYSPLANAUTH	SYSIBM.DSNAPH01	GRANTEE, NAME, EXECUTEAUTH, GRANTEETYPE
	SYSIBM.DSNAPX01	GRANTOR, GRANTORTYPE
	SYSIBM.DSNAPX02	NAME
SYSIBM.SYSPLANDEP	SYSIBM.DSNGGX05.	DNAME
SYSIBM.SYSRELS	SYSIBM.DSNDLX04	CREATOR, TBNAME, RELNAME
SYSIBM.SYSRESAUTH	SYSIBM.DSNAGH01	GRANTEE, QUALIFIER, NAME, OBTYPE, GRANTEETYPE
	SYSIBM.DSNAGX01	GRANTOR, QUALIFIER, NAME, OBTYPE, GRANTORTYPE
SYSIBM.SYSROUTINEAUTH	SYSIBM.DSNOAX01	GRANTOR, SCHEMA, SPECIFICNAME, ROUTINETYPE, GRANTEETYPE, EXECUTEAUTH, GRANTORTYPE

Table 25. Indexes that are added or updated sequentially using the work file database (for migration from Version 8) (continued)

Catalog table name	Index name	Column names
SYSIBM.SYSROUTINES	SYSIBM.DSNOFX01	NAME, PARM_COUNT, ROUTINETYPE, PARM_SIGNATURE, SCHEMA, PARM1, PARM2, PARM3, PARM4, PARM5, PARM6, PARM7, PARM8, PARM9, PARM10, PARM11, PARM12, PARM13, PARM14, PARM15, PARM16, PARM17, PARM18, PARM19, PARM20, PARM21, PARM22, PARM23, PARM24, PARM25, PARM26, PARM27, PARM28, PARM29, PARM30, VERSION
	SYSIBM.DSNOFX02	SCHEMA, SPECIFICNAME, ROUTINETYPE, VERSION
	SYSIBM.DSNOFX07	NAME, PARM_COUNT, ROUTINETYPE, SCHEMA, PARM_SIGNATURE, PARM1, PARM2, PARM3, PARM4, PARM5, PARM6, PARM7, PARM8, PARM9, PARM10, PARM11, PARM12, PARM13, PARM14, PARM15, PARM16, PARM17, PARM18, PARM19, PARM20, PARM21, PARM22, PARM23, PARM24, PARM25, PARM26, PARM27, PARM28, PARM29, PARM30, VERSION
SYSIBM.SYSSCHEMAAUTH	SYSIBM.DSNSKX01	GRANTEE, SCHEMANAME, GRANTEETYPE
	SYSIBM.DSNSKX02	GRANTOR, GRANTORTYPE
SYSIBM.SYSSEQUENCEAUTH	SYSIBM.DSNWCX02	GRANTOR, SCHEMA, NAME, GRANTORTYPE
	SYSIBM.DSNWCX03	GRANTEE, SCHEMA, NAME, GRANTEETYPE
SYSIBM.SYSSTMT	SYSIBM.DSNPSX01	PLNAME, NAME
	SYSIBM.DSNPSX02	PLNAME, NAME, STMTNOI, STMTNO, SECTNOI, SECTNO, SEQNO
SYSIBM.SYSSYNONYMS	SYSIBM.DSNDYX02	TBCREATOR, TBNAME
SYSIBM.SYSTABAUTH	SYSIBM.DSNATX01	GRANTOR, GRANTORTYPE
	SYSIBM.DSNATX05	TCREATOR, TTNAME, TIMESTAMP
SYSIBM.SYSTABLEPART	SYSIBM.DSNDPX05	DBNAME, TSNAME
SYSIBM.SYSTABLES	SYSIBM.DSNDTX05	DBNAME, TSNAME
SYSIBM.SYSUSERAUTH	SYSIBM.DSNAUH01	GRANTEE, GRANTEDTS, GRANTEETYPE
	SYSIBM.DSNAUX02	GRANTOR, GRANTORTYPE
SYSIBM.SYSVIEWDEP	SYSIBM.DSNGGX06	DCREATOR, DNAME, DTYPE
SYSIBM.SYSVOLUMES	SYSIBM.DSNSSH02	SGNAME

PSPI From V8

From V9.1 PSPI For migration from DB2 Version 9.1, the following indexes are new and changed for existing catalog tables.

Table 26. Indexes that are added or updated sequentially using the work file database (for migration from Version 9.1)

Catalog table name	Index name	Column names
SYSIBM.SYSCOLAUTH	SYSIBM.DSNACX02	CREATOR, TNAME, TIMESTAMP
	SYSIBM.DSNACX03	GRANTOR, GRANTORTYPE, CREATOR, TNAME, TIMESTAMP

Table 26. Indexes that are added or updated sequentially using the work file database (for migration from Version 9.1) (continued)

Catalog table name	Index name	Column names
	SYSIBM.DSNACX04	GRANTEE, GRANTEETYPE, CREATOR, TNAME, TIMESTAMP
SYSIBM.SYSCOLUMNS	SYSIBM.DSNDCX05	TBCREATOR, TBNAME
SYSIBM.SYSDBAUTH	SYSIBM.DSNADH02	NAME
SYSIBM.SYSDBRM	SYSIBM.DSNDBX01	PLNAME
	SYSIBM.DSNDBX02	PLNAME, NAME
SYSIBM.SYSFIELDS	SYSIBM.DSNDFX01	TBCREATOR, TBNAME, NAME
SYSIBM.SYSINDEXES	SYSIBM.DSNDXX07	TBCREATOR, TBNAME
SYSIBM.SYSINDEXPART	SYSIBM.DSNDRX03	IXCREATOR, IXNAME
SYSIBM.SYSKEYS	SYSIBM.DSNDKX02	IXCREATOR, IXNAME
	SYSIBM.DSNDKX03	IXCREATOR, IXNAME, COLSEQ
SYSIBM.SYSPLANAUTH	SYSIBM.DSNAPX02	NAME
SYSIBM.SYSPLANDEP	SYSIBM.DSNGGX05.	DNAME
SYSIBM.SYSRELS	SYSIBM.DSNDLX04	CREATOR, TBNAME, RELNAME
SYSIBM.SYSSTMT	SYSIBM.DSNPSX01	PLNAME, NAME
	SYSIBM.DSNPSX02	PLNAME, NAME, STMTNOI, STMTNO, SECTNOI, SECTNO, SEQNO
SYSIBM.SYSSYNONYMS	SYSIBM.DSNDYX02	TBCREATOR, TBNAME
SYSIBM.SYSTABAUTH	SYSIBM.DSNATX05	TCREATOR, TTNAME, TIMESTAMP
SYSIBM.SYSTABLEPART	SYSIBM.DSNDPX05	DBNAME, TSNAME
SYSIBM.SYSTABLES	SYSIBM.DSNDTX05	DBNAME, TSNAME
SYSIBM.SYSVIEWDEP	SYSIBM.DSNGGX06	DCREATOR, DNAME, DTYPE
SYSIBM.SYSVOLUMES	SYSIBM.DSNSSH02	SGNAME

PSPI From V9.1

You might need more storage for the work file database if you have a large amount of data to sort and a large amount of concurrent sort activity. If you are sorting compressed data, allow for the same amount of storage that you would need if the data were not compressed. The maximum amount of storage that you need is enough to satisfy the requirements of the largest combination of concurrent activities that use the work file database. The amount of storage that is required for a sort depends on the following variables:

- Data size
- Sort key size

You can estimate the total amount of work file space that is needed to perform the sort as follows:

- Let MIN be the operation of selecting the lowest value from a set of values.
- Let FLOOR be the operation of discarding the decimal portion of a real number.
- Let CEILING be the operation of rounding a real number up to the next-highest integer.
- Let *Data* be the total data length in bytes.
- Let *Key* be the total length of the sort key.

- Let *Prefix* be the 6-byte header.
- Let *Rows* be the total number of rows that are being sorted.

Then calculate as follows:

```
Records per page = MIN(MAXROWS, FLOOR (4076 / (Data + Key + Prefix)))

Total pages = CEILING (Rows / Records per page)

Total segments = CEILING (Total pages / 24)
```

The number of records per page cannot exceed 255 (the value of MAXROWS).

This result tells you how much storage is needed in the work file database after sort processing. However, if a merge phase was required during sort processing, an additional intermediate copy of the records might exist at any given time. For most subsystems, you can assume that about half of the records that are involved in a sort have two copies. Therefore, a multiplier value of 1.5 is safe. If you want to be conservative, choose 2 for your multiplier value. Therefore, the amount of storage that is used in the work file database during sort processing can vary from 1 to 2 times the storage that is needed after sort processing. The actual storage that is used might also increase if you have little available buffer pool storage.

When a large object (LOB) column is part of a result table, and the result table must be placed in a work file for sorting, the actual LOB column data is not placed in the work file. Therefore, LOB columns do not require large increases in the amount of work file space that DB2 requires. For work file calculations, you can assume 51 bytes of storage per LOB column for the work file.

To determine the number of tracks that are needed, convert the number of pages into bytes, and divide the result by the number of bytes per unit. Let *r* be the number of 4096-byte records per track, and let *safety_factor* be a number from 1.5 to 2.0. For 3390 devices, *r* is 12.

```
Tracks = CEILING (Total pages / r) * safety_factor
```

Example 1: Consider a table (TABLE1) that contains 45 327 rows, for which you want to create a **nonunique index** on COL1 CHAR(3) NOT NULL, COL2 CHAR(4), COL3 VARCHAR(20), and COL4 SMALLINT. Determine the amount of temporary storage that DB2 needs to create this index as follows:

- *Data* = 3 + (4 + 1) + (20 + 1) + (2 + 1) + 4 = 36
- *Key* = 36 (*Data* plus RID is key for CREATE INDEX)
- *Rows* = 45 327
- *Records per page* = MIN(MAXROWS, FLOOR (4076 / (36 + 36 + 6))) = 52
- *Total pages* = CEILING (45 327 / 52) = 872
- *Segments* = CEILING (872 / 24) = 37
- *Tracks* = CEILING (872 / 12) * 1.5 = 111

Example 1 is a data page calculation for storing index keys in the work file database. For this example, 111 tracks of a 3390 storage device are needed. The 2-byte length field of a VARCHAR column is not a part of *Data* for CREATE INDEX. The RID field is a part of *Data*, and the *Key* includes the entire *Data* portion, including the RID.

Example 2: Consider TABLE1 again and the following SQL query:

GUPI

```
SELECT COL1,COL2,COL3,COL4
   FROM TABLE1
   ORDER BY COL2,COL3,COL1;
```

GUPI

This query, which includes an ORDER BY clause, requires a sort. Determine the amount of temporary storage that is required for this table as follows:

- *Data* = 3 + (4 + 1) + (20 + 2 + 1) + (2 + 1) = 34
- *Key* = (4 + 1) + (20 + 1) + 3 = 29
- *Rows* = 45 327
- *Records per page* = MIN(MAXROWS, FLOOR (4076 / (34 + 29 + 6))) = 59
- *Total pages* (final result) = CEILING (45 327 / 59) = 769
- *Segments* (final result) = CEILING (769 / 24) = 33
- *Total pages* (during processing) = CEILING (1.5 * 769) = 1154
- *Segments* (during processing) = CEILING (1.5 * 35) = 53
- *Tracks* = CEILING (1238 / 12) = 104

For this example, which is a table calculation, 104 tracks of a 3390 storage device are needed. The 2-byte length field of a VARCHAR column is a part of *Data* for CREATE INDEX. The RID field is not a part of *Data*, and the *Key* does not include the entire *Data* portion.

You can use the sort summary trace record, IFCID 0096, to simplify some of the calculations. This record shows the number of records that are sorted, the sort record size (*Data* + *Key*), and an indication of whether a merge phase was required for an individual sort request.

Also note that if the overall record length (*Data* + *Key* + *Prefix*) exceeds 100 bytes, DB2 attempts to create the work file in a table space with a 32 KB page size. If the overall record length is 100 bytes or less, DB2 uses a table space with a 4 KB page size. Therefore, you must have at least one 4 KB work file table space and one 32 KB work file table space.

Storage requirements for installation

If you are installing DB2, use the following calculations to determine either PRIQTY (primary space allocation) or DSSIZE and MAXPARTITION values for creating work file table spaces. During installation, you must create at least one 4 KB work file table space and one 32 KB work file table space.

For 4 KB work file table spaces:

- Let *total space* be the total amount of space in megabytes (from the TEMP 4K SPACE field on panel DSNTIP9.
- Let *table spaces* be the number of table spaces (from the TEMP 4K TBL SPACES field on panel DSNTIP9.
- Let CEILING be the operation of rounding a real number up to the next-highest integer.
- Let *GB per table space* be the amount of space per work file table space in gigabytes.

Calculate the following:

```
GB per table space = CEILING (total space / table spaces / 1024)

If GB per table space < 1 GB, then PRIQTY = CEILING (total space / table spaces
If GB per table space ≤ 16 384 GB, then DSSIZE = 4 GB
 and MAXPARTITIONS = GB per table space / 4
Otherwise, DSSIZE = 4 GB and MAXPARTITIONS = 4096
```

For 32 KB work file table spaces:

- Let *total space* be the total amount of space in megabytes (from the TEMP 32K SPACE field on panel DSNTIP9.
- Let*table spaces* be the number of table spaces (from the TEMP 32K TBL SPACES field on panel DSNTIP9.
- Let CEILING be the operation of rounding a real number up to the next-highest integer.
- Let *GB per table space* be the amount of space per work file table space in gigabytes.

Calculate the following:

```
GB per table space = CEILING (total space / table spaces / 1024)

If GB per table space < 1 GB, then PRIQTY = CEILING (total space / table spaces)
If GB per table space ≤ 16 384 GB, then DSSIZE = 4 GB
 and MAXPARTITIONS = GB per table space / 4
If GB per table space ≤ 131 072 GB, then DSSIZE = 32 GB
 and MAXPARTITIONS = GB per table space / 32
Otherwise, DSSIZE = 32 GB and MAXPARTITIONS = 4096
```

Related concepts:

Diagnostic traces for attachment facilities (DB2 Administration Guide)

Related tasks:

From V8 Changing your work database for a migrated installation of DB2 (DB2 Administration Guide) From V8

Temporary table space storage requirements:

DB2 uses declared temporary tables for processing some types of scrollable cursors.

GUPI The following scrollable cursors require declared temporary tables:

- SENSITIVE STATIC SCROLL.
- INSENSITIVE SCROLL.
- ASENSITIVE SCROLL, if the effective cursor sensitivity is INSENSITIVE. A cursor that meets the criteria for a read-only cursor has an effective sensitivity of INSENSITIVE.

Before application programmers can use cursors that require declared temporary tables, you need to ensure that the work file database exists. You also need to create temporary table spaces for the declared temporary tables that are large enough to process your cursors. For example:

```
CREATE TABLESPACE DTTTS IN WORKFILE
  SEGSIZE 4;
```

If more than one table space in the work file database is in the subsystem, DB2 chooses the table spaces to use for static scrollable cursors. GUPI

Important: If you use declared temporary tables, you must define at least one of the table spaces in the work file database to have a page size of 32 KB or greater.

Calculating the size of the longest row in the declared temporary table:

The page size of the table space in the work file database must be large enough for the longest row in the declared temporary table. A row in the declared temporary table might be considerably larger than the row in the table for which the static scrollable cursor is used.

About this task

The size of the row depends on these factors:
- The number of columns that are stored in the declared temporary table
- The size of each column

The number of columns in the declared temporary table depends on these factors:
- The number of columns in the select list of the SELECT statement for the cursor
- The number of expressions in the select list that contain more than a single column name
- If the SELECT statement contains an ORDER BY clause, the number of columns in the ORDER BY clause
- An indication of whether the result table is read-only

Procedure

To calculate the size of the longest row in the declared temporary table:
1. Identify the columns in the declared temporary table.

   For a read-only result table, the following items are columns in the declared temporary table:
   - Each expression in the select list
   - Each column in the ORDER BY clause that is not in the select list
   - One additional column that DB2 generates

   For a result table that is not read-only, the following items are columns in the declared temporary table:
   - Each column in the select list
   - Each expression in the select list that contains more than a single column name
   - Three additional columns that DB2 generates
2. Determine the length of each column in the declared temporary table.
   a. For columns other than the columns that DB2 generates, determine the data type of each column. See Retrieving catalog information about columns (DB2 SQL) for more information about determining column data types.

      Determine the length of each column, based on the data type, in the following way:
      - For a declared temporary table column that is the result of the concatenation of CHAR, VARCHAR, GRAPHIC, or VARGRAPHIC data types, use the information in Retrieving catalog information about columns (DB2 SQL) to determine the column length. Add 1 byte if the column is nullable.

- For a declared temporary table column that is the result of an expression that contains LOB columns, specify a length of 120 for each column, literal, or host variable that is referenced in the expression. Then add these lengths for the expression.
- For a LOB data type, specify a length of 120.
- For a ROWID data type, specify a length of 42.
- For a declared temporary table column of any other data type, use the information in Retrieving catalog information about columns (DB2 SQL) to determine the column length. Add 1 byte if the column is nullable.

b. For the columns that DB2 generates, determine the total length for those columns in the following way:
   - If the result table of cursor is read-only, the length for the added column is 22 bytes.
   - If the result table of cursor is not read-only, the total length for the three added columns is 33 bytes.

3. Find the sum of the column lengths that you calculated in the previous step.

Example

Suppose that table T1 has the following columns and data types.

C1 CLOB(100M)

C2 CHAR(10)

C3 VARCHAR(100) NOT NULL

C4 INTEGER NOT NULL

C5 INTEGER

Now suppose that you declare two scrollable cursors for the table:

GUPI

```
DECLARE CUR1 INSENSITIVE SCROLL CURSOR FOR
  SELECT C1, C2 || C3, C4 FROM T1
  WHERE C3 > :HV;

DECLARE CUR2 SENSITIVE STATIC SCROLL
CURSOR FOR
  SELECT C1, C2||C3, C4 FROM T1
  WHERE C3 > :HV;
```

GUPI

The result table for cursor CUR1 is read-only. Therefore, the columns, column lengths, and maximum row length of the declared temporary table for CUR1 are as follows:

Column	Data type	Effective length
C1	CLOB(100M)	120
C2 || C3	VARCHAR(110)	113
C4	INTEGER NOT NULL	4
C5	INTEGER	5
One column added by DB2	(N/A)	22

Column	Data type	Effective length
Total length	(N/A)	265

The result table for cursor CUR2 is not read-only, it is read-write. Therefore, the columns, column lengths, and maximum row length of the declared temporary table for CUR2 are as follows:

Column	Data type	Effective length
C1	CLOB(100M)	120
C2	CHAR(10)	11
C3	VARCHAR(100) NOT NULL	102
C4	INTEGER NOT NULL	4
C2 || C3	VARCHAR(110)	113
Three columns added by DB2	(N/A)	33
Total length	(N/A)	383

Default database storage requirements

The size of the default database depends on column lengths, page sizes, and index column lengths. The estimated size of your data, multiplied by 2, usually provides an adequate planning estimate for the default database size.

Dump data set size storage requirements

When planning for your DB2 subsystem storage, you must reserve enough space for the dump data sets. The storage requirements depend on factors such as the number of dump data sets you have defined and whether you invoke authorized Language Environment® functions.

Recommendation: Use the following guidelines to estimate storage requirements for the dump data sets:

- Have at least two dump data sets.
- Have approximately 1000 cylinders for all dump data sets.

  If you invoke authorized Language Environment functions, a different style of dump might be generated, which requires approximately 520 cylinders.
- Have 3.25 MB of space for DB2 volatile summary storage data.

Summary data in dumps is usually enough to diagnose most problems. In addition to summary data, DB2 also requests a SDUMP from the operating system to provide these additional storage areas if enough space is available in the dump data set:

- DB2 system services address space
- DB2 database services address space
- DB2 distributed data facility (DDF) address space
- Allied address space of the failing allied task
- IRLM address space for data sharing environments

DB2 passes the following parameters to the SDUMP service aid through the SDATA keyword: SQA, ALLPSA, LSQA, SUMDUMP, and CSA (subpools 231 and 241).

After DB2 SVC dump processing is complete, z/OS issues message IEA911E to indicate whether enough space was available in the dump data set to contain the

requested storage areas. If this message indicates that a partial dump was taken, but the 3.25 MB of summary storage is available in the dump, this dump is probably enough for problem diagnosis. Otherwise, IBM Software Support might request that you re-create the problem if storage areas that are required for problem determination are not included in the dump.

System databases storage requirements

If you are installing or migrating, DB2 automatically creates the resource limit facility database and the DB2 Connect™ database. The storage requirements for these databases depend on the amount of user data.

Archive log data sets storage requirements

If you decide to store the archive log data sets on disk, you need to reserve enough space on the device. The active log data set and the BSDS are both written to the same location. Therefore, you must reserve enough storage for the active log and the BSDS.

Recommendation: You should store archive logs on disk, especially if DB2 will be running in data sharing mode.

The total amount of storage that is required for the logs and BSDSs is calculated by the CLIST and displayed in the messages on panel DSNTIPC1.

The installation CLIST uses the amount of space that is computed for the active log data sets for archive primary and secondary space. The CLIST computes this size by starting with the size of the active log data sets in bytes and dividing this number by the block size, which is specified on installation panel DSNTIPA. Primary space for the archive log is the same as for the active log. Secondary space is small and is used if it is needed for cylinder rounding differences on different devices.

Related concepts:

"Active log data sets storage requirements" on page 32

"Bootstrap data sets storage requirements" on page 35

Related reference:

"CLIST calculations panel 2: DSNTIPC1" on page 495

Profile tables storage requirements

If you use the monitoring functions in the DB2 optimization tools, you need to reserve disk space for the profile tables.

Depending on the amount of monitoring that is to be done, the output tables can consume large amounts of disk storage, and some significant virtual storage. This monitoring is optional, and you decide how much monitoring you want to do.

For light to medium monitoring, the amount of required disk space is approximately 100 to 200 MB. To prevent overflows, you need to manage this space by backing up and deleting old data to make room for new monitor data, or by adjusting the amount of disk space that is allocated to the profile tables. You can manage the disk space with optimization client tools or by directly manipulating the profile tables.

Related reference:

Profile tables (DB2 Performance)

Installation CLIST storage calculation

If you choose not to use the estimates for the model sites, you can use the detailed information for disk storage estimates in the installation CLIST.

Recommendation: Use the model site estimates the first time that you install DB2. Use the model approach that is described in "DB2 subsystem storage requirements" on page 29 to estimate DB2 disk use. After your site has some experience in operating DB2, you can recalculate your disk estimates.

The CLIST contains the algorithms that DB2 uses to calculate storage based on the parameters that you supply during installation or migration. You can use these algorithms to calculate the storage needs of your site on a data-set-by-data-set basis.

To see the algorithms that DB2 uses, print or edit the installation CLISTs and REXX EXEC. The DSNTCALC EXEC contains most of the data set calculations. You can run the CLIST to calculate the sizes. CLIST calculations panel 1: DSNTIPC displays the storage sizes that the CLIST calculates.

Related reference:

"CLIST calculations panel 1: DSNTIPC" on page 490

Virtual storage requirements for address spaces

DB2 uses several types of private address spaces, and each type requires storage.

- DB2 distributed data facility (DDF) address space (DSN1DIST)
- IRLM address space (IRLMPROC)
- DB2 system services address space (DSN1MSTR)
- DB2 database services address space (DSN1DBM1)
- DB2 allied agent address spaces
- DB2 stored procedures address spaces (WLM-named)
- DB2 administrative task scheduler address space

DB2 also uses extended common service area (ECSA), 64-bit common storage, and the z/OS Shared Memory Facility.

You might notice that the sample jobs sometimes use a region size of 0 KB. This region size is meant to simplify the installation process in those particular cases. The following topics provide some recommendations about DB2 region sizes. These recommendations are based on average use under normal circumstances on typical systems. Your requirements might be quite different.

DB2 distributed data facility address space (DSN1DIST) storage requirements

The distributed data facility address space supports network communications with other remote systems and execution of database access requests on behalf of remote users.

Recommendation: Use the default region size of 0 KB. This address space is started as part of DDF initialization. The start-up procedure is DSN1DIST.

IRLM address space (IRLMPROC) storage requirements

DB2 uses the IRLM to manage locks. When row locking is used, the number of locks that DB2 acquires might increase, which might in turn increase the amount of storage that IRLM requires. The number of locks that are acquired is dependent on your application.

You can estimate the IRLM control block structure at 540 bytes per lock. IRLM no longer supports placing locks in ECSA. All IRLM locks are now placed in the IRLM private address space.

The PC and MAXCSA parameters are no longer used, but you must maintain them for compatibility reasons. You must specify the parameters and values, but their values are not used. The MAXCSA value must be in the range 0-9999. The amount of available storage for IRLM private control blocks, including locks, is determined by the operating system and site-specific IPL parameters. IRLM reserves approximately 10% of the available private storage to be used for must-complete lock requests.

Use the MODIFY irlmproc,STATUS,STOR command to view and monitor the amount of private storage that IRLM has available.

You can dynamically adjust the amount of below-the-bar private storage by using the MODIFY irlmproc SET,PVT command. This command changes only the monitoring threshold of private storage for IRLM.

You can dynamically adjust the limit for above-the-bar private storage by using the MODIFY irlmproc SET,MLT command. This command updates only the MEMLIMIT that z/OS uses to control the amount of above-the-bar storage that can be requested by an address space. A new value remains in effect until the next time IRLM is stopped and restarted or until the MODIFY command is issued successfully again.

Neither of these MODIFY commands change the physical amount of storage that the operating system assigns to the address space.

Enabling data sharing further increases the storage that IRLM requires. Sysplex query parallelism requires additional storage beyond what is required for data sharing.

Related reference:

START irlmproc (z/OS IRLM) (DB2 Commands)

DB2 system services address space (DSN1MSTR) storage requirements

A variety of system-related functions are performed in the DB2 system services address space. The system services address space needs less space than the database services address space.

Recommendation: Specify 0 KB for the system services address space. The default start up procedure is DSN1MSTR.

DB2 database services address space (DSN1DBM1) storage requirements

The DB2 database services address space is the largest DB2 address space.

This address space uses storage above the 2 GB bar. The default start up procedure is DSN1DBM1. First, plan for a minimum of 90 MB in this address space, with 2 MB below the 16 MB line.

Most modules reside in the 31-bit extended private area. Storage usage below the 16-MB line is typically less than 2 MB.

Allied agent address space storage requirements

In DB2, any user address space that attaches to DB2 is referred to as an allied agent address space. This address space can include Resource Recovery Services attachment facility (RRSAF), TSO attachment facility, IMS attachment facility, CICS attachment facility, and batch address spaces.

The size of the DB2 attachment facility code in the allied agent address space depends on which attachment facilities you use. TSO requires about 130 KB for the DSN command. CAF, IMS, and RRSAF each require about 36 KB for the DB2 attachment facility code. For all attachment facilities, except CICS Transaction Server and RRSAF, the DB2 attachment facility code must run below the 16 MB line of virtual storage. Applications can run above the 16 MB line.

Related concepts:

CICS Transaction Server for z/OS DB2 Guide

Administrative task scheduler address space storage requirements

Each DB2 subsystem has a coordinated administrative task scheduler address space that it can start by using a z/OS started task procedure.

Therefore, if there are many DB2 subsystems running on a single z/OS system, there is a separate administrative task scheduler with a separate name for each one.

Two instances of the same administrative task scheduler cannot run simultaneously. To avoid starting up a duplicate administrative task scheduler, at startup the administrative task scheduler checks all of the address spaces control block first to ensure that there is no address space other than itself with the same name. If another address space with the same name is already up and running, the administrative task scheduler that is starting up immediately shuts down with a console error message. The administrative task scheduler can check only the address spaces that are in the same system, not in the entire Sysplex.

The administrative task scheduler address space stays up, even when DB2 comes down.

WLM-established stored procedures address spaces storage requirements

WLM-established stored procedures address spaces are WLM-established address spaces that can provide multiple isolated environments for stored procedures.

Each WLM-established stored procedures address space is associated with a Workload Manager environment.

DB2 for z/OS stored procedures support both main programs and subprograms; this support requires additional storage for each task control block (TCB). However, because you can run fewer programs in an address space, you can use less storage below the 16 MB line in each address space.

The amount of space that is required by a WLM-established stored procedures address space depends on the applications. Stored procedures that internally store a large amount of in-process work require more space than those that do not store much in-process work. Also, Java stored procedures require more space than others, such as COBOL stored procedures.

The overall amount of required space depends on the number of WLM-established stored procedures address spaces that are running, and the amount of space that is required for each.

Related tasks:

"Setting up a WLM application environment for stored procedures during installation" on page 543

"Setting up a WLM application environment for stored procedures during migration" on page 617

Refreshing a WLM application environment for stored procedures (DB2 Administration Guide)

Maximizing the number of procedures or functions that run in an address space (DB2 Performance)

Shared memory storage requirements

Shared memory is a type of virtual storage that allows multiple address spaces to easily address common storage. This memory resides above the 2 GB bar.

The shared memory object is created at DB2 startup, and all DB2 address spaces for the subsystem (DSN1DIST, DSN1DBM1, DSN1MSTR, and Utilities) are registered with z/OS to be able to access the shared memory object.

To define the size of the shared memory and the 64-bit common area, use the HVSHARE and HVCOMMON parameters of the IEASYSxx member in the parmlib concatenation.

Restriction: Ensure that you have defined a high enough value for HVSHARE to satisfy all component requests for shared memory within the z/OS image. The default value is 510 TB. DB2 Version 10 requires 128 GB of contiguous 64-bit shared private storage for each DB2 subsystem above the 2 GB bar. This storage is virtual, and it is controlled by the z/OS HVSHARE parameter in IEASYSxx. This storage is not backed at allocation, only as it is used. You can monitor the space usage with IFCIDs 0225 and 0217.

IFCID 0225 has statistics that you can use to determine the amount of currently defined storage and how much of it is currently allocated.

Related information:

z/OS IEASYSxx member (MVS Initialization and Tuning Reference)

Common service area storage requirements

Some of the DB2 load modules and control blocks are in common storage. Most of the space is in the extended common service area (ECSA) and 64-bit common storage.

With few exceptions, the CSA-resident load modules are link-edited with the residency attribute of RMODE(ANY). Most of the modules reside in ECSA (above the 16 MB line of virtual storage), as do most of the global control blocks. The IRLM control blocks are above the 2 GB bar.

Monitor your use of CSA and ECSA, and increase those values if necessary. By monitoring CSA below the 16-MB line, you can determine whether you need to increase the size of the ECSA.

When you IPL z/OS, you can override the CSA size with this syntax:

```
CSA=(a,b)
```

where:
- *a* is the number of kilobytes of CSA storage below the 16 MB line
- *b* is the number of kilobytes of ECSA storage above the 16 MB line

These values are rounded down (CSA) or up (ECSA) to the next 1 MB boundary.

To define the size of the 64-bit common area, you can use the HVCOMMON parameter of the IEASYSxx member in the parmlib concatenation.

DB2 Version 10 requires 64-bit common storage. DB2 requires that at least 6 GB of 64-bit common storage is defined for each DB2 subsystem that runs on a z/OS image.

Restriction: Ensure that you have defined a high enough value for HVCOMMON to satisfy requirements for all DB2 subsystems plus all other use on the LPAR. The default value is 66 GB. The 6 GB of 64-bit common storage that is requested for each member must be contiguous.

Related reference:

Common Service Area (CSA/Extended CSA) (MVS Initialization and Tuning Guide)

Calculating the storage requirement for the common service area:

You need to calculate the approximate storage requirement for the common service area (CSA). Under normal conditions, your virtual storage requirements in the CSA (below the 16-MB line) are unlikely to increase.

About this task

In a data sharing environment, if an IRLM performs member recovery or structure rebuilding, additional virtual storage is required. Any other increase in the amount of virtual storage that is needed occurs within the extended private area of the DB2 database address space and the extended private area of the distributed data address space.

Procedure

To calculate the approximate residual requirement for CSA (below the 16-MB line):

1. Start with up to 40 KB for each DB2 subsystem.
2. Add 24 KB for each started IRLM.

Calculating the storage requirement for the extended common service area:

You need to calculate the approximate requirement for the extended common service area (ECSA).

Procedure

To estimate storage that is needed for ECSA (above the 16-MB line) for each DB2 subsystem:

1. Start with 3 MB of ECSA for the base and the first 100 users.
2. Start with 0.1 MB for IRLM.
3. Add 1.9 MB for IRLM required trace buffers.
4. Add 1.9 MB for IRLM optional trace buffers.
5. Add 4 KB for each additional user.
6. Add 4 KB for each active remote thread.
7. Add 4 MB or more for instrumentation facility interface (IFI) buffers as requested by the monitoring programs.
8. Add up to 1 MB for each concurrently executing DB2 command.

What to do next

If you use the distributed data functions of DB2, you may find that you need more virtual storage. You can estimate how much your storage needs are likely to increase in the ECSA above the 16-MB line by adding the following amounts:

1. 1 KB for each conversation
2. 2 KB for each thread that uses distributed processing
3. 1 KB for each DB2 site in your network
4. 40 KB for code that relates to distributed processing

Specify this sum or a value that is larger than this sum as the second value of the CSA parameter of the IEASYS*xx* z/OS logical PARMLIB member. The logical PARMLIB is usually referred to as SYS1.PARMLIB. Specifying values that are too high is preferable to specifying values that are too low; making your values too low can result in a need to IPL z/OS. For example, if the ECSA size is too small, z/OS places DB2's global load modules and control blocks in CSA below the 16-MB line instead of above it. This can cause problems with coexisting z/OS subsystems.

Distributed threads use extended system queue area (ESQA) when they are paused. The amount of storage that is used might vary depending on the z/OS release that is used. Estimate 4 KB of ESQA for every active distributed thread.

Virtual storage requirements for storage pools and working storage

During the ISPF tailoring session, you specify values that the DSNTINST CLIST uses to calculate main storage size.

Recommendation: Determine these values based on your estimated application workload before you install or migrate DB2.

These values provide an estimate of the private area that is needed by the DSN1DBM1 address space, the largest of the DB2 address spaces. If the estimated virtual storage for the address space is not available, you can re-evaluate the sizes that you requested.

The calculations in this topic are planning estimates. The noted values do not provide the exact limits, but they indicate a reasonable range of values.

The sum of the following values must fit the region size that DB2 supports:
- VSAM data set control block storage size
- Working storage size

The CLIST adds a fixed code size to the sum of these values to determine the main storage size.

Storage pools that were previously below the 2 GB bar have now been moved above the 2 GB bar. Their values are no longer included in the region size calculation. These storage pools are the buffer pool, the sort pool, the record identifier (RID) pool, and the environmental descriptor manager (EDM) pool.

After you specify the values listed above, the CLIST calculates the EDM pool size and the size that is needed for the data set control blocks. The CLIST adds the working storage size and the fixed code size to update the region size that is used in the DB2 startup procedures. The CLIST also displays this information on CLIST calculations panel 1: DSNTIPC.

Use the formulas in this topic to estimate your storage needs. For your reference, the default values are included where appropriate.

Related concepts:

"DB2 database services address space (DSN1DBM1) storage requirements" on page 47

Related tasks:

Calculating virtual storage requirements for a dictionary (DB2 Performance)

Related reference:

"CLIST calculations panel 1: DSNTIPC" on page 490

Calculating buffer pool size

Buffer pools are areas of virtual storage that are used to satisfy the buffering requirements for one or more table spaces or indexes. All DB2 subsystems use virtual buffer pools, which are backed by central storage or auxiliary storage. Buffer pools are created above the 2 GB bar.

About this task

Virtual buffer pools: For best results, use at least 100 KB of buffer pool space for each concurrent user. A value of 300 KB or more is recommended for improved performance. Very simple SQL statements that access small amounts of data can require less than this amount. Complex SQL statements that access large amounts of data can require more than this amount. Dynamic SQL statements might require a dynamic bind, which requires additional buffers for catalog tables and indexes. If an SQL statement requires directory access, additional buffers are also required for directory page sets. UPDATE, DELETE, and INSERT statements require more buffers to keep updated pages in a buffer pool.

To efficiently support prefetch, allow 400 KB for each object that is concurrently accessed. For example, if you have a SELECT statement with a single table space scan, allow 400 KB for that statement. For another example, suppose you have a SELECT statement with a three-table sort merge join that uses an index for each table, and a parallel degree of 4. To obtain a rough estimate, calculate the total number of objects that are accessed and multiply the number of objects by 400 KB and then by the degree of parallelism. In this example, the equation would be (3 tables + 3 indexes + 1 work file for the sort) x 400 KB x 4. The result is 11200 KB

that is recommended for this statement. Typically, these objects should be defined in separate buffer pools to provide better performance control. For example, you should have at least three separate buffer pools: one for indexes, one for tables, and one for work files.

During installation, you can set the buffer pool sizes on the installation panels. Later, you can use the ALTER BUFFERPOOL command to alter the sizes and other attributes of as many as 50 buffer pools for 4-KB page sets, 10 buffer pools for 8-KB page sets, 10 buffer pools for 16-KB page sets, and 10 buffer pools for 32-KB table spaces. The ALTER BUFFERPOOL command can make the changes dynamically while DB2 is running.

Important: Do not allocate more storage for buffer pools than available real storage for buffer pools. If you attempt to use more than the available real storage, performance will suffer.

DB2 limits the total amount of storage that is allocated for virtual buffer pools to approximately twice the amount of real storage. If you specify more than this amount for virtual buffer pools, DB2 allocates buffer pools during startup until twice the amount of real storage is used. DB2 then allocates the remaining buffer pools as follows:

Page size	Number of pages
4 KB	2000
8 KB	1000
16 KB	500
32 KB	250

After these storage limits have been reached, you cannot increase the amount of virtual buffer pool storage unless you increase the amount of real storage that is available to the z/OS image.

Procedure

To calculate the virtual buffer pool sizes for your subsystem:

Use the following table.

Table 27. Virtual buffer pool size calculation

Virtual buffer pool calculation			Default
Buffers for BP0	____ x 4 KB = _____	20 000 x 4 KB	= 80 000 KB
Buffers for BP1	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
Buffers for BP2	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
.			
.			
.			
Buffers for BP49	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
Buffers for BP8K0	+____ x 8 KB = _____	+ 1000 x 8 KB	= 8000 KB
Buffers for BP8K1	+____ x 8 KB = _____	+ 0 x 8 KB	= 0 KB
.			
.			
Buffers for BP16K0	+___ x 16 KB = ____	+ 500 x 16 KB	= 8000 KB
Buffers for BP16K1	+___ x 16 KB = ____	+ 0 x 16 KB	= 0 KB
.			

Table 27. Virtual buffer pool size calculation (continued)

Virtual buffer pool calculation			Default
.			
Buffers for BP32K	+___ x 32 KB = ____	+250 x 32 KB	= 8000 KB
.			
.			
Buffers for BP32K9	+___ x 32 KB = ____	+ 0 x 32 KB	= 0 KB
	= ____		= 104 000KB

Related concepts:

Changing the size of the group buffer pool (DB2 Data Sharing Planning and Administration)

Calculating sort pool size

The DB2 sort process uses two kinds of storage: local storage and buffer pool storage. Sort pool storage and buffer pools have storage above the 2 GB bar.

Calculating sort pool storage in local storage:

The sort process creates fixed-length storage pools in local storage for internal sort structures and work areas. Local storage is created above the 2 GB bar at allocation time.

About this task

The DB2 sort work area (in-memory) has the following storage boundaries for **each** concurrent sort operation:

Minimum sort storage = 240 KB
Maximum sort storage = 128,000 KB

DB2 initially allocates storage for each sort and gradually adds more storage until the maximum sort work area limit is reached or the maximum number of nodes (32,000) are populated at the bottom of the sort tree, whichever occurs first. With 32,000 nodes at the bottom of the sort tree, the average run size for each sorted string is 64,000 records.

The default size of the sort pool is 10,000 KB, but you can override this default value by entering the desired sort pool size on installation panel DSNTIPC.

Procedure

To estimate the required storage for a sort pool:

Use the following formula:

```
32000 * (16 + sort key length + sort data length)
```

Related tasks:

Improving the performance of sort processing (DB2 Performance)

Related reference:

"CLIST calculations panel 1: DSNTIPC" on page 490

Adjusting sort pool storage in buffer pool storage:

Sort processing uses pages in the DB2 buffer pool for its initial input, for work files that contain intermediate results, and for the final output.

About this task

The buffer pools are not always dedicated to sort work files. The amount of sort activity determines how much the buffer pools are used. For heavy sort activity, sort records that have been written to the work files are temporarily written to the disk until buffer pool space becomes available.

DB2 considers the buffers that it uses for work files as sequentially accessed pages.

Procedure

To adjust the percentage of the buffer pool that is used for work files:

Use the VPSEQT parameter of the ALTER BUFFERPOOL command. If a buffer pool is used only for work files, you might set VPSEQT to 100%. If you do not have enough allocated storage to complete sort processing, you must allocate more disk space for the work file database.

Related concepts:

"Work file database storage requirements" on page 35

Calculating RID pool size

The *RID pool* is an area of local storage that is reserved for record identifier (RID) sort processing, including RID list sort processing.

About this task

The RID pool is created at startup, but no space is allocated until RID storage is needed. When RID storage is needed, it is allocated above the 2 GB bar in 32 KB blocks, which are known as RID blocks.

Consider the following when you calculate RID pool size:

- Regardless of the current RID pool usage, each agent that requests a new RID map and RID list is granted two (32 KB) RID blocks (1 RID map and 1 RID list).
- If a thread needs another RID list block to be added to the existing RID map and RID list, the next 32 KB block is granted only if the RID pool usage is below the maximum RID pool size. It is possible for the RID pool size to be larger than the setting of the MAXRBLK subsystem parameter because every new agent is granted at least one RID map and one RID list.
- Startup RID storage = 0 (but acquired in 32 KB blocks as needed)
- Maximum RID storage = 10,000 MB

The default size of the RID pool is 400,000 KB, but you can override this default value by entering the desired RID pool size on installation panel DSNTIPC.

Procedure

To estimate the required storage for the RID pool:

Use the following formula:

```
number of concurrent RID processing activities * average number of RIDs *
2 * 5 (bytes per RID)
```

Related tasks:

Increasing RID pool size (DB2 Performance)

Calculating EDM pool size

The environmental descriptor manager (EDM) controls areas of virtual storage that hold cached objects. These areas are called pools, and thus the term *EDM pool* is frequently used to describe these areas of virtual storage.

About this task

In DB2 Version 10, EDM pools are all located above the 2 GB bar. These pools are the prepared-statement cache pool, the skeleton pool for packages and plans, and the database descriptor (DBD) pool. In previous releases, EDM pool storage was used for thread copies of plans and packages. In Version 10, this storage use is deprecated, and the thread copies of plans and packages are allocated in other storage areas by EDM.

The CLIST generates the default sizes for these pools, which are displayed on CLIST calculations panel 1: DSNTIPC. These estimations do not account for every factor that affects EDM pool size. Your actual EDM pool size might vary.

Related reference:

"CLIST calculations panel 1: DSNTIPC" on page 490

Calculating EDM pool space for the prepared-statement cache:

When you use the cache, prepared statements are stored in the EDM pool, as are static SQL statements. The number of prepared statements that are stored in the cache depends on the characteristics of the dynamic SQL statements that your application executes. This EDM pool space is above the 2-GB bar.

About this task

One type of application typically benefits from caching prepared statements, whereas the other type usually does not. The first type of application uses dynamic SQL statements that are embedded in an application and used repeatedly. Applications and queries with this type of SQL benefit most from caching prepared statements because the statement can be used from the cache. However, applications that contain SQL statements that are infrequently used pay the cost of being added to the cache. For example, queries from DB2 QMF™ are likely to be prepared and executed only once. Caching prepared statements does not benefit applications that extensively use this kind of SQL.

The installation CLIST calculates the size of the EDM pool for the prepared-statement cache and displays this value on panel DSNTIPC. Calculated values of less than 5000 KB are forced to 5000 KB. Values above 1,048,576 KB are reduced to 1,048,576 KB.

Procedure

To estimate the space that is needed for the prepared-statement cache:

Use the following calculation:

```
EDMSTMTC = 4 * [(EDMSCWK * 3) + (EDMSCWK / 8)]
```

EDMSCWK is calculated as follows:

```
EDMSCWK = [(NUMTHRDS / 1024) + (NUMPLANS / 4)] * SPCPLANE + 3264
```

where:

- *NUMTHRDS* is the maximum number of concurrent threads (CTHREAD) plus the maximum number of DBATS (MAXDBAT)
- *NUMPLANS* is the number of plans in the subsystem
- *SPCPLANE* is the estimated space for plans. It is calculated as follows:

  ```
  SPCPLANE = TRUNC((NUMSTMTE * NUMSTMTX + 2047) / 1024)
  ```

  where:
  - *NUMSTMTE* is the average number of SQL statements that are issued per plan (EXECUTED STMTS field on panel DSNTIPD)
  - *NUMSTMTX* is the calculated space for the average SQL statement. It is calculated as follows

    ```
    NUMSTMTX = 1433 + (NUMSTMTL - 1) * 200
    ```

    where *NUMSTMTL* is the average number of tables per SQL statement (TABLES IN STMT field on panel DSNTIPD)

Calculating EDM pool space for the skeleton copies of packages:

This EDM pool space is above the 2 GB bar and is used to store skeleton copies of packages (SKPT). The initial value for this pool is set by the EDM_SKELETON_POOL field on installation panel DSNTIPC.

About this task

The installation CLIST calculates the value of EDM_SKELETON_POOL. Calculated values of less than 5120 KB are forced to 5120 KB. Values above 2,097,152 KB are reduced to 2,097,152 KB.

Procedure

To estimate the space that is needed for the skeleton copies of packages:

Use the following calculation:

```
EDM_SKELETON_POOL = max-plans * statement-size * statements-executed + 3264
```

where:

- *max-plans* is the maximum number of unique plans that you want in the EDM pool at any given time. Estimate this value by taking one fourth of the value of the PLANS field on installation panel DSNTIPD.
- *statement-size* is the average statement size. To calculate the average statement size, add 1.4 KB for the single table statements to the product of 0.2 KB for each additional table in the statement multiplied by one less than the value of the TABLES IN STMT field on installation panel DSNTIPD. (1.4 KB + (0.2 KB * (TABLES IN STMT -1)))
- *statements-executed* is the average number of executed statements. The CLIST uses the value of the EXECUTED STMTS field on installation panel DSNTIPD.

Calculating EDM pool space for database descriptors:

The final part of the EDM pool calculation involves the space for database descriptors (DBDs). This portion of the EDM pool is above the 2-GB bar.

About this task

The installation or migration CLISTs for data sharing increase the calculated value for DBDs by 25 percent. Calculated values of less than 5000 KB are forced to 5000 KB; values above 2097152 KB are reduced to 2097152 KB.

Procedure

To calculate the EDM pool space for database descriptors:

1. Estimate the number of concurrently open databases.
   - Let *NUMCONCR* be the value of the MAX USERS field on installation panel DSNTIPE.
   - Let *NUMCONRM* be the value of the MAX REMOTE ACTIVE field on installation panel DSNTIPE.
   - Let *concplan* be the sum of *NUMCONCR* and *NUMCONRM*.
   - Let *DSCBSIZE* be 16. This is the estimated number of bytes of below-the-line storage per data set, for operating system control blocks, such as allocation, ENQ, and SMF.
   - Let *NUMINDEX* be the estimated average number of indexes per table.
   - Let *NUMTABLE* be the estimated number of tables per database (TABLES field on DSNTIPD).
   - Let *NUMTABSP* be the estimated number of table spaces per database (TABLE SPACES field on DSNTIPD).

   a. Calculate the maximum number of data sets by using the following formula:

      ```
      MAXMAXDS = (7400 - concplan) * 1024 / DSCBSIZE
      ```

   b. Calculate the number of concurrent databases using the following formula:

      ```
      Concurrent_DBs = MAXMAXDS / (NUMTABLE * NUMINDEX + NUMTABSP )
      ```

2. Estimate the average size of the database descriptor

   The database descriptor size is 12 KB for the default values. The database descriptor size depends on the number of table spaces, tables, indexes, columns, partitions, referential relationships, table check constraints, and index keys in the database. The DSNTINST CLIST contains the algorithm for calculating the DBD size. The maximum size of a database descriptor is 25% of the size of the EDM pool. Therefore, you need to ensure that the EDM pool size is at least four times the estimated size of your largest database descriptor.

   Estimate the number of table spaces, indexes, tables, referential relationships, and other in the database by using the following variables:
   - Let *COLLEN* be the SYSCOLUMNS row length.
   - Let *FORLEN* be the SYSFOREIGNKEYS row length.
   - Let *IPTLEN* be the SYSINDEXPART row length.
   - Let *IXSLEN* be the SYSINDEXES row length.
   - Let *KEYLEN* be the SYSKEYS row length.
   - Let *NUMCOLUM* be the estimated number of columns per table (COLUMNS field on DSNTIPD).
   - Let *NUMINDEX* be the estimated average number of indexes per table.
   - Let *NUMINDKY* be the estimated average number of index keys.
   - Let *NUMPART* be the estimated average number of partitions per table.
   - Let *NUMREL* be the estimated average number of relations per table.

- Let *NUMTABLE* be the estimated number of tables per database (TABLES field on DSNTIPD).
- Let *NUMTABSP* be the estimated number of table spaces per database (TABLE SPACES field on DSNTIPD).
- Let *PCTPTS* be the estimated average percentage of partitioned table spaces.
- Let *RELLEN* be the SYSRELS length row length.
- Let *TABLEN* be the SYSTABLES length row length.
- Let *TPTLEN* be the SYSTABLEPART row length.
- Let *TSPLEN* be the SYSTABLESPACE row length.

a. Calculate the size needed for the table spaces with the following formula:

```
DBD_TSs = NUMTABSP * (TSPLEN + TPTLEN * NUMPART * PCTPTS)
```

b. Calculate the size needed for the indexes with the following formula:

```
DBD_IXs = (NUMTABLE * NUMINDEX * (IXSLEN + KEYLEN * NUMINDKY + IPTLEN * NUMPART * PCTPTS))
```

c. Calculate the size needed for the tables with the following formula:

```
DBD_TBs = NUMTABLE * (TABLEN + COLLEN * NUMCOLUM)
```

d. Calculate the size needed for the referential relationships with the following formula:

```
DBD_RefRels = NUMTABLE * (NUMREL + RELLEN + FORLEN * NUMINDKY)
```

e. Calculate the size needed for the other factors with the following formula:

```
DBD_Other = (NUMTABLE * (1 + NUMINDEX) + NUMTABSP)
```

f. Calculate the size needed for the DBD with the following formula:

```
DBD_size = [(DBD_TSs + DBD_IXs + DBD_TBs + DBD_RefRels + DBD_Other + 4095) / 4096] * 4
```

3. To calculate the total amount of EDM pool space that is needed for DBDs, multiply the number of concurrently open databases by the average size of the database descriptor.

```
Concurrent_DBs * DBD_size
```

Calculating data set control block storage

Calculate the data set control block storage requirements by determining the total number of open data sets (DSMAX).

Procedure

To determine the main storage requirements for your data set control blocks:

- Let *concdb* be the number of concurrent databases, which is specified on installation panel DSNTIPE.
- Let *tables* be the number of tables in each database, which is specified on installation panel DSNTIPD.
- Let *indexes* be the number of indexes in each table. The installation CLIST sets this variable to 2.
- Let *pctpts* be the percentage of partitioned table spaces.
- Let *avgpart* be the average number of partitions in each partitioned table space.
- Let *partts* be the average number of partitioned table spaces per database.
- Let *nonparti* be the number of non-partitioning indexes per partitioned table space.

1. Calculate the total number of open tables with the following formula:

```
opntab = concdb * tables
```

2. Calculate the number of open data sets for partitioned table spaces with the following formula:

```
opnptsds = opntab * (pctpts / 100) * avgpartconcdb *
(((tables *  indexes) + tblspaces) +
((2 * partts * avgpart) - (2 * partts)))
```

3. Calculate the number of open data sets with the following formula:

```
concdb * (((tables *  indexes) + tblspaces) +
((2 * partts * avgpart) - (2 * partts)))
```

   You can modify DSMAX by editing job DSNTIJUZ. The maximum number of concurrently open data sets that is allowed by DB2 is 100,000.

4. To calculate the main storage that is required for your data set control blocks, use the following formula:

```
DSMAX * 1.3 KB
```

Example

The default, as calculated by the DSNTINST CLIST, is

```
9960 * 1.3 KB = 12948 KB
```

This method of calculation ignores partitioned table spaces and partitioning index spaces. It also assumes that all data sets in the database are open if the database is in use. You could enter a smaller value for the number of concurrent databases if only a few of the data sets in a database are typically opened. The larger the value of DSMAX, the longer data sets stay open.

Recommendations:

- Move the scheduler work area (SWA) above the 16 MB line of z/OS virtual storage by using JES initialization statements, JES exit routines, or the SMF exit routine (IEFUJV). This way, you can save approximately 1 KB for each open data set in virtual storage below the 16 MB line and avoid potential storage errors. To determine the amount of storage that is needed below the 16-MB line, use 0.016 KB for the multiplication factor if the SWA is above the line or 1.2 KB for the multiplication factor if the SWA is below the line. The preceding calculations and the calculations in the CLIST presume that the SWA is above the line.
- If you do not move the SWA above the 16 MB line, you should not use a DSMAX value greater than 5000.

Related concepts:

How DB2 determines the initial value of DSMAX (DB2 Performance)

Calculating working storage requirements

Working storage is private virtual storage in the DB2 database services address space, which is primarily above the 2 GB bar. Working storage is used to hold temporary data that is needed for the processing of an SQL request.

About this task

If you use dynamic SQL, you need more working storage and less EDM pool space than if you use static SQL. If you use static SQL, you have larger plan sizes, typically varying from 15 KB to 1 MB. Typical sites would use about 300 KB for each thread of working storage for dynamic SQL users, and 100 KB per thread for static SQL users. A *thread* is a structure that describes an application connection to DB2. The CLIST does not include information about open compressed table spaces. Compression dictionaries are stored above the 2-GB bar. Therefore, if you use compressed table spaces, you need additional storage.

Procedure

To estimate the amount of working storage:

1. Start with 600 KB.
2. Add 40 KB for each concurrent DB2 user. This value is specified by the CTHREAD subsystem parameter.
3. Add 40 KB for each remote DB2 user. This value is specified as MAXDBAT subsystem parameter.
4. Estimate the amount of working storage that is needed by using the following formula:

```
600 KB + (CTHREAD + MAXDBAT) * 40 +
(CONDBAT - MAXDBAT) * 4
```

Example

The default, as calculated by the DSNTINST CLIST, is:

```
600KB + (200 + 200) * 40 + (1000 - 200) * 4 = 55800 KB
```

Related concepts:

How DB2 allocates allied threads (DB2 Performance)

Related tasks:

Compressing your data (DB2 Performance)

Related reference:

"MAX USERS field (CTHREAD subsystem parameter)" on page 334

"MAX REMOTE CONNECTED field (CONDBAT subsystem parameter)" on page 335

"MAX REMOTE ACTIVE field (MAXDBAT subsystem parameter)" on page 334

Calculating virtual storage below the 16-MB line

This calculation produces an estimate of virtual storage constraints below the 16-MB line in the DB2 database services address space.

About this task

Most of the needed virtual storage is in extended private storage and storage above the 2 GB bar, including the buffer pool, the EDM pool, and almost all of the code and working storage. This is the difference between the total storage and the estimated region size. The estimated region size does not include extended private storage or storage above the 2 GB bar. It includes only the data set control block storage size and some of the code.

Procedure

To estimate the size of storage below the 16-MB line:

Use the following formula:

```
600 KB + MAX USERS + MAX REMOTE ACTIVE + (DSMAX * 0.016)
```

Example

The default, as calculated by the DSNTINST CLIST, is

```
600 KB + 200 + 200 + (9960 * 0.016 KB), = 1160 KB
```

If the scheduler work area (SWA) is above the 16-MB line, multiply the number of data sets by 0.016 KB; if the SWA is below the 16-MB line, multiply the number of data sets by 1.2 KB.

Recommendation: If you use DSMAX of more than 5000, your SWA should be above the 16-MB line.

The CLIST calculations panel, DSNTIPC, displays storage sizes that are calculated by the DSNTINST CLIST.

Related reference:

"CLIST calculations panel 1: DSNTIPC" on page 490

Calculating real storage requirements

DB2 can use real storage to reduce I/O and processor times and to improve response time and throughput. The amount of real storage that DB2 needs varies greatly.

About this task

Some users find that their organizations need several times the estimates listed below, whereas others need less. The amount of storage is an important parameter in DB2 performance. Performance monitoring programs give you a more accurate estimate of your storage requirements than the formulas in this topic because these programs can take specific environmental characteristics into account.

For the DB2 buffer pools, the EDM pool, log buffers, and working storage, the amount of real storage must be the same as the amount of virtual storage. Paging activity in the buffers is an indication of a problem. If you do not have enough real storage, you might need to reduce your buffer pool size, which allows fewer concurrent users. You also need space to contain locks, the working set of code in all address spaces, log buffers, and common storage space. In Version 10, log buffers are permanently page-fixed. Therefore, the entire size of the output buffer needs to be used to estimate real storage. Because some of the figures that are used in virtual storage calculations are maximums, whereas the real storage figures typically use activity for the peak, changes are needed in the calculations. The virtual storage figures concentrate on the most constrained address space, but real storage work must include them all. For more information about each category, see the information specified in the first two tables below.

Some storage for castout engines or DASD read and write of a compressed index might be fixed through a PAGEFIX operation. More castout engines or more compressed indexes can cause more storage to be permanently fixed. MVS Real Storage Manager (RSM) controls where the PAGEFIX occurs in real storage, and it can occur below the 16 MB line. If you have concerns about this behavior, consult with IBM Software Support for MVS RSM.

Procedure

To estimate real storage requirements:

1. Multiply the virtual size of each storage area by the factor that is indicated in the following table to estimate the real size for that area.

Table 28. Real storage size calculation

Category	Default	Virtual size	× Factor	= Real size
Buffer pools	104,200 KB	_____	1.0	_____
Sort pool	10,000 KB	_____	0.5	_____
RID pool	400,000	_____	0.5	_____
EDM pool	33,600 KB	_____	1.0	_____
Data set size	17,928 KB	_____	0.6	_____
Code size + 1100 (4 address spaces)	40,000 KB	40,000 KB	0.5	20,000 KB
Working storage + (DSCF + DDF)	55,800 KB	_____	1.0	_____
Log buffers	4000 KB	_____	1.0	_____
Lock space	5000 KB	_____	0.4	_____
Common storage	2160 KB	_____	0.4	_____
			Total real storage size =	_____

2. Plan to add real storage for additional users.

   The following table uses rough estimates to approximate the amount of additional real storage that is needed by several kinds of users. If you have more concurrent users, plan to add real storage.

Table 29. Additional real storage for more users

Type of user	Additional real storage
Transaction	150 KB
Query	400 KB
Batch	700 KB

Recommendation: Look at the OMEGAMON report for IFCID 0225, which calculates the virtual storage per user, to see if your estimated real storage requirements are sufficient.

Example

The following table shows the default real storage size calculations.

Table 30. Default real storage size calculation

Category	Default virtual size	× Factor	= Real size
Buffer pools	104,000 KB	1.0	104,000 KB
Sort pool	10,000 KB	0.5	5000 KB
RID pool	400,000	0.5	200,000
EDM pool	33,600 KB	1.0	33,600 KB
Data set size	17,928 KB	0.6	10,756.8 KB
Code size + 1100 (4 address spaces)	40,000 KB	0.5	20,000 KB
Working storage + (DSCF + DDF)	55,800 KB	1.0	55,800 KB
Log buffers	4000 KB	1.0	4000 KB
Lock space	5000 KB	0.4	2000 KB
Common storage	2160 KB	0.4	864 KB

Table 30. Default real storage size calculation (continued)

Category	Default virtual size	× Factor	= Real size
		Total real storage size =	436,021 KB

Related tasks:

"Calculating buffer pool size" on page 52
"Calculating EDM pool size" on page 56
"Calculating working storage requirements" on page 60

Planning for performance of DB2

When you install or migrate DB2, consider your performance goals for your subsystem or data sharing member. Use z/OS Workload Manager (WLM) to define these goals.

Also, consider the following guidance:

- Monitoring and managing performance are more difficult with few plans.
- Never-ending threads and plan reuse accumulate EDMPOOL storage and working storage.
- Each COMMIT statement needs package accounting.
- Consider recycling threads.
- Finding unused packages and cleaning up mistakes are difficult unless ownership is known.
- Performance monitoring may require analyzing everything in a CICS region instead of just one plan.
- Plan is a required short name.
- Package name is optional, long, multipart, and not always easy to find.
- Consider differentiating by plan within a CICS region and having at least 5 to 20 plans within a region. You can work faster and make changes on a more granular basis.

Related concepts:

z/OS performance options for DB2 (DB2 Performance)

Related tasks:

Controlling resource usage (DB2 Performance)

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Converting EXPLAIN tables (before migration)

In DB2 10, EXPLAIN tables that are in pre-Version 8 format are no longer supported. EXPLAIN tables that are in Version 8 or Version 9 format and EBCDIC-encoded EXPLAIN tables are deprecated. Before you migrate to Version 10, convert your EXPLAIN tables to Version 8 or Version 9.1 format and Unicode encoding.

About this task

Starting in DB2 10 conversion mode (from both Version 8 and Version 9.1), DB2 returns SQLCODE -20008 reason code 2 for statements or commands that invoke EXPLAIN processing if EXPLAIN tables are in a pre-Version 8 format. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9 format. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned.

Because converting your EXPLAIN tables might be time-consuming, you should complete this task as far before migration as possible.

Procedure

To convert your EXPLAIN tables before migration:

- From V8 If you have a Version 8 subsystem, follow the steps in Converting EXPLAIN tables (in Version 8) From V8
- From V9.1 If you have a Version 9.1 subsystem, follow the steps in Converting EXPLAIN tables (in Version 9.1). From V9.1

What to do next

After you migrate to Version 10, you will need to also convert your EXPLAIN tables to Version 10 format.

Changes that might affect your migration from Version 8

If you are migrating from DB2 Version 8, be aware of the changes that might affect your migration.

Proper planning should alleviate any system inconveniences.

Compatibility mode is now called conversion mode (from Version 8 or Version 9.1)

In DB2 Version 10, the terms *conversion mode from Version 8* and *conversion mode from Version 9.1* replace the term *compatibility mode*. Also, the terms *conversion mode* from Version 8* and *conversion mode* from Version 9.1* replace the term *compatibility mode**.

Availability of new functions in conversion mode from Version 8 and new-function mode

In most cases, new functions are available only in new-function mode. However, optimization and virtual storage enhancements also work in conversion mode from Version 8.

The DB2 10 for z/OS information assumes that your DB2 subsystem is running in Version 10 new-function mode. Generally, new functions that are described, including changes to existing functions, statements, and limits, are available only in new-function mode. The exceptions to this general statement are new and changed optimization enhancements, which are also available in conversion mode

from Version 8 unless stated otherwise. Also, the convergence of the temporary database into the work file temporary database applies to conversion mode from Version 8.

If you migrate from Version 8, you cannot use Version 9.1 new-function mode functions in Version 10 conversion mode from Version 8. You can use those functions in Version 10 new-function mode. Any functions that are available in conversion mode from Version 8 are also available in all other modes before new-function mode.

Application and SQL release incompatibilities for migration from Version 8

When you migrate from DB2 Version 8 to Version 10, be aware of the application and SQL release incompatibilities.

GUPI

Plan for the following changes in Version 9.1 and Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changes in BIND PACKAGE and BIND PLAN defaults (change introduced in Version 9.1)

In DB2 Version 9, the default value for bind option CURRENTDATA is changed from YES to NO. This applies to the BIND PLAN and the BIND PACKAGE subcommands, the CREATE TRIGGER SQL statements for trigger packages, and the CREATE PROCEDURE and the ALTER PROCEDURE ADD VERSION SQL statements for SQL PL procedure packages. Specifying NO for CURRENTDATA is the best option for performance.

The default value for bind option ISOLATION is changed from RR to CS. This applies to the BIND PLAN and the remote BIND PACKAGE subcommands. For the BIND PACKAGE subcommand, the current default (plan value) stays. The default change does not apply to implicitly built CTs (for example, DISTSERV CTs).

If DBPROTOCOL(PRIVATE) is specified, the Version 10 BIND or REBIND command parser issues error message DSNT225I and fails the BIND or REBIND request to prevent any outbound private protocol communications as a requester. DSNT225I indicates that DBPROTOCOL(PRIVATE) is no longer a supported bind option.

All BIND statements for plans and packages that are bound during the installation or migration process specify the ISOLATION parameter explicitly, except for routines that do not fetch data. The current settings are maintained for compatibility.

Plan for the XML data type (change introduced in Version 9.1)

Drop any user-defined data types with the name “XML” to prevent problems with the new Version 9 built-in XML data type. You can re-create the existing user-defined data types with new names.

Changes to XMLNAMESPACES (change introduced in Version 9.1)

In DB2 Version 8, in the XMLNAMESPACES function, if the XML-namespace-uri argument has a value of http://www.w3.org/XML/1998/namespace or http://www.w3.org/2000/xmlns/, DB2 does not issue an error. Starting in Version 9 conversion mode, DB2 issues an error.

Changes to serialization of empty elements (change introduced in Version 9.1)

In Version 8, DB2 serializes empty XML elements in a different way than it serializes them in Version 10. In Version 8, empty element "a" is serialized as `<a></a>`. Starting in Version 9 conversion mode, empty element "a" is serialized as `<a/>`.

Availability of LOB or XML values in JDBC or SQLJ applications with progressive streaming (change introduced in Version 9.1)

In previous releases, if a JDBC or SQLJ application retrieves LOB data into an application variable, the contents of the application variable are still available after the cursor is moved or closed. Version 9 supports streaming. The IBM Data Server Driver for JDBC and SQLJ uses progressive streaming as the default for retrieval of LOB or XML values. When progressive streaming is in effect, the contents of LOB or XML variables are no longer available after the cursor is moved or closed.

Adjust applications that depend on error information that is returned from DB2-supplied stored procedures (change introduced in Version 9.1)

Adjust any applications that call one of the following stored procedures and then check and process the specific SQLCODE or SQLSTATE that is returned by the CALL statement:

- SQLJ.INSTALL_JAR
- SQLJ.REMOVE_JAR
- SQLJ.REPLACE_JAR
- SQLJ.DB2_INSTALL_JAR
- SQLJ.DB2_REPLACE_JAR
- SQLJ.DB2_REMOVE_JAR
- SQLJ.DB2_UPDATEJARINFO

Starting in Version 9, these stored procedures return more meaningful SQLCODEs and SQLSTATEs than they return in previous releases of DB2. The other input and output parameters of these stored procedures have not changed.

For example, the following application needs to change because -20201 is no longer the SQLCODE that is returned. Successful execution (SQLCODE 0) is not affected.

```
CALL SQLJ.REMOVE_JAR(...)
IF (SQLCODE = -20201) THEN
DO;
...
END;
```

Some materialized query tables should be dropped (change introduced in Version 9.1)

Before migrating to conversion mode from Version 8, drop all materialized query tables that are based on the SYSIBM.SYSROUTINES catalog table. During migration to conversion mode from Version 8, if any materialized query tables are based on the SYSIBM.SYSROUTINES catalog table, SQLCODE -750 is issued.

Before migrating to enabling-new-function mode from Version 8, drop all materialized query tables that are based on the SYSIBM.SYSPACKSTMT catalog table. During migration to enabling-new-function mode from Version 8, if any materialized query tables are based on the SYSIBM.SYSPACKSTMT catalog table, SQLCODE -750 is issued.

Fully define objects (change introduced in Version 9.1)

Ensure that you do not have any incomplete object definitions in your DB2 Version 8 catalog. For example, if a table has a primary or unique key defined but the enforcing primary or unique key index does not exist, the table definition is considered incomplete. You need to complete or drop all such objects before you begin migration because their behavior will be different in Version 10. For example, if you attempt to create an enforcing primary key index to complete a table definition in Version 10 and the residing table space is implicitly created, the index is treated as a regular index instead of an enforcing index.

Changes to PL/I applications with no DECLARE VARIABLE statements (change introduced in Version 9.1)

For PL/I applications with no DECLARE VARIABLE statements, the rules for host variables and string constants in the FROM clause of a PREPARE or EXECUTE IMMEDIATE statement have changed. A host variable must be a varying-length string variable that is preceded by a colon. A PL/I string cannot be preceded by a colon.

Automatic rebind of plans and packages created before DB2 Version 6

If you have plans and packages that were bound before DB2 Version 6 and you specified YES or COEXIST in the AUTO BIND field of panel DSNTIPO, DB2 Version 10 autobinds these packages. Thus, you might experience an execution delay the first time that such a plan is loaded. Also, DB2 might change the access path due to the autobind, potentially resulting in a more efficient access path.

If you specify NO in the AUTO BIND field of panel DSNTIPO, DB2 Version 10 returns SQLCODE -908, SQLSTATE 23510 for each attempt to use such a package or plan until it is rebound.

IBMREQD is no longer reliable as a release dependency mark

The IBMREQD field in DB2 catalog tables is no longer a reliable indicator for determining release dependencies. Use the RELCREATED or RELBOUND fields instead.

Changed behavior for ODBC data conversion for the SQL_BINARY type (change introduced in Version 9.1)

In releases before Version 9.1, when ODBC applications used the SQL_BINARY type to bind parameter markers, ODBC mapped the SQL_BINARY type to CHAR FOR BIT DATA. In Version 10, when the DB2 server is in Version 10 new-function mode, ODBC maps SQL_BINARY to BINARY. Because CHAR FOR BIT DATA fields are padded with blanks, and BINARY fields are not padded, applications might experience differences in behavior.

For example, in releases before Version 9.1, if the target CHAR FOR BIT DATA column was shorter than the SQL_BINARY input host variable, and the truncated values were blanks, DB2 did not generate an error. In Version 10, if the target BINARY column is shorter than the SQL_BINARY input host variable, and the truncated values are hexadecimal zeroes, DB2 generates an error.

Changed behavior of the INSERT statement with the OVERRIDING USER VALUE clause (change introduced in Version 9.1)

When the INSERT statement is specified with the OVERRIDING USER VALUE clause, the value for the insert operation is ignored for columns that are defined with the GENERATED BY DEFAULT or GENERATED ALWAYS attribute.

DESCRIBE no longer returns LONG type values (change introduced in Version 9.1)

Because DB2 no longer stores LONG type values in the catalog, when you execute a DESCRIBE statement against a column with a LONG VARCHAR or LONG VARGRAPHIC data type, the DESCRIBE statement returns the values as VARCHAR or VARGRAPHIC data types.

DB2 enforces the restrictions about where a host variable array can be specified (change introduced in Version 9.1)

host-variable-array is the meta-variable for host variable arrays in syntax diagrams. *host-variable-array* is included only in the syntax for multi-row FETCH, multi-row INSERT, multi-row MERGE, and EXECUTE in support of a dynamic multi-row INSERT or MERGE statement. *host-variable-array* is not included in the syntax diagram for *expression*, so a host variable array cannot be used in other contexts. In previous releases, if you specified *host-variable-array* in an unsupported context, you received no errors. In Version 10, if a host variable array is referenced in an unsupported context, DB2 issues an error.

For more information about where you can specify the *host-variable-array* variable, see Host variable arrays in an SQL statement (DB2 Application programming and SQL).

DEBUGSESSION system privilege required for continued debugging of SQL procedures (change introduced in Version 9.1)

After you migrate to new-function mode, users that debug external SQL procedures need the DEBUGSESSION system privilege. (External SQL procedures were previously called SQL procedures in Version 8.) Only users of the new Unified Debugger enabled client platforms need this system privilege. Users of the Version 8 SQL Debugger-enabled client platforms do not need this system privilege.

Changes to the result length of the DECRYPT function (change introduced in Version 9.1)

The result length of the DECRYPT function is shortened to 8 bytes less than the length of the input value. If the result expands because of a difference between input and result CCSIDs, you must cast the encrypted data to a larger VARCHAR value before the DECRYPT function is run.

Changes to the result length and data type of some functions (change introduced in Version 9.1)

For the following built-in functions, if all parameters have data type DECFLOAT(*n*), the result has data type DECFLOAT(34):

- AVG
- STDDEV
- STDDEV_SAMP
- SUM
- VARIANCE
- VARIANCE_SAMP

For the following built-in functions, if the input is all integer or decimal values, or is a mixture of float or double values with integer or decimal values, the result data type is DOUBLE:

- CORRELATION
- COVARIANCE
- COVARIANCE_SAMP

COLTYPE column in SYSIBM.SYSCOLUMNS and SYSIBM.SYSCOLUMNS_HIST for LONG column types (change introduced in Version 9.1)

When new tables are created with LONG VARCHAR or LONG VARGRAPHIC columns, the COLTYPE values in SYSIBM.SYSCOLUMNS and SYSIBM.SYSCOLUMNS_HIST contain VARCHAR or VARG.

CREATEDBY column in SYSIBM.SYSDATATYPES, SYSIBM.SYSROUTINES, SYSIBM.SYSSEQUENCES, and SYSIBM.SYSTRIGGERS (change introduced in Version 9.1)

The CREATEDBY column might contain a different value than in previous releases of DB2. The column might contain a different value in static CREATE statements for distinct types, functions, and procedures or when a dynamic SQL statement sets the CURRENT SQLID value to a value other than USER.

DB2 returns all DSNWZP output in the same format as DB2 parameters (change introduced in Version 9.1)

In previous releases, DSNWZP returned the current setting of several system parameters in a format other than the one used by the system parameter macros. For example, DSN6SPRM expected the setting for EDMPOOL in kilobytes, and DSNWZP returned it in bytes. In Version 10, DB2 returns all DSNWZP output in the same format as DB2 parameters. Modify programs that call DSNWZP if they compensate for the format differences.

DB2 enforces the restriction that row IDs are not compatible with character strings when they are used with a set operator (change introduced in Version 9.1)

In previous releases, DB2 did not always enforce the restriction that row IDs are not compatible with character strings. In Version 10, DB2 enforces the restriction that row IDs are not compatible with string types when they are used with a set operator (UNION, INTERSECT, or EXCEPT).

Database privileges on the DSNDB04 database now give you those privileges on all implicitly created databases (change introduced in Version 9.1)

Because database privileges on the DSNDB04 database now give you those privileges on all implicitly created databases, careful consideration is needed before you grant database privileges on DSNDB04. For example, in Version 10, if you have the STOPDB privilege on DSNDB04, you also have the STOPDB privilege on all implicitly created databases.

Implicitly created objects that are associated with LOB columns require additional privileges (change introduced in Version 9.1)

In releases before Version 9.1, implicitly created objects that are associated with LOB columns do not require CREATETAB and CREATETS privileges on the database of the base table. Those implicitly created objects also do not require the USE privilege on the buffer pool and storage group that is used by the LOB objects. In Version 10, these privileges are required.

Adjust applications to use LRHCLR instead of LGDISCLR (change introduced in Version 9.1)

The LGDISCLR field in the DSNDQJ00 macro has been removed. Update applications that use the LGDISCLR value in the DSNDQJ00 mapping macro to determine whether a log record is a compensation log record to use the LRHCLR value instead.

Changed behavior for the CREATE statement (change introduced in Version 9.1)

You can no longer create databases with the AS TEMP clause or table spaces that specify TEMP as the target database. The TEMP database is no longer used by DB2. The WORKFILE database is the only temporary database.

The DECLARE statement and the work file database (change introduced in Version 9.1)

If you have applications in Version 8 that issue DECLARE SENSITIVE STATIC SCROLL CURSOR or DECLARE GLOBAL TEMPORARY TABLE statements, ensure that the work file database exists and that it has at least one table space with a 32 KB page size to avoid errors.

Adjust monitor programs that access OP buffers (change introduced in Version 9.1)

Adjust assignment strategies of monitor programs that access OP buffers. In Version 8, traces are left in a disabled state, which consumes CPU for trace data that cannot be retrieved. In Version 10, traces that are started with a destination of OPX choose the next available buffer that is not in use and traces are no longer left in a disabled state.

In addition, in Version 8, when the thread that owns an OP buffer terminates, OP traces are left in a disabled state and can be reactivated by starting another trace to that buffer. In Version 10, if an OP buffer terminates and the only destinations for the trace records are OP buffers, the traces that are started to that buffer are stopped. If an OP buffer terminates and the trace is started to both OP and non-OP destinations, the traces that are started to that buffer are modified to use non-OP destinations only.

The message format of DSNW128I and DSNW129I has changed, so modify automation that is based on those message formats.

Changed behavior for system-required objects (change introduced in Version 9.1)

After you migrate to Version 10 new-function mode, if the containing table space is implicitly created, you cannot drop any system-required objects, except for the LOB table space. This restriction applies even if you explicitly created these objects in a previous release. The following statements will not work properly if the system-required objects were implicitly created by DB2:

CREATE AUXILIARY TABLE
: If you issue a CREATE AUXILIARY TABLE statement and an auxiliary table that was implicitly created by DB2 already exists for the same base table, the CREATE AUXILIARY TABLE statement fails and DB2 issues SQLCODE -646, SQLSTATE 55017, and reason code 3.

CREATE LOB TABLESPACE
: If you issue a CREATE LOB TABLESPACE statement to create a LOB table

space in an implicitly created database, the CREATE LOB TABLESPACE statement fails and DB2 issues SQLCODE -20355, SQLSTATE 429BW, and reason code 1.

CREATE DATABASE
If you specify an eight-character database name that begins with DSN and is followed by exactly five digits in a CREATE DATABASE statement, the CREATE DATABASE statement fails and DB2 issues SQLCODE -20074, SQLSTATE 42939.

CREATE INDEX
If you create an index on a primary key, unique key, or ROWID column that is defined as GENERATED BY DEFAULT, the index will be treated as a regular index instead of an enforcing index.

CREATE AUXILIARY INDEX
If you issue a CREATE AUXILIARY INDEX statement and an auxiliary index that was implicitly created by DB2 already exists for the same base table, the CREATE AUXILIARY INDEX statement fails and DB2 issues SQLCODE -748, SQLCODE 54048, and reason code 3.

CREATE
If you issue a CREATE statement and do not specify an IN clause or table space name, and the default buffer pool is not large enough, DB2 chooses a 4 KB, 8 KB, 16 KB, or 32 KB buffer pool, depending on the record size. If you issue a CREATE statement and do not specify an IN clause or table space name, DB2 implicitly creates a partitioned-by-growth table space. If you drop the table, DB2 also drops the containing table space.

DROP TABLE
If you issue a DROP TABLE statement to drop an auxiliary table from a table space that was implicitly created by DB2, the DROP TABLE statement fails and DB2 issues SQLCODE -20355, SQLSTATE 429BW, and reason code 2.

DROP TABLESPACE
If you issue a DROP TABLESPACE statement to drop an implicitly created LOB table space, the DROP TABLESPACE statement fails and DB2 issues SQLCODE -20355, SQLSTATE 429BW, and reason code 2.

DROP INDEX
If you issue a DROP INDEX statement to drop an enforcing primary key, unique key, or ROWID index from a table space that was implicitly created, the DROP INDEX statement fails and DB2 issues SQLCODE -669, SQLSTATE 42917, and reason code 2. If you issue a DROP INDEX statement to drop an auxiliary index from a table space that was implicitly created, the DROP INDEX statement fails and DB2 issues SQLCODE -20355, SQLSTATE 429BW, and reason code 2.

Changes to INSERT, UPDATE, or DELETE statements on some indexes (change introduced in Version 9.1)

In Version 10, you cannot execute INSERT, UPDATE, or DELETE statements that affect an index in the same commit scope as ALTER INDEX statements on that index.

LOBs with a maximum length greater than 1 GB can now be logged (change introduced in Version 9.1)

In releases before Version 9.1, only LOBs with a maximum length of 1 GB or less could be logged. In Version 10, LOBs with a maximum length that is greater than 1 GB can be logged.

DB2 returns an error when a LOB value is specified for an argument to a stored procedure and the argument value is longer than the target parameter and the excess is not trailing blanks (change introduced in Version 9.1)

In releases before Version 9.1, DB2 did not return an error when a LOB value was specified for an argument to a stored procedure and the argument value was longer than the target parameter and the excess was not trailing blanks. DB2 truncated the data and the procedure executed. In Version 10, DB2 returns an error.

Changes to string formatting of decimal data

Explanation

DB2 Version 10 changed the formatting of decimal data by the CHAR and VARCHAR built-in functions and CAST specifications with a CHAR or VARCHAR result type. For input data that contains decimals, leading zeros are removed, and leading zeros are not added to values that did not already contain leading zeros. If the scale of the decimal value is zero, the decimal character is not returned. Also, the CHAR function no longer returns leading blanks for positive decimal values. The result of the CHAR function for decimal data is now consistent with the result of CAST(*decimal-expression* AS CHAR).

After migration to Version 10, packages that were bound before Version 10 use the old behavior for these functions. Materialized query tables and indexes on expressions that were created before Version 10 also continue to use the old behavior.

Views and inline SQL functions use the behavior of the SQL statement that references the object. It is possible for references to the same view or function in different applications to get different behavior for these functions or casts.

Possible impact to your DB2 environment

These changes might cause unexpected output from applications that use the CHAR or VARCHAR functions for decimal data or the CAST(*decimal-expression* AS CHAR) or CAST(*decimal-expression* AS VARCHAR) specifications.

Actions to take

These changes occur in Version 10 conversion mode (from both Version 8 and Version 9.1). You can temporarily override these changes on a subsystem level by using the BIF_COMPATIBILITY subsystem parameter. You can also temporarily override these changes on an application level by adding schema SYSCOMPAT_V9 to the front of the PATH bind option or CURRENT PATH special register. This approach works for CHAR and VARCHAR functions and does not affect CAST specifications. The recommended approach is to modify your applications to handle the Version 10 behavior for these functions, as described in the following steps.

To modify your applications to handle the Version 10 behavior for CHAR, VARCHAR, and CAST:

1. Identify applications that need to be modified to handle this change. You can use IFCID trace 0366 to identify affected applications.
2. Ensure that the BIF_COMPATIBILITY subsystem parameter is set to V9_DECIMAL_VARCHAR.

To handle the change for the CHAR function only, you can set the subsystem parameter to V9 and complete the following steps for the CHAR function.

3. Change any affected applications to handle the new Version 10 CHAR and VARCHAR behavior, including stored procedures, non-inline user-defined functions, and trigger packages. Rewrite affected CAST specifications with the appropriate CHAR or VARCHAR function and a CAST to the correct length if needed.
4. Rebind and prepare packages with the PATH(SYSCURRENT,SYSIBM) rebind option to use the new Version 10 CHAR and VARCHAR built-in functions. Repeat this step for native stored procedures (SQLPL) and non-inline SQL scalar functions.
5. For views that reference these casts or built-in functions, determine whether the view needs to be changed to have the expected output. Drop and re-create the views with the PATH(SYSCURRENT,SYSIBM) rebind option only if necessary. Rebind any applications that reference the views with the PATH(SYSCURRENT,SYSIBM) option to use the new Version 10 CHAR and VARCHAR built-in functions. Repeat this step for inline SQL scalar functions.
6. For materialized query tables or indexes on expressions that reference these casts or built-in functions, drop and re-create the materialized query tables or indexes on expressions with the PATH(SYSCURRENT,SYSIBM) rebind option. Issue the REFRESH TABLE statement for materialized query tables. Rebind any applications that reference the materialized query tables or indexes on expressions with the PATH(SYSCURRENT,SYSIBM) option to use the new Version 10 CHAR and VARCHAR built-in functions.
7. Change the value of the BIF_COMPATIBILITY subsystem parameter to CURRENT. When the subsystem parameter value is CURRENT, new applications, rebinds, and CREATE statements use the new CHAR, VARCHAR, and CAST behavior.

Materialized query tables and indexes on expressions use the CHAR, VARCHAR, and CAST behavior that is specified during its creation. If a reference statement has a different behavior that is specified by the BIF_COMPATIBILITY parameter or a different path, the materialized query table or index on expression is not used.

Related reference:

"BIF COMPATIBILITY field (BIF_COMPATIBILITY subsystem parameter)" on page 456

Change to maximum result length of VARCHAR function

In Version 10, the maximum result length of the VARCHAR function is changed from 32767 to 32704.

Leading or trailing blanks from the VARCHAR_FORMAT function format string are no longer removed (change introduced in Version 9.1)

Leading or trailing blanks from the format string for the VARCHAR_FORMAT function are no longer removed. Existing view definitions are recalculated as part of Version 10, so the new rules take effect. You can continue to use existing SQL statements that use a materialized query table that references the VARCHAR_FORMAT function, but they use the old rules and remove leading and trailing blanks. Existing references to the VARCHAR_FORMAT function in bound statements only get the new behavior when they have been bound or rebound in Version 10.

Changes to VARCHAR_FORMAT function length attribute

In Version 10, for VARCHAR_FORMAT functions, the length attribute of the result is the length attribute of the format string, up to a maximum of 255. To apply this change, use the following guidance:

- Existing view definitions that reference the VARCHAR_FORMAT function should be regenerated with an ALTER VIEW statement.
- Existing materialized query statements that reference the VARCHAR_FORMAT function should be dropped and re-created.
- Bound SQL statements that reference the VARCHAR_FORMAT function will only use the new behavior when they have been bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- DESCRIBE statements will only determine the result data type for the VARCHAR_FORMAT function using the modified rules in the DESCRIBE statements have been bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- Existing indexes that involve an expression that reference the VARCHAR_FORMAT function should be dropped and re-created.

If an application is using the DSNTIAUL program, the result string is padded with characters '00'X. Consider this incompatible change for any applications that use the VARCHAR_FORMAT function and are dependent on the output from DSNTIAUL.

New format element for VARCHAR_FORMAT function

In Version 10, a new format element, “RRRR”, is supported. In previous versions, this format element was interpreted as two adjacent specifications of the “RR” format element.

Change to result of VARCHAR_FORMAT function with “HH12” format element

In Version 10, if the “HH12” format element is specified in a VARCHAR_FORMAT function and the time component of the first argument is 24:00:00, the input timestamp value is adjusted to 00:00:00 and the date is changed to the next day. In previous releases, the timestamp value is adjusted to 12:00:00 and the day is not changed.

Change to result of VARCHAR_FORMAT function with “J” format element

In Version 10, if the “J” format element is specified in a VARCHAR_FORMAT function, the result is different from the result of the JULIAN_DAY function for dates earlier than October 15, 1582.

New supported data types for VARCHAR_FORMAT function

The VARCHAR_FORMAT function has been extended to allow date, character, and graphic string input for the first argument, and graphic string input for the second argument. If the function is invoked with one of the newly supported data types, and an existing user-defined function named VARCHAR_FORMAT also supports the data type, the function might resolve to the built-in function rather than the user-defined function. If the reference to the existing function uses the unqualified name and SYSIBM precedes the schema that was used for the user-defined function, the new function will be invoked rather than the user-defined function.

Change to maximum result length of REPEAT function

In Version 10, the maximum result length of the REPEAT function is changed from 32767 to 32704.

Change to maximum result length of XMLTABLE function

In Version 10, the maximum length of a VARCHAR data type result column of the XMLTABLE function is changed from 32767 to 32704.

DB2 drops certain indexes when a unique constraint is dropped (change introduced in Version 9.1)

In releases before Version 9.1, if a unique constraint was dropped, DB2 did not drop the index that enforced uniqueness. Starting in Version 9, if a table is in an implicitly created table space, and a unique constraint on that table is dropped, DB2 drops the index that enforces uniqueness.

DB2 enforces restriction on specifying a CAST FROM clause for some forms of CREATE FUNCTION statements (change introduced in Version 9.1)

The CAST FROM clause is included only in the syntax diagram for the CREATE FUNCTION statement for an external scalar function. The CAST FROM clause is not included in the syntax diagrams for the other variations of CREATE FUNCTION (external table function, sourced function, or SQL function); the clause cannot be used for these other variations. In previous releases, if you specified a CAST FROM clause in an unsupported context, you received no errors. Starting in Version 9, if a CAST FROM clause is specified in an unsupported context, DB2 issues an error.

DB2 enforces restrictions on specifying the AS LOCATOR clause and TABLE LIKE clause (change introduced in Version 9.1)

The AS LOCATOR clause for LOBs is included in the syntax diagram for the CREATE FUNCTION statement for an SQL function. This clause is not supported in other contexts when identifying an existing SQL function such as in an ALTER, COMMENT, DROP, GRANT, or REVOKE statement. In previous releases, if you specified an AS LOCATOR clause for LOBs in an unsupported context, you might not have received an error. Starting in Version 9, if an AS LOCATOR clause for LOBs is specified in an unsupported context, DB2 issues an error.

The TABLE LIKE clause for a trigger transition table is included only in the syntax diagram for the CREATE FUNCTION statement for an external scalar function, external table function, or sourced function. This clause is not supported for SQL functions or in other contexts when identifying an existing function such as in an ALTER, COMMENT, DROP, GRANT, or REVOKE statement, or in the SOURCE clause of a CREATE FUNCTION statement. In previous releases, if you specified a TABLE LIKE clause for a trigger transition table in an unsupported context, you might not have received an error. Starting in Version 9, if a TABLE LIKE clause for a trigger transition table is specified in an unsupported context, DB2 issues an error.

DB2 enforces restriction on the CCSID parameter for the DECRYPT_BIT and DECRYPT_BINARY functions (change introduced in Version 9.1)

The CCSID parameter is not supported by the DECRYPT_BIT and DECRYPT_BINARY built-in functions. In previous releases, if you specified an argument for the CCSID parameter for these functions, you received no errors. Starting in Version 9, if an argument is specified for the CCSID parameter in an unsupported context, DB2 issues an error.

Changed behavior of CREATE PROCEDURE for an SQL procedure (change introduced in Version 9.1)

With the introduction of native SQL procedures in Version 9, the semantics of the CREATE PROCEDURE statement for an SQL procedure has changed. Starting in Version 9, all SQL procedures that are created without the FENCED option or the EXTERNAL option in the CREATE PROCEDURE statement are native SQL procedures. In previous releases of DB2, if you did not specify either of these options, the procedures were created as external SQL procedures.

If you do specify FENCED or EXTERNAL, the meanings are the same as in previous releases of DB2. Both of these keywords mean that an external SQL procedure is to be created.

Explicitly qualify names of variables, parameters, and columns in SQL procedures (change introduced in Version 9.1)

As of Version 9, the rules that are used for name resolution within a native SQL procedure differ from the rules that were used for SQL procedures in prior releases. Because an SQL parameter or SQL variable can have the same name as a column name, you should explicitly qualify the names of any SQL parameters, SQL variables or columns that have non-unique names. For more information about how the names of these items are resolved, see References to SQL parameters and SQL variables (DB2 SQL). The rules that are used for name resolution within external SQL procedures remain unchanged.

Make any necessary program changes for possibly different values for RETURNED_SQLSTATE and DB2_RETURNED_SQLCODE (change introduced in Version 9.1)

Starting in Version 9, when an SQL statement other than GET DIAGNOSTICS or compound-statement is processed, the current diagnostics area is cleared before DB2 processes the SQL statement. Clearing of the diagnostics area can result in different values being returned for RETURNED_SQLSTATE and DB2_RETURNED_SQLCODE for a GET DIAGNOSTICS statement than what would be returned if the GET DIAGNOSTICS statement were issued from within an external SQL procedure. Additionally, there might be some differences in the values returned for the SQLSTATE and SQLCODE SQL variables than would have been returned from an external SQL procedure. (External SQL procedures were previously called SQL procedures in Version 8.)

SQLSTATE and SQLCODE SQL variables after a GET DIAGNOSTICS statement (change introduced in Version 9.1)

Starting in Version 9, the SQLSTATE and SQLCODE SQL variables are not cleared following a GET DIAGNOSTICS statement.

Coding multiple SQL statements in a handler body (change introduced in Version 9.1)

Previous releases of DB2 did not allow for a compound statement within a handler. A workaround to include multiple statements within a handler (without support for a compound statement in a handler) was to use another control statement, such as an IF statement, which in turn contained multiple statements. Starting in Version 9, DB2 supports a compound statement within a handler body. The compound statement is recommended for including multiple statements within a handler body.

Unhandled warnings (change introduced in Version 9.1)

Starting in Version 9, when a native SQL procedure completes processing with an unhandled warning, DB2 returns the unhandled warning to the calling application.

The behavior of an external SQL procedure is unchanged from releases prior to Version 9. When such a procedure completes processing with an unhandled warning, DB2 does not return the unhandled warning to the calling application.

Change your programs to handle any changed messages from SQL procedures (change introduced in Version 9.1)

Starting in Version 9, DB2 issues different messages for the new native SQL procedures than it does for external SQL procedures. (External SQL procedures were previously called SQL procedures in Version 8.) For external SQL procedures, DB2 continues to issue DSNHxxxx messages. For native SQL procedures, DB2 issues SQL return codes. The relationship between these messages is shown in the following table:

Table 31. Relationship between DSNHxxxx messages that are issued for external SQL procedures and SQLCODEs that are issued for native SQL procedures

DSNHxxxx message[1]	SQLCODE[2]
DSNH051I	-051
DSNH385I	+385
DSNH590I	-590
DSNH4408I	-408
DSNH4777I	n/a
DSNH4778I	-778
DSNH4779I	-779
DSNH4780I	-780
DSNH4781I	-781
DSNH4782I	-782
DSNH4785I	-785
DSNH4787I	-787

Note:

1. These messages are used for external SQL procedures, which can be defined by specifying EXTERNAL or FENCED in Version 10.
2. These messages are used for native SQL procedures in Version 10.

Enhanced data type checking for zero-length characters (change introduced in Version 9.1)

Starting in Version 9, when you specify a CHAR data type with a length of 0 in the SQLDA, DB2 issues SQLCODE -804 regardless of the null indicator value.

Adding a column generates a new table space version (change introduced in Version 9.1)

In previous releases, adding a column to a table did not generate a new table space version. Starting in Version 9, adding a column to a table with an ALTER TABLE ADD COLUMN statement generates a new table space version.

You cannot add a column and issue SELECT, INSERT, UPDATE, or DELETE statements in the same commit scope

You cannot have a version-generating ALTER TABLE ADD COLUMN statement and SELECT, INSERT, UPDATE, or DELETE statements in the same commit scope. If a version-generating ALTER TABLE ADD COLUMN statement follows SELECT, INSERT, UPDATE, or DELETE statements in the same commit scope, SQLCODE

-910 is issued. SQLCODE -910 is also issued if SELECT, INSERT, UPDATE, or DELETE statements follow a version-generating ALTER TABLE ADD COLUMN statement in the same commit scope.

CAST FROM clause of CREATE FUNCTION statement for SQL functions is no longer supported (change introduced in Version 9.1)

The CAST FROM clause of the CREATE FUNCTION statement for SQL functions is no longer supported. Starting in Version 9, if you issue a CREATE FUNCTION statement for an SQL function with a CAST FROM clause, DB2 issues an error.

Specifying ALTER DATABASE STOGROUP for work file databases (change introduced in Version 9.1)

In previous releases of DB2, you could not execute ALTER DATABASE STOGROUP on a work file database. Beginning with DB2 Version 9.1 conversion mode, this restriction is removed.

DB2 enforces restrictions about where an INTO clause can be specified (change introduced in Version 9.1)

The INTO clause (as related to queries) is included only in the syntax diagram for the SELECT INTO statement. The INTO clause is not included in the syntax diagrams for select-clause, subselect, fullselect, or select-statement. In previous releases, if you specified an INTO clause in an unsupported context in a query, you might not have received an error. Starting in Version 9, if an INTO clause is specified in an unsupported context, DB2 issues an error.

Change to how a positive, signed integer in an ORDER BY clause is treated

Beginning in Version 10 conversion mode (from both Version 8 and Version 9.1), a positive, signed integer in an ORDER BY clause is treated as a *sort-key-expression*. This change might affect the result of queries that contain an ORDER BY clause with a positive, signed integer. Such integers were previously interpreted as column numbers.

Binding DBRMs directly into plans is no longer supported

For pre-existing plans that are bound from DBRMs, you can use the COLLID parameter of the REBIND PLAN command to create packages. If you execute a plan that is bound from DBRMs, DB2 performs an automatic rebind that creates packages from the DBRMs and binds those packages into a plan. However, the recommendation is to use REBIND with the COLLID option so that you can specify bind options and receive more diagnostic information. If the installation uses the RACF access control module, owners of plans with DBRMs need to explicitly rebind the plans to convert the DBRMs to packages.

Some BIND PLAN and REBIND PLAN command options are no longer supported

The ACQUIRE(ALLOCATE) option of the BIND PLAN and REBIND PLAN commands is no longer supported. If you specify ACQUIRE(ALLOCATE), DB2 issues a warning message and uses ACQUIRE(USE).

Also, the MEMBER option of BIND PLAN and REBIND PLAN is no longer supported. If you specify MEMBER, DB2 issues a warning message, binds the specified DBRM into a package, and binds the package into a plan.

Plans and packages should be converted to DRDA protocol

Plans and packages that were previously bound using DBPROTOCOL(PRIVATE) should be converted to DRDA® protocol before migration to Version 10. In Version

10, plans and packages that were bound with the DBPROTOCOL(PRIVATE) bind option and access remote locations cannot run. Applications that use packages or plans that were bound with DBPROTOCOL(PRIVATE) and access remote locations fail with SQLCODE -904. A rebind of those plans and packages must be explicitly performed before they can execute successfully. Job DSNTIJPM identifies the objects that must be converted to use DRDA protocol.

Change to GRANT statement

The PUBLIC AT ALL LOCATIONS clause is no longer allowed in the GRANT statement for table and view privileges as an alternative to PUBLIC. The DSNTPPCK program in Version 8 and Version 9.1 analyzes the embedded SQL statements in packages and plans for private protocol SQL, which is invalid in Version 10. The program produces a report that states which packages and member DBRMs of plans contain the invalid syntax. The program scans either the SYSIBM.SYSSTMT catalog table, the SYSIBM.SYSPACKSTMT catalog table, or both. Applications that issue dynamic SQL statements with the invalid PUBLIC AT ALL LOCATIONS clause will receive SQLCODE -199.

Change to IMMEDWRITE option of BIND PACKAGE command

In DB2 Version 10, if IMMEDWRITE is not specified on a BIND PACKAGE command, the default is “I”, or INHERITFROMPLAN. In previous versions, the default was NO.

Changes to conversion of special characters in collection IDs and package names

In Version 10, DRDA character type parameter data is sent between client drivers and DB2 in UTF-8 Unicode, if those client drivers also have this support. Character type parameter data includes package names and collection IDs. Prior to Version 10, collection IDs and package names were sent in EBCDIC, and then converted to Unicode before being stored in the DB2 catalog. A collection ID or package name that is in the DB2 catalog from a bind that was initiated by an older driver might not match an ID or name that is sent to DB2 by a newer driver. This mismatch, which is caused by the way that some special characters are converted, can cause package-not-found errors. Job step DSNTGEN of job DSNTIJPM identifies package names and collection IDs that contain special characters that cause mismatches. From the upgraded client drivers, those packages need to be bound with the ACTION(REPLACE) option.

Changes to the RELEASE bind option

In releases prior to Version 10, the RELEASE bind option had no effect on database access threads. Starting in Version 10, by default, DB2 honors the RELEASE bind option for database access threads. You can modify this behavior by using the new MODIFY DDF PKGREL command.

Database metadata stored procedures are converted to Unicode

In DB2 Version 10, DB2-supplied database metadata stored procedures are encoded in Unicode. The Version 10 migration process redefines the stored procedures to use the following new load modules:

Table 32. Metadata stored procedures and associated load modules

Stored procedure	Load module
SYSIBM.SQLCOLPRIVILEGES	DSNACPRU
SYSIBM.SQLCOLUMNS	DSNACOLU
SYSIBM.SQLFOREIGNKEYS	DSNAFNKU
SYSIBM.SQLFUNCTIONCOLS	DSNAFCOU

Table 32. Metadata stored procedures and associated load modules (continued)

Stored procedure	Load module
SYSIBM.SQLFUNCTIONS	DSNAFUNU
SYSIBM.SQLGETTYPEINFO	DSNATYPU
SYSIBM.SQLPRIMARYKEYS	DSNAPRKU
SYSIBM.SQLPROCEDURECOLS	DSNAPCOU
SYSIBM.SQLPROCEDURES	DSNAPRCU
SYSIBM.SQLSPECIALCOLUMNS	DSNASPCU
SYSIBM.SQLSTATISTICS	DSNASTAU
SYSIBM.SQLTABLEPRIVILEGES	DSNATBPU
SYSIBM.SQLTABLES	DSNATBLU
SYSIBM.SQLUDTS	DSNAUDTU

Some applications call the database metadata stored procedures to retrieve double-byte (DBCS) data. Those applications must be modified to use Unicode if they are bound on a DB2 server that has an EBCDIC SBCS CCSID, and the MIXED parameter is set to NO in the application defaults load module, *dsnhdecp*.

AUTHID is the default owner of packages that are bound by DSNTRIN

In previous releases, the stored procedures and user-defined functions that are provided as part of the DB2 base product (DB2-supplied routines) were created and bound by processing inline DDL and bind statements in DSNTIJSG and other installation jobs. By default, the ID that was used to run the job was also the authorization ID for creating the routines and the default package owner for those routines.

In Version 10, DB2-supplied routines are created and bound by running program DSNTRIN in job DSNTIJRT. The AUTHID parameter of DSNTRIN specifies the authorization ID for creating the routines. This ID is also the default owner of packages that are bound for those routines.

New default DEFINE attribute for dependent objects

As of Version 10, if the DEFINE attribute is not specified in the CREATE statement for explicitly created dependent objects (auxiliary indexes, XML indexes, and base table indexes), DB2 uses the DEFINE attribute of the base table space.

Exception: If the DEFINE attribute is not specified for explicitly created LOB table spaces, there is no correlation with the base table space until the auxiliary table is created. The DEFINE attribute is not inherited from the base, and the default is DEFINE YES.
Implicitly created dependent objects (base table indexes, LOB and XML table spaces, and their dependent indexes) inherit the DEFINE attribute of the base table space if it is DEFINE NO. Otherwise, if the base table space attribute is DEFINE YES, the value of the IMPDSDEF subsystem parameter is used for the dependent objects.

Change for creating partitioned table spaces

To create a classic partitioned table space in DB2 Version 10, you must specify SEGSIZE 0 and the NUMPARTS keyword of the CREATE TABLESPACE statement.

Change to default for CREATE TABLESPACE statements

In previous releases of DB2, if a CREATE TABLESPACE statement contains the NUMPARTS clause but neither the MAXPARTITIONS clause nor the SEGSIZE clause, a classic partitioned table space is created. Beginning in Version 10, the same statement results in a range-partitioned table space with a segment size of 32 by default. You might observe this difference when a subsequent CREATE INDEX statement with the specified ranges fails because index partitioning of a range-partitioned table space is not supported.

In response, you can change the default segment size through the DSN6SYSP.DPSEGSZ subsystem parameter, which is externalized as the DEFAULT PARTITION SEGSIZE field on panel DSNTIP7. The value of the DPSEGSZ parameter can range from 0 to 64 in increments of 4; for example, 0, 4, 8, and so on up to 64. The default is 32.

When DPSEGSZ is set to 0, a CREATE TABLESPACE statement that contains the NUMPARTS clause but neither the MAXPARTITIONS clause nor the SEGSIZE clause results in a classic partitioned table space, which is the behavior of previous releases. Note that the DPSEGSZ parameter is provided for compatibility only. It is deprecated in Version 10, and you should take steps to modify affected CREATE TABLESPACE statements or plan to use partition-by-growth table spaces instead of classic partitioned table spaces.

Upgrade to supported COBOL and PL/I compilers

If you use the Version 10 precompiler, you must upgrade to COBOL compilers that DB2 10 for z/OS supports. See the DB2 Program Directory for information about supported compilers. The generated SQLCA and SQLDA for COBOL have changed. The generated attribute for binary data items is now COMP-5 instead of COMP-4 or COMP. Applications that are compiled on compilers that do not support the COMP-5 attribute no longer work.

For some COBOL and PL/I compilers that are no longer supported, you can use a version of the precompiler that allows you to precompile applications that have dependencies on these unsupported compilers. You can use this version of the precompiler with the following unsupported compilers:

- OS/VS COBOL V1.2.4
- OS PL/I 1.5 (PL/I Opt. V1.5.1)
- VS/COBOL II V1R4
- OS PL/I 2.3

The load module for this precompiler is DSNHPC7. This precompiler is meant only to ease the transition from unsupported compilers to supported compilers. This precompiler has the following restrictions:

- There is no corresponding DB2 coprocessor function to match this precompiler.
- The precompiler does not support SQL procedures.
- Only COBOL and PL/I are supported.
- The SQL flagger is not supported.
- The precompiler produces Version 7 DBRMs, and does not support any capability that is newer than Version 7.
- The application defaults module must be named DSNHDECP.

Support of this precompiler is deprecated in Version 10.

GRAPHIC and NOGRAPHIC SQL processing options are removed

If you specify the SQL processing options GRAPHIC or NOGRAPHIC, DB2 issues a standard warning message about an invalid option. These options are superseded by the CCSID SQL processing option.

SELECT FROM data change statements in BEFORE triggers no longer supported

The following statements are no longer allowed in the body of a BEFORE trigger:

- SELECT FROM DELETE
- SELECT FROM INSERT
- SELECT FROM MERGE
- SELECT FROM UPDATE

RETURN statement in scalar functions must follow *option-list*

In previous DB2 versions, RETURN statements in scalar functions could be in any order, relative to other clauses. After migration to Version 10 conversion mode (from Version 8 or Version 9.1), the RETURN statement in CREATE FUNCTION or ALTER FUNCTION statements must follow *option-list*. If a RETURN statement precedes *option-list* in one of these statements, DB2 issues SQLCODE -199.

Changes to ROUND_TIMESTAMP and TRUNC_TIMESTAMP functions

In Version 10, the ROUND_TIMESTAMP and TRUNC_TIMESTAMP functions return the first day of the first ISO week of the ISO year when an ISO year format is specified.

Also, whenever a CC or SCC format is specified, a start of a century is considered to be year 01.

Changes to result of NEXT_DAY function

In Version 10, the result data type for the NEXT_DAY function is determined from the input data. If the first input argument is a string, the result is TIMESTAMP(6) WITHOUT TIME ZONE. Otherwise, the data type of the result is the same as the data type of the first input argument. For example, if the input is a date data type, the result is also a date data type. The following rules apply to the change in the NEXT_DAY function:

- Existing view definitions that reference the NEXT_DAY function should be regenerated with an ALTER VIEW statement.
- Existing materialized query statements that reference the NEXT_DAY function should be dropped and re-created.
- Bound SQL statements that reference the NEXT_DAY function only use the modified rules to determine the result data type if the statements are bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- DESCRIBE statements only use the modified rules to determine the result data type for the NEXT_DAY function if the statements are bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- Existing indexes that involve an expression that reference the NEXT_DAY function should be dropped and re-created.

Changes to MONTHS_BETWEEN function

In previous releases, the MONTHS_BETWEEN function ignored the time portion of arguments. In Version 10, the MONTHS_BETWEEN function takes the time portion of arguments into consideration.

Changes to TIMESTAMPDIFF function

In previous releases, the TIMESTAMPDIFF function allowed string input values that had more than six digits to the right of the decimal point. In Version 10, an error is issued if the string input value for TIMESTAMPDIFF function has more than six digits to the right of the decimal point.

Static SQL applications that use parallelism

DB2 incrementally rebinds the statements that use parallelism after migration to Version 10. Incremental rebinds can cause performance degradation, so you should manually rebind those statements. You can run a query in job DSNTIJPM before you migrate to determine which statements can use parallelism, and are therefore candidates for incremental rebinds. You should consider rebinding those statements after migration, as soon as your Version 10 system is stable. After you migrate to Version 10, you can also run a performance trace, class 3 or class 10 for IFCID 360, to identify the plans and packages that contain static SQL queries that use parallelism, and therefore need to be rebound.

Enforced SELECT authorization checking for UPDATE and DELETE statements

DB2 Version 10 checks for the SELECT privilege or appropriate administrative privilege before allowing a user to execute UPDATE or DELETE statements that reference an existing value in the target table. This authorization checking applies regardless of how the statements are executed (for example, bound in a package or executed in a dynamic statement). The authorization now also applies regardless of the setting of the SQLRULES(STD) bind option for static statements or the CURRENT RULES special register for dynamic statements. If the user does not have the necessary SELECT authorization or administrative privilege, a negative SQLCODE is returned.

Increased limit for work file record length

In Version 10 new-function mode, the limit for the row length in the result of a JOIN or the row length of a SORT record is increased from 32 767 bytes (1 page) to 65 529 bytes. The sort key maximum length is also increased from 16 000 bytes to 32 000 bytes. Applications that exceed the old limits fail in conversion mode (from both Version 8 and Version 9.1) with SQLCODE -670 or SQLCODE -136.

New restrictions for EXPLAIN tables

In DB2 Version 10 conversion mode (from both Version 8 and Version 9.1), EXPLAIN tables must be in Version 8 or later format and preferably encoded in Unicode. When EXPLAIN tables are in a format prior to the Version 8 format, DB2 returns SQLCODE -20008 reason code 2 for statements or commands that invoke EXPLAIN processing. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9.1 format, regardless of the encoding type. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned. When you convert EXPLAIN tables to Unicode encoding, applications that join with EXPLAIN tables might have different results because of the CCSID conversion. For more information, see Objects with different CCSIDs in the same SQL statement (DB2 Internationalization Guide).

Recommendation: Before you begin migration to Version 10, convert all EXPLAIN tables to the current version (8 or 9.1) format and Unicode encoding, and then check for joins to those tables. You can use job DSNTIJXA to convert most tables to current release format. You can use jobs DSNTIJXB and DSNTIJXC to migrate EBCDIC-encoded EXPLAIN tables to Unicode.

MEMBER CLUSTER table spaces indicated by MEMBER_CLUSTER column

In previous versions of DB2, a “K” or “I” in the TYPE column of the SYSTABLESPACE catalog table indicated that the table space had MEMBER CLUSTER structure. In Version 10, a new MEMBER_CLUSTER column on the SYSTABLESPACE catalog table is populated during the enabling-new-function mode (from both Version 8 and Version 9.1) migration process. For existing MEMBER CLUSTER table spaces, values of “K” in the TYPE column of SYSTABLESPACE are replaced with “L”, and values of “I” are replaced with blank. The MEMBER_CLUSTER column is populated with “Y”. After migration to enabling-new-function mode (from Version 8 or Version 9.1), applications that query “K” or “I” in the TYPE column must query the new MEMBER_CLUSTER column instead.

Changed values for the modification level in the product signature

The DB2 Version 10 product signature has the form DSN1001*m*, where *m* is the modification level. Values 0 and 1 are reserved for maintenance levels in conversion mode from Version 8, conversion mode* from Version 8, enabling-new-function mode from Version 8, and enabling-new-function mode* from Version 8. Values 2 and 3 are for maintenance levels in conversion mode from Version 9.1, conversion mode* from Version 9.1, enabling-new-function mode from Version 9.1, and enabling-new-function mode* from Version 9.1. Values 5, 6, 7, 8, and 9 are for maintenance levels in new-function mode. Value 4 is undefined.

Changed behavior for the CREATE FUNCTION statement

In all forms of the CREATE FUNCTION statement, a parameter list is required. Functions without parameters must have empty parentheses specified, as in the following example: `CREATE FUNCTION F1 ( ) RETURNS INT RETURN 1`.

In previous releases, if you specified CREATE FUNCTION without a parameter list (for example, `CREATE FUNCTION F1 RETURNS INT RETURN 1`), you received no errors. In Version 10, if you specify CREATE FUNCTION without a parameter list, DB2 issues an error.

Also, a CREATE FUNCTION statement for a non-inline SQL scalar function cannot have a parameter that is a distinct type that is based on a LOB data type. The inline SQL scalar functions have supported, and will continue to support, distinct type parameters where the underlying base data type is a LOB. For the syntax of this statement, see CREATE FUNCTION in the DB2 for z/OS SQL Reference Guide.

Different SQLSTATE returned for some DELETE or UPDATE statements

In previous releases of DB2, a SQLSTATE value of "00000" was returned for some DELETE or UPDATE statements that did not contain a WHERE clause, when SQLWARN flags were set. Those situations occurred during bind processing. As of DB2 Version 10, a SQLSTATE value of "01504" is returned in those situations. This change does not affect SQL statements that are executed on behalf of a DB2 for z/OS requester.

Changed default behavior of multiple-row inserts for ODBC z/OS applications

As of DB2 Version 10, the default behavior for multiple-row inserts is changed from non-atomic to atomic. To change the default behavior back to non-atomic, set keyword PARAMOPTATOMIC = 0 in the data source section of the ODBC initialization file.

Changes to ALTER TABLESPACE statement error codes

Before Version 10 conversion mode (from Version 8 or Version 9.1), if you execute an ALTER TABLESPACE MAXPARTITIONS statement on catalog objects, DB2 issues SQLCODE -644. After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), DB2 issues SQLCODE -607.

Also before Version 10 conversion mode (from Version 8 or Version 9.1), if you execute an ALTER TABLESPACE MAXPARTITIONS statement with a value of MAXPARTITIONS that is invalid with the page size and DSSIZE values, DB2 issues SQLCODE -4701. After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), DB2 issues SQLCODE -644.

Change to CREATE and ALTER statements

CREATE and ALTER statements for native SQL procedures no longer support the DEFAULT keyword for the SQL PATH option.

Change to DESCRIBE statement

In Version 10, when a DESCRIBE statement is used on a result table that includes a distinct type, it will now return information about the distinct type.

New restrictions on using DSNTIAUL

DSNTIAUL can no longer be used to process CREATE FUNCTION (SQL scalar) statements that would result in a package or CREATE TRIGGER statements. DSNTIAUL also cannot be used to process any other statement that contains *SQL-routine-body*. These statements are CREATE PROCEDURE (SQL external), CREATE PROCEDURE (SQL native), CREATE FUNCTION (SQL table), ALTER PROCEDURE (SQL native) with an ADD or REPLACE clause, and ALTER FUNCTION (SQL scalar) with an ADD or REPLACE clause.

Changes to SYSROUTINES

As of Version 10, the CREATEDTS column of SYSIBM.SYSROUTINES will always reflect the time that a CREATE statement was first issued for a routine. In previous releases of DB2, the CREATEDTS column might have contained different values if multiple versions of a routine were created.

Catalog restructured

In Version 10, the DB2 catalog is restructured to reduce lock contention. As a result, binding takes longer in Version 10 than it did in previous versions. Also, execution of the DECLARE GLOBAL TEMPORARY TABLE statement takes longer in Version 10.

Changed data type for an untyped parameter marker

In previous releases, an untyped parameter marker for a unary minus operator has an assumed DOUBLE data type. In DB2 Version 10, the assumed data type is DECFLOAT(34).

Changes to handling of special values Infinity, sNaN, and NaN

In previous releases, when DB2 returns a decimal floating-point (DECFLOAT) value for Infinity, NaN, or sNaN as a CHAR or VARCHAR string, the string is in mixed case. In DB2 Version 10, the values are returned in upper case as follows: INFINITY, NAN, or SNAN.

Changes for INSTEAD OF triggers

Changes are introduced for existing INSTEAD OF triggers that were defined on a view that has a ROWID column or a column that is based on an underlying column of any of the following types:

- A security label column.
- A row change timestamp column.
- A column that is defined with AS ROW BEGIN.
- A column that is defined with AS ROW END.
- A column that is defined with AS TRANSACTION START ID.

For such INSTEAD OF triggers, the following actions now fail with a negative SQLCODE:

- Rebinding the INSTEAD OF trigger package.
- Dropping and re-creating the INSTEAD OF trigger.
- Running the REPAIR DBD utility on a database that includes the INSTEAD OF trigger.
- Issuing ALTER TABLE ALTER COLUMN SET DATA type on a table that has a view on which the INSTEAD OF trigger is defined.

Change to positioned update or delete statements

Explanation

In DB2 Version 10, packages must be rebound if both of the following statements are true:

- The package contains static cursors that were bound in a previous version.
- The package contains dynamic UPDATE or DELETE statements that contain the WHERE CURRENT of clause. The WHERE CURRENT of clause indicates a positioned UPDATE or DELETE statement.

Possible impact to your DB2 environment

If any such packages are not rebound in Version 10, positioned UPDATE and DELETE statements fail with SQLCODE -20249 when they run against a down-level cursor.

Actions to take

After migration to Version 10 conversion mode (from Version 8 or Version 9.1), rebind packages that contain static cursors with positioned UPDATE or DELETE statements.

Change to stored procedure parameter values returned to non-Java clients

In previous releases, when a remote application calls a DB2 for z/OS stored procedure, the data types of the returned output data match the data types of the corresponding CALL statement arguments. Starting in Version 10 conversion mode (from both Version 8 and Version 9.1), the data types of the returned output data

match the data types of the parameters in the stored procedure definition. This change can cause conversion failures for some applications that use non-Java client drivers such as .NET to call stored procedures on DB2. To prepare for this change, modify the CALL requests of your non-Java client applications to specify argument types that conform to the semantics of the client driver. If you want to temporarily override the Version 10 behavior, you can set the DDF_COMPATIBILITY subsystem parameter to SP_PARMS_NJV.

The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in Version 10, it will be removed in a later release of DB2.

Change to results of JDBC method PreparedStatement.setTimestamp

TIMESTAMP WITH TIME ZONE is first supported in Version 10 new-function mode. Under certain circumstances, an application that invokes `PreparedStatement.setTimestamp` might produce different results before and after conversion to Version 10 new-function mode. If the second parameter of `PreparedStatement.setTimestamp` has the `java.sql.Timestamp` data type, and the target data type is not known, the IBM Data Server Driver for JDBC and SQLJ chooses TIMESTAMP as the target data type before Version 10 new-function mode. However, starting with Version 10 new-function mode, unless the input value is 0001-01-01-00:00:00.000000 or 9999-12-31-23:59:59.999999, the driver chooses TIMESTAMP WITH TIME ZONE as the target data type. If the driver chooses the TIMESTAMP data type, and the target type is actually TIMESTAMP WITH TIME ZONE, the database manager sets the time zone in the target column using the value of the IMPLICIT_TIMEZONE DECP value. This value might differ from the value that is inserted prior to Version 10 new-function mode.

To produce the same results when `PreparedStatement.setTimestamp` is executed, specify a `com.ibm.db2.jcc.DBTimestamp` value as the second parameter.

Change in how DB2 returns stored procedure output parameter data to remote clients

When an application on a client system calls a stored procedure on a DB2 10 for z/OS server, DB2 now handles the output parameters differently. Previously, DB2 returned stored procedure output parameters that were formatted according to the SQL type of the corresponding argument in the CALL statement. DB2 10 now returns output parameters that are formatted according to the SQL type of the corresponding parameter in the stored procedure declaration. This new behavior provides improved performance at the server by avoiding unnecessary server data conversions. Also, this new behavior is consistent with the existing server behavior for the return of query and select output data and it is consistent with the behavior of other DB2 family servers. Therefore, it provides applications with a more consistent and predictable interface to DB2.

In general, for applications that conform to client standards, this change has no impact to the calling application. In some cases, however, application changes might be needed. To prepare for this change, examine your remote applications that call DB2 stored procedures. If necessary, modify the CALL statements in your remote applications to specify argument data types that match the data types of the parameters in the stored procedure definitions.

When the change in behavior occurs

In general, the new behavior occurs after migration to DB2 10 for z/OS conversion mode from Version 8 (CM8) or conversion mode from Version 9.1 (CM9). However, the following exceptions apply:

- For applications using an IBM Data Server Driver for JDBC and SQLJ type 4 connectivity to access a DB2 10 data sharing group in CM8 or CM9, where the enableSysplexWLB client property is set to true:

  In such a configuration, the change in behavior occurs when the data sharing group is migrated to DB2 10 new-function mode (NFM). As long as the data sharing group is in CM8 or CM9 to support coexistence, DB2 uses the old behavior. The old behavior is used because the Java client driver might be caching data descriptors for outputs from the server. When the enableSysplexWLB property is set to true, the old behavior is maintained to ensure that the cached client descriptors are valid regardless of which member of the data sharing group is accessed.

  To prepare for this change in behavior while DB2 10 is in CM8 or CM9 to support coexistence, test your applications with the enableSysplexWLB client property set to false. Or, test the applications against a stand-alone DB2 10 server, if such a system is available for testing. The new DB2 10 behavior will be used for each of those applications even though the DB2 data sharing group is in CM8 or CM9.

- For applications using IBM Data Server Driver for non-Java clients (such as .NET), Version 9 or earlier:

  In such a configuration, the change in behavior occurs, by default, after migration to DB2 10 for z/OS CM8 or CM9. However, the change in behavior can be temporarily deferred. To temporarily override the DB2 10 behavior for returning stored procedure output parameter data to IBM Data Server non-Java clients only (such as .NET), set the DDF_COMPATIBILITY subsystem parameter to SP_PARMS_NJV. The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in DB2 10, it will be removed in a later release of DB2. For more information about DDF_COMPATIBILITY, see Subsystem parameters that are not on installation panels (DB2 Installation and Migration).

Examples of application impacts

The following examples use IBM Data Server Java and .NET client drivers. Other drivers might behave differently. Because DB2 10 no longer converts data types when returning stored procedure output parameter data, there might be similar impacts to applications for other data types that are not discussed in the examples. If the calling application specifies arguments that are different from the declared parameter types, then the mismatch between the argument and parameter data types is handled by the client system, according to the client's programming model and the client's capabilities. If the client performs data type conversions such as for numeric data, the result of the conversion performed by the client might differ from the result that was previously returned when the DB2 server performed the conversion.

Example 1: A stored procedure parameter is declared as an INTEGER data type, but the application CALL statement specifies a SMALLINT argument.

- Behavior before DB2 10: DB2 converts the INTEGER data and returns a 2-byte SMALLINT value, which maps to an Int16 .NET data type.

- New behavior in DB2 10: DB2 returns a 4-byte INTEGER value, which maps to an Int32 .NET data type.

Sample application impact (.NET): If the application uses an IBM Data Server .NET client, the stored procedure CALL statement now fails with an invalid conversion error because the Int16 and Int32 .NET data types are not compatible with each other. The application must be changed to specify an Int32 argument in the CALL statement as required by the .NET programming model. Applications that are coded with an Int32 argument and call stored procedures that have an INTEGER parameter require no change.

The .NET programming model enforces strong data typing. Thus, even though the argument type and the parameter type are compatible SQL types, the .NET driver enforces strong data type checking according to the .NET semantics. Refer to the .NET documentation for information about .NET strong type checking.

Example 2: A stored procedure parameter is declared as a REAL data type, but the application CALL statement specifies a DOUBLE argument.

- Behavior before DB2 10: DB2 converts the output data to a DOUBLE value before returning it to the client.
- New behavior in DB2 10: DB2 returns the data to the client as a REAL value.

Sample application impact (Java): If an application that uses an IBM Data Server Java client uses the registerOutParameter() method to register the output parameter as a DOUBLE value before calling a stored procedure that has a REAL parameter, the following behavior occurs if the Java application uses the getObject() method to retrieve the output value:

- Before Version DB2 10, DB2 converted the REAL value to a DOUBLE value and returned it to the client. The Java driver returned the DOUBLE value directly to the application as the result of the getObject() method.
- Beginning in DB2 10, DB2 returns the REAL value to the client. The Java driver converts the REAL value to a DOUBLE value and returns it to application as the result of the getObject() method. APAR IC80974 for LUW clients and APAR PM58951 (JCC 3.63.131) and PM58952 (JCC 4.13.136) for z/OS clients are required in the IBM Data Server Java driver to ensure that the conversion is completed by the getObject() method. If the APAR fix is not applied, the Java application gets a ClassCastException.

Example 3: A stored procedure parameter is declared as a TIMESTAMP data type, but the application CALL statement specifies a VARCHAR argument.

- Behavior before DB2 10: DB2 converts the fixed-length TIMESTAMP data and returns a 26-byte VARCHAR value, containing a TIMESTAMP value in ISO format (*yyyy-mm-dd-hh.mm.ss*[*.ffffffff*]). The VARCHAR value maps to a String .NET data type.
- New behavior in DB2 10: DB2 returns a 26-byte fixed-length TIMESTAMP value, containing a TIMESTAMP value in ISO format. The TIMESTAMP value maps to a DateTime .NET data type.

Sample application impact (Java): If the application uses an IBM Data Server Java client, then an application change might be required, depending on what method the Java application uses to retrieve the parameter data.

- Before DB2 10, the getString() method previously returned TIMESTAMP data in ISO format, while the getTimestamp() method returned TIMESTAMP data in Java format

- Beginning in DB2 10, both the getString() and getTimestamp() methods return the TIMESTAMP value in Java format (*yyyy-mm-dd hh*:*mm*:*ss*[.*fffffffff*]).

An application change might be required if the Java application uses the getString() method and the application depends on receiving TIMESTAMP values in ISO format. Applications that use the getTimestamp() method require no change.

Similar considerations apply for TIME parameters. Beginning in DB2 10, DB2 returns TIME parameters as TIME data values in ISO format (*hh.mm.ss*), regardless of the SQL type of the corresponding argument in the CALL statement. For Java clients, the getString() method now returns TIME data in Java format (*hh.mm.ss*). If a Java application uses the getString() method and requires TIME data in ISO format, the application must be examined for possible changes.

Changes to datetime built-in functions

Explanation

Many datetime functions allow arguments containing *string representations of datetime values*. Valid formats for those strings are described in the DB2 SQL Reference.

In DB2 Version 10 conversion mode (from Version 8 or Version 9.1), the following additional string formats are allowed in the specified limited contexts:

- A string value of seven characters representing a date is allowed as an argument to the DATE function only.
- A string value of 8, 13, or 14 characters representing a point in time is allowed as an argument to the TIMESTAMP function only.

Possible impact to your DB2 environment

After migration to Version 10, applications that provide a seven-character string argument to represent a date for built-in functions other than the DATE function will return an error. Applications that provide a string value of 8, 13, or 14 characters to represent a point in time as an argument for built-in functions other than the TIMESTAMP function also return an error.

Actions to take

To prepare for this change, use valid string formats that are described in String representations of datetime values (DB2 SQL) in arguments for all of your datetime functions.

Qualify user-defined function names

If you use a user-defined function that has the same name as a built-in function that has been added to Version 10, ensure that you fully qualify the function name. If the function name is unqualified and “SYSIBM” precedes the schema that you used for this function in the SQL path, DB2 invokes one of the built-in functions.

For a list of built-in functions, including those that have been added in Version 10, see Functions (DB2 SQL).

SQLCODE changes

Some SQLCODE numbers and message text might have changed in DB2 Version 10. Also, the conditions under which some SQLCODEs are issued might have changed.

SQL reserved words

Version 10 has several new SQL reserved words. Refer to Reserved words (DB2 SQL) for the list of reserved words, and adjust your applications accordingly.

GUPI

Utility release incompatibilities for migration from Version 8

When you migrate from DB2 Version 8 to Version 10, be aware of the utility release incompatibilities.

Plan for the following changes in Version 9.1 and Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changes to the REORG SHRLEVEL CHANGE utility (change introduced in Version 9.1)

If you use the REORG SHRLEVEL CHANGE utility, you might need to change your operating procedures. If you ran concurrent REORG TABLESPACE SHRLEVEL CHANGE PART x on the same table space in your previous DB2 release, you need to change the way you submit the jobs so that only one executes at a time. If you allocate your own shadow data sets (for example, if you use user-managed VSAM data sets), you need to change your jobs to use the NPI data set naming convention and you need to allocate larger data sets for the NPI shadows. Also, the BUILD2 phase of REORG TABLESPACE PART x SHRLEVEL CHANGE is eliminated.

If you have not been using RETRY with REORG SHRLEVEL CHANGE or REFERENCE, in Version 10, your REORGs retry after failed drain attempts, and if the retries are not successful, the jobs terminate with return code 8 instead of abending.

During a REORG TABLESPACE PART SHRLEVEL CHANGE, applications that access partitions other than the ones being reorganized might now encounter timeouts if they access NPI data sets during the last iteration of the LOG phase or during the SWITCH phase. Adjust the values of the DRAIN_WAIT and MAXRO parameters to minimize application timeouts.

Changes to the REORG utility (change introduced in Version 9.1)

The REORG utility has been updated to reduce the amount of virtual storage that is used to build dictionaries. Because the REORG utility needs less virtual storage, the 254 partition restriction for reorganizing compressed table spaces has been removed. REORG no longer issues message DSNU1146I.

Changes to DSN1LOGP (change introduced in Version 9.1)

In previous releases, when you invoked DSN1LOGP with an RBA or LRSN range and the lower or higher bounds are not found in the available log files, DSN1LOGP returned return code 0. In Version 10, when you invoke DSN1LOGP with an RBA or LRSN range and the lower or higher bounds are not found in the available log files, DB2 issues a warning message to let you know that not all of the expected information can be formatted in the available logs and returns return code 4.

Changes to INTEGER EXTERNAL option (change introduced in Version 9.1)

In DB2 Version 8, the INTEGER EXTERNAL option of the UNLOAD utility had a default value of 11 bytes. As of DB2 Version 9.1, the INTEGER EXTERNAL option has a default value of 20 bytes. To force a length of 11 bytes, you can specify INTEGER EXTERNAL(11).

Changes to RECOVER utility

For point-in-time recovery, the RECOVER utility now enforces the recovery of a base table space together with its associated auxiliary LOB or XML table spaces.

Changes to COPY CHANGELIMIT

The default value of the COPY CHANGELIMIT option has changed from 1,10 to 10. This new default value means that COPY creates image copies as follows:

- If the percentage of changed pages is greater than 0 and less than 10, COPY creates an incremental image copy.
- If the percentage of changed pages is greater than or equal to 10, COPY creates a full image copy.
- If no pages have changed, no copies are made.

If this behavior is not the outcome that you want for any COPY statements with the CHANGELIMIT default option, change those statements to specify appropriate CHANGELIMIT values.

Changes to RUNSTATS INDEX

In DB2 Version 10, the following changes are made to the RUNSTATS INDEX utility:

- The default SHRLEVEL value is changed from REFERENCE to CHANGE.
- The KEYCARD behavior is always in effect, and the KEYCARD keyword is deprecated. If you specify KEYCARD in the utility job statement, no error message is issued to indicate that the keyword is deprecated.

Changes to DSN1PRNT

In DB2 Version 10, when you run DSN1PRNT with the FORMAT EXPAND option, and the input data sets constitute a full image copy, the input data sets must contain all pages of the original table space, including dictionary pages.

Changes to DSNJU004

The output of the DSNJU004 (print log map) utility has changed. The output of the system-level backup BSDS record now includes the z/OS level of the backup and the catalog copy option. This information is collected by the BACKUP system utility in DB2 Version 10.

Changes to LISTDEF

After you migrate to DB2 Version 10, the LISTDEF utility supports the DEFINED YES and DEFINED NO options. By default, DEFINED YES is used, which means that only defined objects are included or excluded from LISTDEF lists. To include or exclude undefined objects in the LISTDEF, add the DEFINED NO or DEFINED ALL option.

In previous versions, if a LISTDEF list contains only undefined objects, DB2 issues message DSNU185I for each undefined object, and LISTDEF returns return code 0. In Version 10, if a list returns only undefined objects, and DEFINED NO or DEFINED ALL is not specified, LISTDEF issues DSNU1008 and returns return code 4, and DSNU185I messages are not issued.

DB2 Version 10 also adds the HISTORY option, which changes the behavior of the BASE, LOB, XML, and ALL keywords. When one of these keywords is specified and the HISTORY keyword is also specified, non-history objects are filtered out of the resulting list. When one of these keywords is specified and the HISTORY keyword is not specified, history objects are filtered out of the resulting list.

MAX_UTIL_PARTS subsystem parameter removed

As of DB2 Version 10, the MAX_UTIL_PARTS subsystem parameter is removed. Because the LOAD utility no longer uses MAX_UTIL_PARTS, there is no longer a limit on the number of partitions of a compressed table space that LOAD can process when the LOAD utility control statement does not contain the PART parameter.

Command release incompatibilities for migration from Version 8

When you migrate from Version 8 to Version 10, be aware of the command release incompatibilities.

GUPI

Plan for the following changes in Version 9.1 and Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changes to DISPLAY command location name length (change introduced in Version 9.1)

In previous releases, the DISPLAY command location name length was limited to 16 characters. Starting in Version 9.1, the length of the DISPLAY command location name can be longer than 16 characters so that the location name can have a value of an IPV6 IP address. When you implement IPV6, you should examine automation routines that examine IP addresses and make adjustments for the longer addresses.

Change to DISPLAY GROUP DETAIL command

In DB2 Version 10, the output of the DISPLAY GROUP DETAIL command includes information about subgroups, if any exist.

Related reference:

-DISPLAY GROUP (DB2) (DB2 Commands)

Changes to DISPLAY THREAD command output length (change introduced in Version 9.1)

In previous releases, if the DISPLAY THREAD command was issued from an MVS console, DB2 displayed 255 lines of thread output per type specified, per member. Starting in Version 9.1, the DISPLAY THREAD command output displays the number of lines per member that is specified in the LIMIT keyword. The default value of LIMIT is 512 lines of output per type specified, per member.

Related reference:

-DISPLAY THREAD (DB2) (DB2 Commands)

Change to REBIND PACKAGE command

In previous releases, the REBIND PACKAGE command did not preserve the old package copies by default. In Version 10, old package copies are saved when REBIND PACKAGE is used. To change this default behavior, change the value of the PLANMGMT subsystem parameter to OFF.

GUPI

Related reference:

REBIND PACKAGE (DSN) (DB2 Commands)

PLAN MANAGEMENT field (PLANMGMT subsystem parameter) (DB2 Installation and Migration)

Security release incompatibilities for migration from Version 8

When you migrate from DB2 Version 8 to Version 10, be aware of the security release incompatibilities.

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Separate security and system administrator authorities

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, users with SYSADM authority can no longer manage security objects such as trusted contexts and roles, issue grants, or revoke grants that are made by others. Also, users with SYSCTRL authority can no longer manage roles, issue grants, or revoke grants that are made by others.

If you set SEPARATE SECURITY to NO, which is the default, the SYSADM and SYSCTRL authorities have the same privileges as in the previous release.

Changes to the sample connection routine (DSN3SATH)

In previous versions of DB2, DSN3SATH retrieved the list of secondary authorization IDs from the address-space-level ACEE for connection processing for utilities that run in batch mode. DSN3SATH included code to retrieve the list of secondary authorization IDs from a task-level ACEE, but that code was commented out. In DB2 Version 10, DSN3SATH has been changed to use the task-level ACEE in this case. This change lets utilities that run under stored procedures retrieve the list of secondary authorization IDs from a task-level ACEE.

Restriction for SET CURRENT SQLID

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, there is an additional restriction for the SET CURRENT SQLID statement. If any of the authorization IDs of the process has SYSADM authority, then CURRENT SQLID must be set to one of the authorization IDs of the application process.

Restriction for BIND PACKAGE OWNER and BIND PLAN OWNER

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, there is an additional restriction for the BIND PACKAGE OWNER and the BIND PLAN OWNER. If any of the authorization IDs of the process has SYSADM authority or SYSCTRL authority, then the OWNER must be set to one of the authorization IDs of the process.

Additional role name restrictions

Previous releases prohibited using SYSADM, SYSCTRL, or PUBLIC as role names. As of Version 10, role names cannot be ACCESSCTRL, DATAACCESS, DBADM, DBCTRL, DBMAINT, NONE, NULL, PACKADM, PUBLIC, SECADM, SQLADM, or any name beginning with "SYS."

Online-updatable subsystem parameters

In previous versions of DB2, users with SYSOPR or higher authorities can update the online-updatable subsystem parameters. In DB2 Version 10, the security-related subsystem parameters can only be updated by users that have SECADM or installation SYSADM authority. The subsystem parameters that are affected by this change are:

- AUTHCACH
- BINDNV
- DBACRVW
- EXTSEC
- TCPALVER

Storage release incompatibilities for migration from Version 8

When you migrate from DB2 Version 8 to Version 10, be aware of the storage release incompatibilities.

Plan for the following changes to Version 9.1 and Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changes to the format of the BSDS (change introduced in Version 9.1)

In releases prior to Version 9.1, you could have the BSDS in a format that allowed only 1000 data sets per copy for archive logs and 31 data sets per copy for active logs. In Version 10, the BSDS must be in the format that supports up to 10,000 data sets per copy for archive logs and 93 data sets per copy for active logs. Any unconverted BSDSs are converted by the DSNJCNVB module as part of job DSNTIJUZ. To determine if DSNJCNVB has already been run, use the DSNJU004 utility. BSDSs that have been converted are unaffected by running DSNTIJUZ.

The work file database is the only temporary database (change introduced in Version 9.1)

Starting in DB2 Version 9.1, the work file database is the only temporary database. The work file database is used for all temporary tables, both external and internal, which simplifies DB2 systems management tasks. The TEMP database is no longer used by DB2.

If you explicitly or implicitly granted privileges on the TEMP database in a previous version of DB2, you need to grant those privileges on the work file database to avoid unexpected authorization errors.

Work file table spaces are now created on DB2-managed storage during installation (change introduced in Version 9.1)

In releases prior to Version 9.1, the DB2 installation process created work file table spaces on user-managed storage. In Version 10, the installation process creates work file table spaces on DB2-managed storage. The default storage group is SYSDEFLT. You can specify a different, user-managed storage group by modifying the parameters in the last step of installation job DSNTIJTM.

Related concepts:

"Work file database storage requirements" on page 35

Related reference:

"Work file database panel: DSNTIP9" on page 323

At least one 32 KB work file table space is required (change introduced in Version 9.1)

Starting in DB2 Version 9.1, you must define at least one 32 KB work file table space. In Version 8, 4 KB work file table spaces were used whenever possible. In Version 10, 32 KB work file table spaces are used when a row length is more than 100 bytes.

Declared global temporary tables and static scrollable cursor result tables require a table space with a 32 KB page size (change introduced in Version 9.1)

Declared global temporary tables and static scrollable cursor result tables require at least one of the table spaces in the work file database to have a 32 KB page size. Eight KB and 16 KB page sizes are not supported for table spaces that are created in the work file database. Member DSNTESQ of the *prefix*.SDSNSAMP library contains a sample query to check your work file database.

Also, global temporary tables require a 32 KB buffer pool, which is required to install DB2. Existing jobs that create a table space in the temporary database might also need to be modified.

New limits for the number of work file table spaces and indexes (change introduced in Version 9.1)

The number of table spaces in the work file database cannot exceed the limit of 500. The number of indexes that are defined on declared global temporary tables that belong to all agents on the local DB2 member cannot exceed the limit of 10000.

You can use the installation CLIST to add table spaces to the work file database as part of the migration process (change introduced in Version 9.1)

In releases prior to Version 9.1, you could create table spaces in the work file database only in the INSTALL mode of the installation CLIST. In Version 10, you can create zero or more additional table spaces in the work file database in the MIGRATE mode of the installation CLIST. The segment size of the table spaces is restricted to 16 until DB2 enters new-function mode.

You can specify 4 KB, 8 KB, 16 KB, or 32 KB buffer pools for user indexes (change introduced in Version 9.1)

In releases prior to Version 9.1, you could specify only 4 KB buffer pools for user indexes. In Version 10, you can specify 4 KB, 8 KB, 16 KB, or 32 KB buffer pools

for user indexes. Indexes that are created during conversion mode from Version 8 require a 4 KB buffer pool, so you need to specify a 4 KB buffer pool in the BUFFERPOOL clause when you create an index. If you don't specify a 4 KB buffer pool in the BUFFERPOOL clause when you create an index in conversion mode from Version 8, DB2 issues SQLCODE = -676, ERROR: ONLY A 4K PAGE BUFFERPOOL CAN BE USED FOR AN INDEX.

Other release incompatibilities for migration from Version 8

When you migrate from DB2 Version 8 to Version 10, be aware of the release incompatibilities.

Plan for the following changes in Version 9.1 and Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

The GROUP ATTACH process is randomized (change introduced in Version 9.1)

In DB2 Version 8, if more than one subsystem was running on a z/OS image in a data sharing group, incoming jobs went to the first active DB2 subsystem. Starting in Version 9.1, the GROUP ATTACH process is randomized by default so that all members that are defined in a data sharing group and running on a z/OS image have an equal chance of getting the attach from a starting application if the application specifies a GROUP ATTACH name instead of an individual subsystem name.

To change the Version 10 behavior to the behavior of Version 8, set the RANDOMATT subsystem parameter to NO for all members of the data sharing group. Alternatively, you can set RANDOMATT for one member of the group to YES, which causes group attachment requests to attempt to connect to that member before attempting to connect to others.

Changed default values for some subsystem parameters and installation panel fields

The default values for some subsystem parameters and installation panel fields have changed. The new values are listed in the following table.

Table 33. Subsystem parameters and installation panel fields with new default values

Parameter	Field	Panel	Version 8 default value	Version 10 default value
none	BP8K0	DSNTIP2	1000	2000
none	DATABASES	DSNTIPE	100	200
CACHEPAC	PACKAGE AUTH CACHE	DSNTIPP	100K	5M
CACHERAC	ROUTINE AUTH CACHE	DSNTIPP	100K	5M
CHKFREQ	MINUTES/CHECKPOINT	DSNTIPL1	500000 records	5 minutes
CONTSTOR	CONTRACT THREAD STG	DSNTIPE	NO	YES
DB2SORT	UT DB2 SORT USE (none in Version 8)	DSNTIP61	DISABLE	ENABLE
DLDFREQ	LEVELID UPDATE FREQ	DSNTIPL1	5	ON

Table 33. Subsystem parameters and installation panel fields with new default values (continued)

Parameter	Field	Panel	Version 8 default value	Version 10 default value
DSMAX	DSMAX	DSNTIPC	9960	20000
EDMDBDC	EDM DBD CACHE	DSNTIPC	102396K	23400K
EDMPOOL	EDMPOOL LIMIT BELOW THE BAR	DSNTIPC	32767K	0[1]
EDMSTMTC	EDM STATEMENT CACHE	DSNTIPC	102396K	113386K
IRLMRWT	RESOURCE TIMEOUT	DSNTIPI	60	30
IRLMSWT	TIME TO AUTOSTART	DSNTIPI	300	120
LRDRTHLD	LONG-RUNNING READER	DSNTIPE	0	10
MAXRBLK	RID POOL SIZE	DSNTIPC	8000K	400000K
MONSIZE	MONITOR SIZE	DSNTIPN	256K	1M
NUMLKTS	LOCKS PER TABLE(SPACE)	DSNTIPJ	1000	2000
PCLOSEN	RO SWITCH CHKPTS	DSNTIPL1	5	10
PRIVATE_PROTOCOL	none	none	YES	NO
RRULOCK	U LOCK FOR RR/RS	DSNTIPI	NO	YES
SEQCACH	none	none	BYPASS	SEQ
SEQPRES	none	none	NO	YES
SRTPOOL	SORT POOL SIZE	DSNTIPC	2000K	10000K
STATIME	STATISTICS TIME	DSNTIPN	5	1
STATROLL	STATISTICS ROLLUP	DSNTIP61	NO	YES
URCHKTH	UR CHECK FREQ	DSNTIPL1	0	5
URLGWTH	UR LOG WRITE CHECK	DSNTIPL1	0	10K

1. Although the default value of EDMPOOL is 0, if you are migrating from Version 8, you should use the same value for EDMPOOL that you used in Version 8. For more information about setting this parameter, see "EDM LIMIT BELOW THE BAR (EDMPOOL subsystem parameter)" on page 493.

If the values that you specified for these parameters are lower than the new default values, you might want to increase your values.

Changed maximum values for some subsystem parameters

The maximum values for some subsystem parameters have changed. The new values are listed in the following table.

Table 34. Subsystem parameters with new maximum values

Parameter	Version 8 maximum value	Version 10 maximum value
CACHEPAC	5M	10M
CACHERAC	5M	10M
CTHREAD	2000	20000
DSSTIME	1440	60
IDBACK	2000	20000
IDFORE	2000	20000
MAXDBAT	1999	19999
MONSIZE	16M	64M

Table 34. Subsystem parameters with new maximum values (continued)

Parameter	Version 8 maximum value	Version 10 maximum value
STATIME	1440	60

Changed acceptable values for subsystem parameter NPGTHRSH (change introduced in Version 9.1)

Starting in Version 9.1, a value of -1 for subsystem parameter NPGTHRSH is no longer supported. The acceptable values for NPGTHRSH are 0 or any integer greater than 1.

Changed acceptable values for the DLDFREQ parameter

The values 0 through 32767 for the DLDFREQ subsystem parameter are no longer supported in Version 10. The acceptable values for DLDFREQ in Version 10 are ON and OFF. A value of 0 in previous releases corresponds to a value of OFF in Version 10. Values 1 through 32767 in previous releases correspond to a value of ON in Version 10.

Changed minimum value for the MONSIZE parameter

The minimum value for the MONSIZE subsystem parameter has changed from 256K to 1M.

Changed minimum value for the OUTBUFF parameter

The minimum value for the OUTBUFF parameter has changed from 40K to 400K. If you specify a value less than 400K, the installation process changes the value to 400K. Check the region size in the startup procedure for the DB2 system services address space (DSN1MSTR) to ensure that it is large enough to accommodate the increased output buffer size. The recommended setting for the REGION size of the address space is 0M.

Changes to STATIME and SYNCVAL parameters

In DB2 Version 10, the STATIME and SYNCVAL subsystem parameters apply only to IFCIDs 0105, 0106, and 0199. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME and SYNCVAL, and the corresponding trace records are written at fixed, one-minute intervals.

Changed maximum degree of parallelism for parallel groups

DB2 Version 10 uses a changed maximum degree of parallelism when the subsystem parameter PARAMDEG is set to 0. To avoid storage shortages, the maximum degree of parallelism has been reduced from ten times the number of CPUs to two times the number of CPUs. If an increase in elapsed time for parallel queries is a problem, specify a value for PARAMDEG that is greater than two times the number of CPUs.

Ensure that the access control authorization exit routine is compatible with Version 10

Ensure that the access control authorization exit routine (DSNX@XAC) is compatible with Version 10. The access control authorization exit routine needs to turn on the Version 10 version identifier (EXPLV10) during access control exit initialization processing. For more information about other Version 10 changes to the access control authorization exit, see Considerations for the access control authorization routine (Managing Security).

Changes to the real-time statistics tables (change introduced in Version 9.1)

In Version 10 new-function mode, the real-time statistics tables are part of the DB2 catalog. In Version 10 conversion mode from Version 8, the real-time statistics data is still in the user-defined tables. During enabling-new-function mode from Version 8 processing, job DSNTIJEN moves the real-time statistics data from your user-defined tables to catalog tables SYSIBM.SYSTABLESPACESTATS and SYSIBM.SYSINDEXSPACESTATS. If you do not have a user-defined real-time statistics database (DSNRTSDB) defined, DSNTIJEN issues a warning. If you revert to conversion mode* from Version 8, DB2 keeps the real-time statistics data in the catalog tables and does not use the user-defined tables. You can drop the user-defined tables after job DSNTIJEN moves the data to the catalog tables.

In releases before Version 9.1, in the user-defined real-time statistics tables, the data types of TOTALROWS and SPACE in SYSTABLESPACESTATS and TOTALENTRIES in SYSINDEXSPACESTATS were FLOAT. In Version 10, they are the BIGINT data type.

Changes to resource limit facility tables (change introduced in Version 9.1)

In releases before Version 9.1, there was a single type of resource limit facility table with a name of the form *authid*.DSNRLST*nn*. If the START RLIMIT command was executed and the resource limit facility table did not exist, the START RLIMIT command terminated. In Version 10, there are two types of resource limit facility tables: tables of the form *authid*.DSNRLST*nn* and tables of the form *authid*.DSNRLMT*nn*. In Version 10 conversion mode from Version 8, if *authid*.DSNRLST*nn* does not exist, START RLIMIT terminates. In Version 10 new-function mode, the START RLIMIT command can complete successfully if either type of resource limit facility table exists.

Resource limit facility changes

Before DB2 Version 10, resource limit facility (RLF) limits applied to connections from remote users. These limits applied, regardless of whether the primary or secondary authorization IDs of the users had installation SYSADM or installation SYSOPR authority. Starting with DB2 Version 10, no RLF limits apply to primary or secondary authorization IDs that have installation SYSADM or installation SYSOPR authority.

Changed behavior for DDF to use getaddrinfo() instead of gethostid() to obtain IP addresses

DB2 Version 10 is an IPv6 system and cannot use gethostid() to obtain IP addresses because gethostid() does not support IPv6. Instead, DDF uses getaddrinfo() to obtain IP addresses. getaddrinfo() resolves the host name that is obtained from gethostname() to one or more IP addresses that are specified in the DNS or local host file. Therefore, it is important that the TCP/IP host name that is obtained from gethostname() matches the host name that is specified as part of the domain name in the DNS or local host file, so that an IP address can be obtained.

For details about other methods of specifying an IP address, see “Starting TCP/IP support” on page 929.

Changed behavior with reactive governing (change introduced in Version 9.1)

The resource limit facility no longer accumulates ASUTIME across SQL statement boundaries. For reactive governing, DB2 resets the ASUTIME after a single statement execution is completed for a parameter marker in an INSERT or

UPDATE statement, or after the close and reopen of a cursor for a FETCH statement. You might need to change your ASUTIME values in the resource limit facility to account for the new behavior. However, if you fall back to DB2 Version 8, you need to adjust the ASUTIME values again to prevent -905 errors.

Fewer SMP/E jobs are required (change introduced in Version 9.1)

Because utilities are now contained in a single FMID, SMP/E jobs DSNRECVM, DSNRECVS, DSNAPPLM, DSNAPPLS, DSNACCPM, and DSNACCPS have been deleted.

DB2 Version 10 requires IRLM 2.3

IRLM 2.3 is delivered with DB2 Version 10. You must use the DB2-supplied IRLM procedure. Although IRLM 2.3 is required in DB2 Version 10, you can use the DXRRL183 load module from IRLM 2.2.

Changes to table space types

The default table space type for new tables is now the partition-by-growth universal table space. The partitioned table space type is deprecated. In new-function mode, you can use an ALTER statement to change single-table table spaces to partition-by-growth universal table spaces.

You cannot use an ALTER statement to change multiple-table table spaces to universal table spaces.

Altered catalog and directory table spaces

During processing in enabling-new-function mode (from both Version 8 and Version 9.1), some catalog and directory table spaces are altered to use row-level locking and some tables use reordered-row format. Changes to the structure of the catalog also include new table spaces, new tables, and the use of new LOB columns. As a result, backup and recovery for the catalog is substantially changed.

DSNTIJIC uses SHRLEVEL CHANGE to copy the DB2 catalog and directory

In DB2 Version 10, job DSNTIJIC is modified to specify SHRLEVEL CHANGE rather than SHRLEVEL REFERENCE. This modification allows other tasks to update the DB2 catalog and directory while DSNTIJIC is executing. If you want to restrict other tasks to only access the catalog and directory when DSNTIJIC is executing, modify your customized copy of DSNTIJIC to use SHRLEVEL REFERENCE.

Change to job DSNTIJMV

In DB2 Version 10, job DSNTIJMV is modified to use the keyword parameter format for the subsystem parameter control statement in IEFSSN*xx* members. In previous versions, job DSNTIJMV used the positional parameter format. The two parameter formats cannot be used together in the same IEFSSN*xx* member.

Example of keyword parameter format:

```
SUBSYS SUBNAME(ssname) INITRTN(DSN3INI)
  INITPARM('DSN3EPX,cmd-prefix<,scope<,group-attach<<,subgroup-attach>>>')
```

Example of positional parameter format:

```
ssname,DSN3INI,'DSN3EPX,cmd-prefix<,scope<,group-attach<<,subgroup-attach>>>'
```

Changes to pending states for ALTER statements

Before Version 10, some ALTER statements produce immediate definition changes, and the table space or index defined in a statement is placed in a restrictive REBUILD-pending (RBDP) or restrictive REORG-pending (REORP) state. In Version 10 new-function mode, these same ALTER statements are pending definition changes, and the table space or index defined in a statement is placed in an advisory REBUILD-pending (ARBDP) or advisory REORG-pending (AREOR) state. Changes are applied when the table space or index is reorganized or the index is rebuilt, and data in the table space or index can be accessed or updated.

SYSDATE added as synonym for CURRENT TIMESTAMP(0)

As of Version 10, SYSDATE is a synonym for CURRENT TIMESTAMP(0). After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), if an object named SYSDATE already exists, all references to the object should specify the name as a delimited identifier ("SYSDATE") to distinguish the object from the CURRENT TIMESTAMP(0) synonym. References to SYSDATE as an ordinary identifier result in SQLCODE -4700.

Added precision variable for CURRENT TIMESTAMP

As of Version 10, an optional timestamp precision can be specified for the CURRENT TIMESTAMP special register. The precision is the number of digits of fraction seconds in the timestamp. If a user-defined function with the name CURRENT_TIMESTAMP already exists, all references to the function should specify the name as a delimited identifier ("CURRENT_TIMESTAMP") after migration to Version 10 conversion mode (from Version 8 or Version 9.1). References to CURRENT_TIMESTAMP in a function call result in SQLCODE -4700.

Catalog and directory data sets converted to SMS-controlled

In DB2 Version 10, all data sets for the DB2 catalog and directory must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating data sets in extended format and to use extended addressability (EA). See member DSNTIJSS of the Version 10 SDSNSAMP library for a sample SMS environment. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

During enabling-new-function mode (from both Version 8 and Version 9.1), some existing table spaces are converted to standalone table spaces. The obsolete table spaces are deleted in enabling-new-function mode (from both Version 8 and Version 9.1), at which time the data sets for those table spaces are also deleted. For details on the table space conversion, see “Job DSNTIJEN” on page 655.

SMF subtype field defined as a two-byte value

The SMF subtype field should be defined as a two-byte value. Previously, DB2 mapped the field SM10xSTF as a single-byte field that corresponds to the high half of the standard two-byte field location. In Version 10, SM10xSTF is changed to a two-byte field. This change results in the removal of unused field SM10xRI from the mapping. This change might require change to applications that parse SMF.

Changed default OP buffer size

The online performance (OP) buffer size in Version 10 defaults to 1 MB unless otherwise specified.

Change to processing of IEEE BFP values

When an underflow IEEE BFP value is encountered during local processing, SQLCODE -802 is issued. In Version 8, no SQLCODE was issued in this case.

Changed sizes and definitions of trace records

DB2 Version 10 introduces changes to the definitions and sizes of some IFCID fields. These changes might require modifications to your trace applications.

For more information, see Changed IFCIDs in Version 9 (DB2 for z/OS What's New?) and Changed IFCIDs in Version 10 (DB2 for z/OS What's New?).

Changed format of the IFCID 0002, 0003, 0148, and 0306 trace records

For IFCID 0002, the format of the distributed data facility area, which is mapped by QLST fields, has changed substantially. For IFCID 0003, the format of the distributed data facility area, which is mapped by QLAC fields, has changed substantially. For IFCID 0148, the format of the distributed accounting records has changed substantially. Any application that reads and interprets location statistics must be modified to account for the Version 10 format of IFCID 0002, 0003, and 0148. For IFCID 0306, the definition of field QW0306AD has changed and the size of the area mapped by field QW0306OF has increased. The first eight bytes of field QW0306AD are now “V10 ” to indicate that the data is from DB2 Version 10 and that the Version 10 mapping should be used.

Change to connection and sign-on routines

Explanation

In previous releases, the EXPLSITE value that is passed to the DSN3@ATH connection and DSN3@SGN sign-on exit routines, relative to an upstream DB2 for z/OS requester system partner, could be either a network address or the partner's actual location name. In DB2 Version 10, the connection and sign-on exit routines are passed an EXPLSITE value that reflects a network address that is relative to the DB2 for z/OS requester system.

Possible impact to your DB2 environment

Because the connection and sign-on exit routines might base authentication decisions on the EXPLSITE value, you might have different authentication results in Version 10 than in previous releases.

Actions to take

To prepare for this change, examine the logic of your connection and sign-on exit routines. Determine if they have dependencies on the EXPLSITE value being relative to remote upstream DB2 for z/OS requester partner location names. If such dependencies exist, before migration to Version 10, modify the exit routine logic to accept EXPLSITE values that are either the network address or the actual location name for upstream DB2 for z/OS requester partners. After migration to Version 10 new-function mode, you can optionally modify the exit routines to remove dependencies on the EXPLSITE value being an actual DB2 for z/OS requester partner location name.

Functions that are deprecated for systems migrating from Version 8

During migration from DB2 Version 8, be aware of the functions that are deprecated in Version 10. Although they are supported in Version 10, support for these functions might be removed in a later release. Avoid creating new dependencies that rely on these functions, and if you have existing dependencies on them, develop plans to remove these dependencies.

The following functions are deprecated in Version 10.

Some DB2-supplied stored procedures

The following DB2-supplied stored procedures are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- SYSPROC.DSNACCOR - Consider using the SYSPROC.DSNACCOX stored procedure instead.
- SYSPROC.DSNAEXP - Consider using the EXPLAIN privilege in place of DSNAEXP to explain statements for users that do not have the authority to execute EXPLAIN statements
- SYSPROC.DSNTBIND - Consider using the SYSPROC.ADMIN_COMMAND_DSN stored procedure instead.
- SYSPROC.DSNUTILS - Consider using the SYSPROC.DSNUTILU stored procedure instead.
- SYSPROC.DSNWZP - Consider using the SYSPROC.ADMIN_INFO_SYSPARM stored procedure instead.

Some subsystem parameters

The following subsystem parameters are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- DISABSCL: In later DB2 releases, subsystems behave as if DISABSCL is set to NO.
- DPSEGSZ: In later DB2 releases, subsystems behave as if DPSEGSZ is set to 32.
- OJPERFEH: In later DB2 releases, subsystems behave as if OJPERFEH is set to YES.
- OPTIOWGT: In later DB2 releases, subsystems behave as if OPTIOWGT is set to ENABLE.
- OPTIXIO: In later DB2 releases, subsystems behave as if OPTIXIO is set to ON.
- PTCDIO: In later DB2 releases, subsystems behave as if PTCDIO is set to OFF.
- RETVLCFK: In later DB2 releases, subsystems behave as if RETVLCFK is set to NO. Use of non-padded indexes is recommended.
- SEQCACH: In later DB2 releases, subsystems behave as if SEQCACH is set to SEQ.
- SEQPRES: In later DB2 releases, subsystems behave as if SEQPRES is set to YES.
- SMSDCFL: In later DB2 releases, you can use the SMS data class parameter with CREATE STOGROUP and ALTER STOGROUP statements instead.
- SMSDCIX: In later DB2 releases, you can use the SMS data class parameter with CREATE STOGROUP and ALTER STOGROUP statements instead.

Acceptable value for DDF_COMPATIBILITY subsystem parameter

The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in Version 10, it will be removed in a later release of DB2.

Some DB2-supplied user-defined functions

The following DB2-supplied user-defined functions are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- DB2XML.SOAPHTTPC
- DB2XML.SOAPHTTPV

Interfaces to DSNHDECP (deprecated in Version 9.1)

Some uses of DSNHDECP are deprecated and will be removed in a later release. If you have code that loads DSNHDECP and maps it with macros, you should plan to change that code by using the techniques that are described in this section.

In DB2 Version 10 new-function mode, you can specify a module name other than DSNHDECP to be used by a DB2 instance in the same way that you can specify other DB2 parameters. In support of this change, DB2 Version 10 has the following interfaces:

- The RRS attachment facility (RRSAF) provides information about which DB2 subsystems are available on an LPAR and provides the address of the DSNHDECP load module that a DB2 subsystem uses. (DSNHDECP is loaded in common storage ECSA.)
- IFCID 0106 provides the fully qualified DSNHDECP load module data set name that a DB2 subsystem uses.
- The precompiler provides the fully qualified data set name for the DSNHDECP load module that is used in a precompilation.
- The GETVARIABLE function provides all of the information that the DSNHDECP module provides.
- The START DB2 command returns a message that includes the fully qualified data set name for the DB2 subsystem parameter module, DSNHDECP load module, and exit routines DSN3@ATH, DSN3@SGN, and DSNX@XAC.

EXPLAIN tables in Version 8 or Version 9 format

In DB2 Version 10, EXPLAIN tables that are in a Version 8 or Version 9 format are deprecated. These tables should be converted to the Version 10 format. Starting in DB2 conversion mode (from both Version 8 and Version 9.1), statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9 format, regardless of the encoding type. For more information about converting your EXPLAIN tables to Version 10 format, see “Migration step 24: Convert EXPLAIN tables to the current format and encoding type” on page 608.

NEWFUN SQL processing options and DECP values

The SQL processing options NEWFUN(YES) and NEWFUN(NO) are deprecated and NEWFUN(V8), NEWFUN(V9), and NEWFUN(V10) options are added in

Version 10. Although these deprecated options are supported in Version 10, you should use NEWFUN(V10) instead of NEWFUN(YES) and use NEWFUN(V8) instead of NEWFUN(NO).

The *dsnhdecp* parameter values NEWFUN=YES and NEWFUN=NO are also deprecated. Although these values are supported in Version 10, you should use NEWFUN=V10 instead of NEWFUN=YES and use NEWFUN=V8 instead of NEWFUN=NO.

Password protection for active log and archive log data sets

Support for password-protected active log and archive log data sets is deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

Functions that are no longer supported for systems migrating from Version 8

If you are migrating to DB2 Version 10 from Version 8, be aware of the functions that are no longer supported.

The following functions are not supported in Version 10.

BookManager® format is being discontinued

IBM is discontinuing the BookManager format as part of an ongoing effort to deliver the best possible information experience to our customers. IBM is optimizing the DB2 for z/OS information for viewing in the information center.

The BookManager books will continue to be provided for the life cycle of DB2 UDB for z/OS Version 8 and DB2 Version 9.1 for z/OS; however, IBM is not delivering BookManager books beginning in DB2 Version 10 for z/OS. The Information Management Software for z/OS Solutions Information Center includes all of the information that you are used to seeing in BookManager and PDF, plus many improved features for finding and viewing the information that you need to do your job.

msys for Setup DB2 Customization Center

In DB2 Version 10, msys for Setup DB2 Customization Center is removed. Instead, use the installation panels. For more information about using the installation panels, see Chapter 4, “Tailoring DB2 jobs to your environment using the installation CLIST,” on page 219.

Optimization Service Center

As of DB2 Version 10, Optimization Service Center (OSC) is no longer supported. The recommended no-charge query optimization and service tool is IBM Data Studio, which is a query serviceability tool for DB2 for z/OS. This tool is based and built on the foundation of Optimization Service Center and Optimization Expert.

Java stored procedures no longer run in resettable JVMs (starting in Version 9.1)

In releases before Version 9.1, Java stored procedures could be run in resettable JVMs. In Version 10, Java stored procedures cannot be run in resettable JVMs.

Because Java stored procedures cannot be run in resettable JVMs, they are no longer automatically prevented from impacting the execution of future Java stored procedures in the following ways:

- Setting system properties.
- Attempting to access or modify the security configuration objects.
- Loading a native library.

Such actions can impact the ability of other Java stored procedures to use that JVM. For more information about resettable JVMs and for a full list of actions that are prevented, see the "Unresettable actions" topic of the Persistent Reusable Java Virtual Machine User's Guide.

Recommendation: After you migrate to Version 10, modify your applications that use functions of resettable JVMs to not run Java stored procedures in resettable JVMs.

DB2-established stored procedure address spaces (starting in Version 9.1)

DB2-established stored procedure address spaces are no longer supported. Stored procedures must be moved to a WLM environment.

Recommendation: Before you migrate to Version 10, modify your DB2-managed stored procedures to be managed by WLM. For more information, see "WLM management of stored procedures" on page 839.

JDBC/SQLJ Driver for OS/390® and z/OS (starting in Version 9.1)

The JDBC/SQLJ Driver for OS/390 and z/OS is no longer supported. All Java application programs and Java routines that are currently written to work with the JDBC/SQLJ Driver for OS/390 and z/OS need to be modified to work with the IBM Data Server Driver for JDBC and SQLJ (formerly known as the DB2 Universal JDBC Driver). See the steps for migrating JDBC and SQLJ applications to the IBM Data Server Driver for JDBC and SQLJ at Migrating from the JDBC/SQLJ Driver for OS/390 and z/OS to the IBM Data Server Driver for JDBC and SQLJ.

All WLM-managed stored procedures address spaces that are set up to execute Java routines must be modified to reference the IBM Data Server Driver for JDBC and SQLJ. Existing WLM environments that are configured to use the JDBC driver fail when the address space initializes. Data sharing groups that have Version 8 or Version 9.1 members in coexistence with members in Version 10 conversion mode (from Version 8 or Version 9.1) fail if a Java routine is invoked on any Version 10 members where the WLM-SPAS JCL does not reference the IBM Data Server Driver for JDBC and SQLJ.

Recommendation: Before you migrate to Version 10, modify your WLM-SPAS JCL to use the IBM Data Server Driver for JDBC and SQLJ.

Creation of simple table spaces (starting in Version 9.1)

Simple table spaces that were created in previous releases can still be read and updated. However, you can no longer create simple table spaces. Because DB2 does not support creating new table spaces in Version 10, it is possible to accidentally drop a simple table space that you cannot recover.

Recommendation: To avoid accidentally dropping a simple table space that you cannot recover, complete one of the following tasks before you migrate to Version 10:

- Issue an ALTER TABLE ... ADD RESTRICT ON DROP statement against tables that are in a simple table space to restrict dropping these simple table spaces.
- Convert your simple table spaces.

Connections from VAX machines and the PASCAL L string data type (starting in Version 9.1)

For DRDA connections to DB2 for z/OS, the following items are no longer supported:

- Connections from VAX machines
- The PASCAL L string data type

AIV Extender, Text Extender, and Net Search Extender (starting in Version 9.1)

AIV Extender, Text Extender, and Net Search Extender have been removed. DB2 Version 10 incorporates substantial text function into the base product. Use the new functions and indexing for text applications.

Net.Data® (starting in Version 9.1)

Net.Data has been removed. WebSphere® is the strategic IBM solution for delivering DB2 data to web applications.

DB2 QMF Visionary Studio program (starting in Version 9.1)

DB2 QMF Visionary Studio program has been removed from DB2 QMF Enterprise Edition.

DB2 Estimator (starting in Version 9.1)

DB2 Estimator, which was available for download from the web for DB2 Version 8, is not provided for DB2 Version 10.

Visual Explain for DB2 for z/OS (starting in Version 9.1)

Visual Explain for DB2 for z/OS, which was available for download from the web for DB2 Version 8, is not provided for DB2 Version 10. Instead, IBM Data Studio for DB2 for z/OS, part of the IBM DB2 Accessories Suite for z/OS, extends the capability of Visual Explain and adds more SQL tuning functions. IBM Data Studio works with both DB2 Version 8 and DB2 Version 10.

BookManager-based online help (starting in Version 9.1)

BookManager-based online help has been removed. For help with installation and DB2I, see the following website:

`http://publib.boulder.ibm.com/infocenter/imzic`

DB2 XML Extender

DB2 XML Extender is no longer supported. In DB2 Version 10, you can use the XML data type instead.

DB2 Management Clients feature

DB2 Management Clients feature, which includes DB2 Administration Server, DB2 Control Center, and DB2 Development Center, is no longer supported. Instead, use IBM Data Studio to design, develop, deploy, and manage your data-driven applications.

Several subsystem parameters

The DBPROTCL subsystem parameter is no longer supported. The default for the DBPROTOCOL bind option is DRDA. You can still specify DBPROTOCOL(PRIVATE) explicitly when binding packages and plans, but DB2 issues a warning message if you perform any bind with this value.

The following subsystem parameters have been removed:

- EDMBFIT
- LOGAPSTG
- MAX_OPT_ELAP
- MAX_UTIL_PARTS
- MORE_UNION_DISTRIBUTION
- OPTIXOPREF
- PARTKEYU
- RELCURHL

  In Version 10, the option to hold a lock after commit is eliminated. DB2 always releases the page or row lock for cursor hold at commit, which allows for greater concurrency.
- SJMISSKY
- STORPROC
- SUPPRESS_TS_CONV_WARNING

  In Version 10, DB2 always operates as if SUPPRESS_TS_CONV_WARNING=NO.
- TABLES_JOINED_THRESHOLD

Some DB2-supplied routines

The following DB2-supplied routines are removed in DB2 Version 10 and are unavailable to callers after migration to conversion mode (from Version 8 or Version 9.1). A report is added to the DSNTIJPM premigration job to detect occurrences of these routines on an existing subsystem or data sharing group, and to specify that these routines are not available in Version 10.

- DSN8.DSN8EXP
- AMI-based DB2 MQ functions

- DB2MQ1C.GETCOL
- DB2MQ1C.MQPUBLISH
- DB2MQ1C.MQREAD
- DB2MQ1C.MQREADALL
- DB2MQ1C.MQREADALLCLOB
- DB2MQ1C.MQREADCLOB
- DB2MQ1C.MQRECEIVE
- DB2MQ1C.MQRECEIVEALL
- DB2MQ1C.MQRECEIVEALLCLOB
- DB2MQ1C.MQRECEIVECLOB
- DB2MQ1C.MQSEND
- DB2MQ1C.MQSUBSCRIBE
- DB2MQ1C.MQUNSUBSCRIBE
- DB2MQ2C.GETCOL
- DB2MQ2C.MQPUBLISH
- DB2MQ2C.MQREAD
- DB2MQ2C.MQREADALL
- DB2MQ2C.MQREADALLCLOB
- DB2MQ2C.MQREADCLOB
- DB2MQ2C.MQRECEIVE
- DB2MQ2C.MQRECEIVEALL
- DB2MQ2C.MQRECEIVEALLCLOB
- DB2MQ2C.MQRECEIVECLOB
- DB2MQ2C.MQSEND
- DB2MQ2C.MQSUBSCRIBE
- DB2MQ2C.MQUNSUBSCRIBE
- DB2MQ1N.GETCOL
- DB2MQ1N.MQPUBLISH
- DB2MQ1N.MQREAD
- DB2MQ1N.MQREADALL
- DB2MQ1N.MQREADALLCLOB
- DB2MQ1N.MQREADCLOB
- DB2MQ1N.MQRECEIVE
- DB2MQ1N.MQRECEIVEALL
- DB2MQ1N.MQRECEIVEALLCLOB
- DB2MQ1N.MQRECEIVECLOB
- DB2MQ1N.MQSEND
- DB2MQ1N.MQSUBSCRIBE
- DB2MQ1N.MQUNSUBSCRIBE
- DB2MQ2N.GETCOL
- DB2MQ2N.MQPUBLISH
- DB2MQ2N.MQREAD
- DB2MQ2N.MQREADALL
- DB2MQ2N.MQREADALLCLOB
- DB2MQ2N.MQREADCLOB

- DB2MQ2N.MQRECEIVE
- DB2MQ2N.MQRECEIVEALL
- DB2MQ2N.MQRECEIVEALLCLOB
- DB2MQ2N.MQRECEIVECLOB
- DB2MQ2N.MQSEND
- DB2MQ2N.MQSUBSCRIBE
- DB2MQ2N.MQUNSUBSCRIBE

• MQSeries® publish and subscribe XML routines
  - DB2MQ1C.PUBLISHXML
  - DB2MQ2C.PUBLISHXML

• MQSeries XML routines
  - DMQXML1C.DXXMQGEN
  - DMQXML1C.DXXMQGENCLOB
  - DMQXML1C.DXXMQINSERT
  - DMQXML1C.DXXMQINSERTALL
  - DMQXML1C.DXXMQINSERTALLCLOB
  - DMQXML1C.DXXMQINSERTCLOB
  - DMQXML1C.DXXMQRETRIEVE
  - DMQXML1C.DXXMQRETRIEVECLOB
  - DMQXML1C.DXXMQSHRED
  - DMQXML1C.DXXMQSHREDALL
  - DMQXML1C.DXXMQSHREDALLCLOB
  - DMQXML1C.DXXMQSHREDCLOB
  - DMQXML1C.MQPUBLISHXML
  - DMQXML1C.MQRCVALLXMLCLOB
  - DMQXML1C.MQREADALLXML
  - DMQXML1C.MQREADALLXMLCLOB
  - DMQXML1C.MQREADXML
  - DMQXML1C.MQREADXMLCLOB
  - DMQXML1C.MQRECEIVEALLXML
  - DMQXML1C.MQRECEIVEXML
  - DMQXML1C.MQRECEIVEXMLCLOB
  - DMQXML1C.MQSENDXML
  - DMQXML1C.MQSENDXMLFILE
  - DMQXML1C.MQSENDXMLFILECLOB
  - DMQXML2C.DXXMQGEN
  - DMQXML2C.DXXMQGENCLOB
  - DMQXML2C.DXXMQINSERT
  - DMQXML2C.DXXMQINSERTALL
  - DMQXML2C.DXXMQINSERTALLCLOB
  - DMQXML2C.DXXMQINSERTCLOB
  - DMQXML2C.DXXMQRETRIEVE
  - DMQXML2C.DXXMQRETRIEVECLOB
  - DMQXML2C.DXXMQSHRED
  - DMQXML2C.DXXMQSHREDALL

- DMQXML2C.DXXMQSHREDALLCLOB
- DMQXML2C.DXXMQSHREDCLOB
- DMQXML2C.MQPUBLISHXML
- DMQXML2C.MQRCVALLXMLCLOB
- DMQXML2C.MQREADALLXML
- DMQXML2C.MQREADALLXMLCLOB
- DMQXML2C.MQREADXML
- DMQXML2C.MQREADXMLCLOB
- DMQXML2C.MQRECEIVEALLXML
- DMQXML2C.MQRECEIVEXML
- DMQXML2C.MQRECEIVEXMLCLOB
- DMQXML2C.MQSENDXML
- DMQXML2C.MQSENDXMLFILE
- DMQXML2C.MQSENDXMLFILECLOB

EXPLAIN tables in pre-Version 8 format and EXPLAIN tables in EBCDIC encoding

In DB2 Version 10, EXPLAIN tables in a pre-Version 8 format are no longer supported. Before migration to Version 10, these tables must be converted to the format of the current (Version 8 or Version 9) release. Starting in DB2 10 conversion mode (from both Version 8 and Version 9), DB2 returns SQLCODE -20008 for statements or commands that invoke EXPLAIN processing if EXPLAIN tables are in a pre-Version 8 format.

Also in Version 10, EXPLAIN tables that are encoded in EBCDIC are no longer supported. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned. However, for transition purposes, EXPLAIN tables that are in Version 8 or Version 9.1 format can be encoded in EBCDIC. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9.1 format, regardless of the encoding type. For more information about converting your EXPLAIN tables, see "Converting EXPLAIN tables (before migration)" on page 64.

Private protocol

The private protocol remote data access method is no longer supported as of Version 10. DRDA protocol must be used instead. Packages and plans that were bound with DBPROTOCOL(PRIVATE) in a previous release can exist in Version 10 and might run in Version 10. If an application that executes packages or plans that were bound with DBPROTOCOL(PRIVATE) only accesses objects that are local to the DB2 subsystem, the application should run successfully. If an application that executes packages that were bound with DBPROTOCOL(PRIVATE) accesses a remote location, the application fails.

REORG TABLESPACE SHRLEVEL NONE on LOB table spaces

In DB2 Version 10 new-function mode, running the REORG TABLESPACE utility with the SHRLEVEL NONE option on a LOB table space is no longer supported. If SHRLEVEL NONE is specified, the utility immediately completes with return code 0 without performing the reorganization. SHRLEVEL REFERENCE or SHRLEVEL CHANGE should be used instead.

Changes that might affect your migration from Version 9.1

If you are migrating from DB2 Version 9.1, be aware of the changes that might affect your migration.

Proper planning should alleviate any system inconveniences.

Availability of new functions in conversion mode from Version 9.1 and new-function mode

In most cases, new functions are available only in new-function mode. However, optimization and virtual storage enhancements also work in conversion mode from Version 9.1.

The DB2 10 for z/OS information assumes that your DB2 subsystem is running in Version 10 new-function mode. Generally, new functions that are described, including changes to existing functions, statements, and limits, are available only in new-function mode. The exceptions to this general statement are new and changed optimization enhancements, which are also available in conversion mode from Version 9.1 unless stated otherwise. Also, the convergence of the temporary database into the work file temporary database applies to conversion mode from Version 9.1.

If you migrate from Version 9.1, you can use Version 9.1 new-function mode functions in Version 10 conversion mode from Version 9.1. Any functions that are available in conversion mode from Version 9.1 are also available in all other modes before new-function mode.

Application and SQL release incompatibilities for migration from Version 9.1

When you migrate from DB2 Version 9.1 to Version 10, be aware of the application and SQL release incompatibilities.

GUPI

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Automatic rebind of plans and packages created before DB2 Version 6

If you have plans and packages that were bound before DB2 Version 6 and you specified YES or COEXIST in the AUTO BIND field of panel DSNTIPO, DB2 Version 10 autobinds these packages. Thus, you might experience an execution delay the first time that such a plan is loaded. Also, DB2 might change the access path due to the autobind, potentially resulting in a more efficient access path.

If you specify NO in the AUTO BIND field of panel DSNTIPO, DB2 Version 10 returns SQLCODE -908, SQLSTATE 23510 for each attempt to use such a package or plan until it is rebound.

IBMREQD is no longer reliable as a release dependency mark

The IBMREQD field in DB2 catalog tables is no longer a reliable indicator for determining release dependencies. Use the RELCREATED or RELBOUND fields instead.

Changes to string formatting of decimal data

Explanation

DB2 Version 10 changed the formatting of decimal data by the CHAR and VARCHAR built-in functions and CAST specifications with a CHAR or VARCHAR result type. For input data that contains decimals, leading zeros are removed, and leading zeros are not added to values that did not already contain leading zeros. If the scale of the decimal value is zero, the decimal character is not returned. Also, the CHAR function no longer returns leading blanks for positive decimal values. The result of the CHAR function for decimal data is now consistent with the result of CAST(*decimal-expression* AS CHAR).

After migration to Version 10, packages that were bound before Version 10 use the old behavior for these functions. Materialized query tables and indexes on expressions that were created before Version 10 also continue to use the old behavior.

Views and inline SQL functions use the behavior of the SQL statement that references the object. It is possible for references to the same view or function in different applications to get different behavior for these functions or casts.

Possible impact to your DB2 environment

These changes might cause unexpected output from applications that use the CHAR or VARCHAR functions for decimal data or the CAST(*decimal-expression* AS CHAR) or CAST(*decimal-expression* AS VARCHAR) specifications.

Actions to take

These changes occur in Version 10 conversion mode (from both Version 8 and Version 9.1). You can temporarily override these changes on a subsystem level by using the BIF_COMPATIBILITY subsystem parameter. You can also temporarily override these changes on an application level by adding schema SYSCOMPAT_V9 to the front of the PATH bind option or CURRENT PATH special register. This approach works for CHAR and VARCHAR functions and does not affect CAST specifications. The recommended approach is to modify your applications to handle the Version 10 behavior for these functions, as described in the following steps.

To modify your applications to handle the Version 10 behavior for CHAR, VARCHAR, and CAST:

1. Identify applications that need to be modified to handle this change. You can use IFCID trace 0366 to identify affected applications.
2. Ensure that the BIF_COMPATIBILITY subsystem parameter is set to V9_DECIMAL_VARCHAR.

   To handle the change for the CHAR function only, you can set the subsystem parameter to V9 and complete the following steps for the CHAR function.
3. Change any affected applications to handle the new Version 10 CHAR and VARCHAR behavior, including stored procedures, non-inline user-defined

functions, and trigger packages. Rewrite affected CAST specifications with the appropriate CHAR or VARCHAR function and a CAST to the correct length if needed.

4. Rebind and prepare packages with the PATH(SYSCURRENT,SYSIBM) rebind option to use the new Version 10 CHAR and VARCHAR built-in functions. Repeat this step for native stored procedures (SQLPL) and non-inline SQL scalar functions.
5. For views that reference these casts or built-in functions, determine whether the view needs to be changed to have the expected output. Drop and re-create the views with the PATH(SYSCURRENT,SYSIBM) rebind option only if necessary. Rebind any applications that reference the views with the PATH(SYSCURRENT,SYSIBM) option to use the new Version 10 CHAR and VARCHAR built-in functions. Repeat this step for inline SQL scalar functions.
6. For materialized query tables or indexes on expressions that reference these casts or built-in functions, drop and re-create the materialized query tables or indexes on expressions with the PATH(SYSCURRENT,SYSIBM) rebind option. Issue the REFRESH TABLE statement for materialized query tables. Rebind any applications that reference the materialized query tables or indexes on expressions with the PATH(SYSCURRENT,SYSIBM) option to use the new Version 10 CHAR and VARCHAR built-in functions.
7. Change the value of the BIF_COMPATIBILITY subsystem parameter to CURRENT. When the subsystem parameter value is CURRENT, new applications, rebinds, and CREATE statements use the new CHAR, VARCHAR, and CAST behavior.

Materialized query tables and indexes on expressions use the CHAR, VARCHAR, and CAST behavior that is specified during its creation. If a reference statement has a different behavior that is specified by the BIF_COMPATIBILITY parameter or a different path, the materialized query table or index on expression is not used.

Related reference:

"BIF COMPATIBILITY field (BIF_COMPATIBILITY subsystem parameter)" on page 456

Change to maximum result length of VARCHAR function

In Version 10, the maximum result length of the VARCHAR function is changed from 32767 to 32704.

Changes to VARCHAR_FORMAT function length attribute

In Version 10, for VARCHAR_FORMAT functions, the length attribute of the result is the length attribute of the format string, up to a maximum of 255. To apply this change, use the following guidance:

- Existing view definitions that reference the VARCHAR_FORMAT function should be regenerated with an ALTER VIEW statement.
- Existing materialized query statements that reference the VARCHAR_FORMAT function should be dropped and re-created.
- Bound SQL statements that reference the VARCHAR_FORMAT function will only use the new behavior when they have been bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- DESCRIBE statements will only determine the result data type for the VARCHAR_FORMAT function using the modified rules in the DESCRIBE statements have been bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.

- Existing indexes that involve an expression that reference the VARCHAR_FORMAT function should be dropped and re-created.

If an application is using the DSNTIAUL program, the result string is padded with characters '00'X. Consider this incompatible change for any applications that use the VARCHAR_FORMAT function and are dependent on the output from DSNTIAUL.

New format element for VARCHAR_FORMAT function

In Version 10, a new format element, “RRRR”, is supported. In previous versions, this format element was interpreted as two adjacent specifications of the “RR” format element.

Change to result of VARCHAR_FORMAT function with “HH12” format element

In Version 10, if the “HH12” format element is specified in a VARCHAR_FORMAT function and the time component of the first argument is 24:00:00, the input timestamp value is adjusted to 00:00:00 and the date is changed to the next day. In previous releases, the timestamp value is adjusted to 12:00:00 and the day is not changed.

Change to result of VARCHAR_FORMAT function with “J” format element

In Version 10, if the “J” format element is specified in a VARCHAR_FORMAT function, the result is different from the result of the JULIAN_DAY function for dates earlier than October 15, 1582.

New supported data types for VARCHAR_FORMAT function

The VARCHAR_FORMAT function has been extended to allow date, character, and graphic string input for the first argument, and graphic string input for the second argument. If the function is invoked with one of the newly supported data types, and an existing user-defined function named VARCHAR_FORMAT also supports the data type, the function might resolve to the built-in function rather than the user-defined function. If the reference to the existing function uses the unqualified name and SYSIBM precedes the schema that was used for the user-defined function, the new function will be invoked rather than the user-defined function.

Change to maximum result length of REPEAT function

In Version 10, the maximum result length of the REPEAT function is changed from 32767 to 32704.

Change to maximum result length of XMLTABLE function

In Version 10, the maximum length of a VARCHAR data type result column of the XMLTABLE function is changed from 32767 to 32704.

Change to how a positive, signed integer in an ORDER BY clause is treated

Beginning in Version 10 conversion mode (from both Version 8 and Version 9.1), a positive, signed integer in an ORDER BY clause is treated as a *sort-key-expression*. This change might affect the result of queries that contain an ORDER BY clause with a positive, signed integer. Such integers were previously interpreted as column numbers.

Binding DBRMs directly into plans is no longer supported

For pre-existing plans that are bound from DBRMs, you can use the COLLID parameter of the REBIND PLAN command to create packages. If you execute a plan that is bound from DBRMs, DB2 performs an automatic rebind that creates

packages from the DBRMs and binds those packages into a plan. However, the recommendation is to use REBIND with the COLLID option so that you can specify bind options and receive more diagnostic information. If the installation uses the RACF access control module, owners of plans with DBRMs need to explicitly rebind the plans to convert the DBRMs to packages.

Some BIND PLAN and REBIND PLAN command options are no longer supported

The ACQUIRE(ALLOCATE) option of the BIND PLAN and REBIND PLAN commands is no longer supported. If you specify ACQUIRE(ALLOCATE), DB2 issues a warning message and uses ACQUIRE(USE).

Also, the MEMBER option of BIND PLAN and REBIND PLAN is no longer supported. If you specify MEMBER, DB2 issues a warning message, binds the specified DBRM into a package, and binds the package into a plan.

Plans and packages should be converted to DRDA protocol

Plans and packages that were previously bound using DBPROTOCOL(PRIVATE) should be converted to DRDA protocol before migration to Version 10. In Version 10, plans and packages that were bound with the DBPROTOCOL(PRIVATE) bind option and access remote locations cannot run. Applications that use packages or plans that were bound with DBPROTOCOL(PRIVATE) and access remote locations fail with SQLCODE -904. A rebind of those plans and packages must be explicitly performed before they can execute successfully. Job DSNTIJPM identifies the objects that must be converted to use DRDA protocol.

Change to GRANT statement

The PUBLIC AT ALL LOCATIONS clause is no longer allowed in the GRANT statement for table and view privileges as an alternative to PUBLIC. The DSNTPPCK program in Version 8 and Version 9.1 analyzes the embedded SQL statements in packages and plans for private protocol SQL, which is invalid in Version 10. The program produces a report that states which packages and member DBRMs of plans contain the invalid syntax. The program scans either the SYSIBM.SYSSTMT catalog table, the SYSIBM.SYSPACKSTMT catalog table, or both. Applications that issue dynamic SQL statements with the invalid PUBLIC AT ALL LOCATIONS clause will receive SQLCODE -199.

Change to IMMEDWRITE option of BIND PACKAGE command

In DB2 Version 10, if IMMEDWRITE is not specified on a BIND PACKAGE command, the default is “I”, or INHERITFROMPLAN. In previous versions, the default was NO.

Changes to conversion of special characters in collection IDs and package names

In Version 10, DRDA character type parameter data is sent between client drivers and DB2 in UTF-8 Unicode, if those client drivers also have this support. Character type parameter data includes package names and collection IDs. Prior to Version 10, collection IDs and package names were sent in EBCDIC, and then converted to Unicode before being stored in the DB2 catalog. A collection ID or package name that is in the DB2 catalog from a bind that was initiated by an older driver might not match an ID or name that is sent to DB2 by a newer driver. This mismatch, which is caused by the way that some special characters are converted, can cause package-not-found errors. Job step DSNTGEN of job DSNTIJPM identifies package names and collection IDs that contain special characters that cause mismatches. From the upgraded client drivers, those packages need to be bound with the ACTION(REPLACE) option.

Changes to the RELEASE bind option

In releases prior to Version 10, the RELEASE bind option had no effect on database access threads. Starting in Version 10, by default, DB2 honors the RELEASE bind option for database access threads. You can modify this behavior by using the new MODIFY DDF PKGREL command.

Database metadata stored procedures are converted to Unicode

In DB2 Version 10, DB2-supplied database metadata stored procedures are encoded in Unicode. The Version 10 migration process redefines the stored procedures to use the following new load modules:

Table 35. Metadata stored procedures and associated load modules

Stored procedure	Load module
SYSIBM.SQLCOLPRIVILEGES	DSNACPRU
SYSIBM.SQLCOLUMNS	DSNACOLU
SYSIBM.SQLFOREIGNKEYS	DSNAFNKU
SYSIBM.SQLFUNCTIONCOLS	DSNAFCOU
SYSIBM.SQLFUNCTIONS	DSNAFUNU
SYSIBM.SQLGETTYPEINFO	DSNATYPU
SYSIBM.SQLPRIMARYKEYS	DSNAPRKU
SYSIBM.SQLPROCEDURECOLS	DSNAPCOU
SYSIBM.SQLPROCEDURES	DSNAPRCU
SYSIBM.SQLSPECIALCOLUMNS	DSNASPCU
SYSIBM.SQLSTATISTICS	DSNASTAU
SYSIBM.SQLTABLEPRIVILEGES	DSNATBPU
SYSIBM.SQLTABLES	DSNATBLU
SYSIBM.SQLUDTS	DSNAUDTU

Some applications call the database metadata stored procedures to retrieve double-byte (DBCS) data. Those applications must be modified to use Unicode if they are bound on a DB2 server that has an EBCDIC SBCS CCSID, and the MIXED parameter is set to NO in the application defaults load module, *dsnhdecp*.

AUTHID is the default owner of packages that are bound by DSNTRIN

In previous releases, the stored procedures and user-defined functions that are provided as part of the DB2 base product (DB2-supplied routines) were created and bound by processing inline DDL and bind statements in DSNTIJSG and other installation jobs. By default, the ID that was used to run the job was also the authorization ID for creating the routines and the default package owner for those routines.

In Version 10, DB2-supplied routines are created and bound by running program DSNTRIN in job DSNTIJRT. The AUTHID parameter of DSNTRIN specifies the authorization ID for creating the routines. This ID is also the default owner of packages that are bound for those routines.

New default DEFINE attribute for dependent objects

As of Version 10, if the DEFINE attribute is not specified in the CREATE statement for explicitly created dependent objects (auxiliary indexes, XML indexes, and base table indexes), DB2 uses the DEFINE attribute of the base table space.

Exception: If the DEFINE attribute is not specified for explicitly created LOB table spaces, there is no correlation with the base table space until the auxiliary table is created. The DEFINE attribute is not inherited from the base, and the default is DEFINE YES.
Implicitly created dependent objects (base table indexes, LOB and XML table spaces, and their dependent indexes) inherit the DEFINE attribute of the base table space if it is DEFINE NO. Otherwise, if the base table space attribute is DEFINE YES, the value of the IMPDSDEF subsystem parameter is used for the dependent objects.

Change for creating partitioned table spaces

To create a classic partitioned table space in DB2 Version 10, you must specify SEGSIZE 0 and the NUMPARTS keyword of the CREATE TABLESPACE statement.

Change to default for CREATE TABLESPACE statements

In previous releases of DB2, if a CREATE TABLESPACE statement contains the NUMPARTS clause but neither the MAXPARTITIONS clause nor the SEGSIZE clause, a classic partitioned table space is created. Beginning in Version 10, the same statement results in a range-partitioned table space with a segment size of 32 by default. You might observe this difference when a subsequent CREATE INDEX statement with the specified ranges fails because index partitioning of a range-partitioned table space is not supported.

In response, you can change the default segment size through the DSN6SYSP.DPSEGSZ subsystem parameter, which is externalized as the DEFAULT PARTITION SEGSIZE field on panel DSNTIP7. The value of the DPSEGSZ parameter can range from 0 to 64 in increments of 4; for example, 0, 4, 8, and so on up to 64. The default is 32.

When DPSEGSZ is set to 0, a CREATE TABLESPACE statement that contains the NUMPARTS clause but neither the MAXPARTITIONS clause nor the SEGSIZE clause results in a classic partitioned table space, which is the behavior of previous releases. Note that the DPSEGSZ parameter is provided for compatibility only. It is deprecated in Version 10, and you should take steps to modify affected CREATE TABLESPACE statements or plan to use partition-by-growth table spaces instead of classic partitioned table spaces.

Change to default SEGSIZE value for universal table spaces

In DB2 Version 10, the default SEGSIZE value for universal table spaces has changed from 4 to 32.

Upgrade to supported COBOL and PL/I compilers

If you use the Version 10 precompiler, you must upgrade to COBOL compilers that DB2 10 for z/OS supports. See the DB2 Program Directory for information about supported compilers. The generated SQLCA and SQLDA for COBOL have changed. The generated attribute for binary data items is now COMP-5 instead of COMP-4 or COMP. Applications that are compiled on compilers that do not support the COMP-5 attribute no longer work.

For some COBOL and PL/I compilers that are no longer supported, you can use a version of the precompiler that allows you to precompile applications that have dependencies on these unsupported compilers. You can use this version of the precompiler with the following unsupported compilers:

- OS/VS COBOL V1.2.4
- OS PL/I 1.5 (PL/I Opt. V1.5.1)
- VS/COBOL II V1R4

- OS PL/I 2.3

The load module for this precompiler is DSNHPC7. This precompiler is meant only to ease the transition from unsupported compilers to supported compilers. This precompiler has the following restrictions:

- There is no corresponding DB2 coprocessor function to match this precompiler.
- The precompiler does not support SQL procedures.
- Only COBOL and PL/I are supported.
- The SQL flagger is not supported.
- The precompiler produces Version 7 DBRMs, and does not support any capability that is newer than Version 7.
- The application defaults module must be named DSNHDECP.

Support of this precompiler is deprecated in Version 10.

GRAPHIC and NOGRAPHIC SQL processing options are removed

If you specify the SQL processing options GRAPHIC or NOGRAPHIC, DB2 issues a standard warning message about an invalid option. These options are superseded by the CCSID SQL processing option.

SELECT FROM data change statements in BEFORE triggers no longer supported

The following statements are no longer allowed in the body of a BEFORE trigger:

- SELECT FROM DELETE
- SELECT FROM INSERT
- SELECT FROM MERGE
- SELECT FROM UPDATE

RETURN statement in scalar functions must follow *option-list*

In previous DB2 versions, RETURN statements in scalar functions could be in any order, relative to other clauses. After migration to Version 10 conversion mode (from Version 8 or Version 9.1), the RETURN statement in CREATE FUNCTION or ALTER FUNCTION statements must follow *option-list*. If a RETURN statement precedes *option-list* in one of these statements, DB2 issues SQLCODE -199.

Changed behavior of LOCATE_IN_STRING function

In Version 10, a negative value for *start* in the LOCATE_IN_STRING function results in the search starting at the end of the source string. When *start* is negative, the starting position is LENGTH(*source-string*) + *start* + 1.

Changes to ROUND_TIMESTAMP and TRUNC_TIMESTAMP functions

In Version 10, the ROUND_TIMESTAMP and TRUNC_TIMESTAMP functions return the first day of the first ISO week of the ISO year when an ISO year format is specified.

Also, whenever a CC or SCC format is specified, a start of a century is considered to be year 01.

Changes to result of NEXT_DAY function

In Version 10, the result data type for the NEXT_DAY function is determined from the input data. If the first input argument is a string, the result is TIMESTAMP(6) WITHOUT TIME ZONE. Otherwise, the data type of the result is the same as the

data type of the first input argument. For example, if the input is a date data type, the result is also a date data type. The following rules apply to the change in the NEXT_DAY function:

- Existing view definitions that reference the NEXT_DAY function should be regenerated with an ALTER VIEW statement.
- Existing materialized query statements that reference the NEXT_DAY function should be dropped and re-created.
- Bound SQL statements that reference the NEXT_DAY function only use the modified rules to determine the result data type if the statements are bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- DESCRIBE statements only use the modified rules to determine the result data type for the NEXT_DAY function if the statements are bound in Version 10 conversion mode (from Version 8 or Version 9.1) or later.
- Existing indexes that involve an expression that reference the NEXT_DAY function should be dropped and re-created.

Changes to MONTHS_BETWEEN function

In previous releases, the MONTHS_BETWEEN function ignored the time portion of arguments. In Version 10, the MONTHS_BETWEEN function takes the time portion of arguments into consideration.

Changes to TIMESTAMPDIFF function

In previous releases, the TIMESTAMPDIFF function allowed string input values that had more than six digits to the right of the decimal point. In Version 10, an error is issued if the string input value for TIMESTAMPDIFF function has more than six digits to the right of the decimal point.

Static SQL applications that use parallelism

DB2 incrementally rebinds the statements that use parallelism after migration to Version 10. Incremental rebinds can cause performance degradation, so you should manually rebind those statements. You can run a query in job DSNTIJPM before you migrate to determine which statements can use parallelism, and are therefore candidates for incremental rebinds. You should consider rebinding those statements after migration, as soon as your Version 10 system is stable. After you migrate to Version 10, you can also run a performance trace, class 3 or class 10 for IFCID 360, to identify the plans and packages that contain static SQL queries that use parallelism, and therefore need to be rebound.

Enforced SELECT authorization checking for UPDATE and DELETE statements

DB2 Version 10 checks for the SELECT privilege or appropriate administrative privilege before allowing a user to execute UPDATE or DELETE statements that reference an existing value in the target table. This authorization checking applies regardless of how the statements are executed (for example, bound in a package or executed in a dynamic statement). The authorization now also applies regardless of the setting of the SQLRULES(STD) bind option for static statements or the CURRENT RULES special register for dynamic statements. If the user does not have the necessary SELECT authorization or administrative privilege, a negative SQLCODE is returned.

Increased limit for work file record length

In Version 10 new-function mode, the limit for the row length in the result of a JOIN or the row length of a SORT record is increased from 32 767 bytes (1 page) to 65 529 bytes. The sort key maximum length is also increased from 16 000 bytes

to 32 000 bytes. Applications that exceed the old limits fail in conversion mode (from both Version 8 and Version 9.1) with SQLCODE -670 or SQLCODE -136.

New restrictions for EXPLAIN tables

In DB2 Version 10 conversion mode (from both Version 8 and Version 9.1), EXPLAIN tables must be in Version 8 or later format and preferably encoded in Unicode. When EXPLAIN tables are in a format prior to the Version 8 format, DB2 returns SQLCODE -20008 reason code 2 for statements or commands that invoke EXPLAIN processing. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9.1 format, regardless of the encoding type. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned. When you convert EXPLAIN tables to Unicode encoding, applications that join with EXPLAIN tables might have different results because of the CCSID conversion. For more information, see Objects with different CCSIDs in the same SQL statement (DB2 Internationalization Guide).

Recommendation: Before you begin migration to Version 10, convert all EXPLAIN tables to the current version (8 or 9.1) format and Unicode encoding, and then check for joins to those tables. You can use job DSNTIJXA to convert most tables to current release format. You can use jobs DSNTIJXB and DSNTIJXC to migrate EBCDIC-encoded EXPLAIN tables to Unicode.

MEMBER CLUSTER table spaces indicated by MEMBER_CLUSTER column

In previous versions of DB2, a “K” or “I” in the TYPE column of the SYSTABLESPACE catalog table indicated that the table space had MEMBER CLUSTER structure. In Version 10, a new MEMBER_CLUSTER column on the SYSTABLESPACE catalog table is populated during the enabling-new-function mode (from both Version 8 and Version 9.1) migration process. For existing MEMBER CLUSTER table spaces, values of “K” in the TYPE column of SYSTABLESPACE are replaced with “L”, and values of “I” are replaced with blank. The MEMBER_CLUSTER column is populated with “Y”. After migration to enabling-new-function mode (from Version 8 or Version 9.1), applications that query “K” or “I” in the TYPE column must query the new MEMBER_CLUSTER column instead.

Changed values for the modification level in the product signature

The DB2 Version 10 product signature has the form DSN1001*m*, where *m* is the modification level. Values 0 and 1 are reserved for maintenance levels in conversion mode from Version 8, conversion mode* from Version 8, enabling-new-function mode from Version 8, and enabling-new-function mode* from Version 8. Values 2 and 3 are for maintenance levels in conversion mode from Version 9.1, conversion mode* from Version 9.1, enabling-new-function mode from Version 9.1, and enabling-new-function mode* from Version 9.1. Values 5, 6, 7, 8, and 9 are for maintenance levels in new-function mode. Value 4 is undefined.

Changed behavior for the CREATE FUNCTION statement

In all forms of the CREATE FUNCTION statement, a parameter list is required. Functions without parameters must have empty parentheses specified, as in the following example: `CREATE FUNCTION F1 ( ) RETURNS INT RETURN 1`.

In previous releases, if you specified CREATE FUNCTION without a parameter list (for example, `CREATE FUNCTION F1 RETURNS INT RETURN 1`), you received no errors. In Version 10, if you specify CREATE FUNCTION without a parameter list, DB2 issues an error.

Also, a CREATE FUNCTION statement for a non-inline SQL scalar function cannot have a parameter that is a distinct type that is based on a LOB data type. The inline SQL scalar functions have supported, and will continue to support, distinct type parameters where the underlying base data type is a LOB. For the syntax of this statement, see CREATE FUNCTION in the DB2 for z/OS SQL Reference Guide.

Different SQLSTATE returned for some DELETE or UPDATE statements

In previous releases of DB2, a SQLSTATE value of "00000" was returned for some DELETE or UPDATE statements that did not contain a WHERE clause, when SQLWARN flags were set. Those situations occurred during bind processing. As of DB2 Version 10, a SQLSTATE value of "01504" is returned in those situations. This change does not affect SQL statements that are executed on behalf of a DB2 for z/OS requester.

Changed default behavior of multiple-row inserts for ODBC z/OS applications

As of DB2 Version 10, the default behavior for multiple-row inserts is changed from non-atomic to atomic. To change the default behavior back to non-atomic, set keyword PARAMOPTATOMIC = 0 in the data source section of the ODBC initialization file.

Changes to ALTER TABLESPACE statement error codes

Before Version 10 conversion mode (from Version 8 or Version 9.1), if you execute an ALTER TABLESPACE MAXPARTITIONS statement on catalog objects, DB2 issues SQLCODE -644. After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), DB2 issues SQLCODE -607.

Also before Version 10 conversion mode (from Version 8 or Version 9.1), if you execute an ALTER TABLESPACE MAXPARTITIONS statement with a value of MAXPARTITIONS that is invalid with the page size and DSSIZE values, DB2 issues SQLCODE -4701. After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), DB2 issues SQLCODE -644.

Change to CREATE and ALTER statements

CREATE and ALTER statements for native SQL procedures no longer support the DEFAULT keyword for the SQL PATH option.

Change to ALTER PROCEDURE statement

When the REGENERATE option is specified for the ALTER PROCEDURE statement for native SQL procedure, any existing comment in the catalog for the routine is retained. In previous DB2 versions, the comment was cleared in this situation.

Change to DESCRIBE statement

In Version 10, when a DESCRIBE statement is used on a result table that includes a distinct type, it will now return information about the distinct type.

New restrictions on using DSNTIAUL

DSNTIAUL can no longer be used to process CREATE FUNCTION (SQL scalar) statements that would result in a package or CREATE TRIGGER statements.

DSNTIAUL also cannot be used to process any other statement that contains *SQL-routine-body*. These statements are CREATE PROCEDURE (SQL external), CREATE PROCEDURE (SQL native), CREATE FUNCTION (SQL table), ALTER PROCEDURE (SQL native) with an ADD or REPLACE clause, and ALTER FUNCTION (SQL scalar) with an ADD or REPLACE clause.

Changes to SYSROUTINES

As of Version 10, the CREATEDTS column of SYSIBM.SYSROUTINES will always reflect the time that a CREATE statement was first issued for a routine. In previous releases of DB2, the CREATEDTS column might have contained different values if multiple versions of a routine were created.

Catalog restructured

In Version 10, the DB2 catalog is restructured to reduce lock contention. As a result, binding takes longer in Version 10 than it did in previous versions. Also, execution of the DECLARE GLOBAL TEMPORARY TABLE statement takes longer in Version 10.

Changed data type for an untyped parameter marker

In previous releases, an untyped parameter marker for a unary minus operator has an assumed DOUBLE data type. In DB2 Version 10, the assumed data type is DECFLOAT(34).

Changes to handling of special values Infinity, sNaN, and NaN

In previous releases, when DB2 returns a decimal floating-point (DECFLOAT) value for Infinity, NaN, or sNaN as a CHAR or VARCHAR string, the string is in mixed case. In DB2 Version 10, the values are returned in upper case as follows: INFINITY, NAN, or SNAN.

Changes for INSTEAD OF triggers

Changes are introduced for existing INSTEAD OF triggers that were defined on a view that has a ROWID column or a column that is based on an underlying column of any of the following types:

- A security label column.
- A row change timestamp column.
- A column that is defined with AS ROW BEGIN.
- A column that is defined with AS ROW END.
- A column that is defined with AS TRANSACTION START ID.

For such INSTEAD OF triggers, the following actions now fail with a negative SQLCODE:

- Rebinding the INSTEAD OF trigger package.
- Dropping and re-creating the INSTEAD OF trigger.
- Running the REPAIR DBD utility on a database that includes the INSTEAD OF trigger.
- Issuing ALTER TABLE ALTER COLUMN SET DATA type on a table that has a view on which the INSTEAD OF trigger is defined.

Change to positioned update or delete statements

Explanation

In DB2 Version 10, packages must be rebound if both of the following statements are true:

- The package contains static cursors that were bound in a previous version.

- The package contains dynamic UPDATE or DELETE statements that contain the WHERE CURRENT of clause. The WHERE CURRENT of clause indicates a positioned UPDATE or DELETE statement.

Possible impact to your DB2 environment

If any such packages are not rebound in Version 10, positioned UPDATE and DELETE statements fail with SQLCODE -20249 when they run against a down-level cursor.

Actions to take

After migration to Version 10 conversion mode (from Version 8 or Version 9.1), rebind packages that contain static cursors with positioned UPDATE or DELETE statements.

Change to stored procedure parameter values returned to non-Java clients

In previous releases, when a remote application calls a DB2 for z/OS stored procedure, the data types of the returned output data match the data types of the corresponding CALL statement arguments. Starting in Version 10 conversion mode (from both Version 8 and Version 9.1), the data types of the returned output data match the data types of the parameters in the stored procedure definition. This change can cause conversion failures for some applications that use non-Java client drivers such as .NET to call stored procedures on DB2. To prepare for this change, modify the CALL requests of your non-Java client applications to specify argument types that conform to the semantics of the client driver. If you want to temporarily override the Version 10 behavior, you can set the DDF_COMPATIBILITY subsystem parameter to SP_PARMS_NJV.

The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in Version 10, it will be removed in a later release of DB2.

Change to results of JDBC method PreparedStatement.setTimestamp

TIMESTAMP WITH TIME ZONE is first supported in Version 10 new-function mode. Under certain circumstances, an application that invokes `PreparedStatement.setTimestamp` might produce different results before and after conversion to Version 10 new-function mode. If the second parameter of `PreparedStatement.setTimestamp` has the `java.sql.Timestamp` data type, and the target data type is not known, the IBM Data Server Driver for JDBC and SQLJ chooses TIMESTAMP as the target data type before Version 10 new-function mode. However, starting with Version 10 new-function mode, unless the input value is 0001-01-01-00:00:00.000000 or 9999-12-31-23:59:59.999999, the driver chooses TIMESTAMP WITH TIME ZONE as the target data type. If the driver chooses the TIMESTAMP data type, and the target type is actually TIMESTAMP WITH TIME ZONE, the database manager sets the time zone in the target column using the value of the IMPLICIT_TIMEZONE DECP value. This value might differ from the value that is inserted prior to Version 10 new-function mode.

To produce the same results when `PreparedStatement.setTimestamp` is executed, specify a `com.ibm.db2.jcc.DBTimestamp` value as the second parameter.

Change to behavior of comma operator in XQuery path expression

A comma operator in a DB2 XQuery path expression results in SQLCODE -16031. In previous releases, a comma operator in the DB2 XQuery path expression predicate was interpreted as the `and` operator.

Change in how DB2 returns stored procedure output parameter data to remote clients

When an application on a client system calls a stored procedure on a DB2 10 for z/OS server, DB2 now handles the output parameters differently. Previously, DB2 returned stored procedure output parameters that were formatted according to the SQL type of the corresponding argument in the CALL statement. DB2 10 now returns output parameters that are formatted according to the SQL type of the corresponding parameter in the stored procedure declaration. This new behavior provides improved performance at the server by avoiding unnecessary server data conversions. Also, this new behavior is consistent with the existing server behavior for the return of query and select output data and it is consistent with the behavior of other DB2 family servers. Therefore, it provides applications with a more consistent and predictable interface to DB2.

In general, for applications that conform to client standards, this change has no impact to the calling application. In some cases, however, application changes might be needed. To prepare for this change, examine your remote applications that call DB2 stored procedures. If necessary, modify the CALL statements in your remote applications to specify argument data types that match the data types of the parameters in the stored procedure definitions.

When the change in behavior occurs

In general, the new behavior occurs after migration to DB2 10 for z/OS conversion mode from Version 8 (CM8) or conversion mode from Version 9.1 (CM9). However, the following exceptions apply:

- For applications using an IBM Data Server Driver for JDBC and SQLJ type 4 connectivity to access a DB2 10 data sharing group in CM8 or CM9, where the enableSysplexWLB client property is set to true:

  In such a configuration, the change in behavior occurs when the data sharing group is migrated to DB2 10 new-function mode (NFM). As long as the data sharing group is in CM8 or CM9 to support coexistence, DB2 uses the old behavior. The old behavior is used because the Java client driver might be caching data descriptors for outputs from the server. When the enableSysplexWLB property is set to true, the old behavior is maintained to ensure that the cached client descriptors are valid regardless of which member of the data sharing group is accessed.

  To prepare for this change in behavior while DB2 10 is in CM8 or CM9 to support coexistence, test your applications with the enableSysplexWLB client property set to false. Or, test the applications against a stand-alone DB2 10 server, if such a system is available for testing. The new DB2 10 behavior will be used for each of those applications even though the DB2 data sharing group is in CM8 or CM9.
- For applications using IBM Data Server Driver for non-Java clients (such as .NET), Version 9 or earlier:

  In such a configuration, the change in behavior occurs, by default, after migration to DB2 10 for z/OS CM8 or CM9. However, the change in behavior can be temporarily deferred. To temporarily override the DB2 10 behavior for returning stored procedure output parameter data to IBM Data Server non-Java

clients only (such as .NET), set the DDF_COMPATIBILITY subsystem parameter to SP_PARMS_NJV. The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in DB2 10, it will be removed in a later release of DB2. For more information about DDF_COMPATIBILITY, see Subsystem parameters that are not on installation panels (DB2 Installation and Migration).

Examples of application impacts

The following examples use IBM Data Server Java and .NET client drivers. Other drivers might behave differently. Because DB2 10 no longer converts data types when returning stored procedure output parameter data, there might be similar impacts to applications for other data types that are not discussed in the examples. If the calling application specifies arguments that are different from the declared parameter types, then the mismatch between the argument and parameter data types is handled by the client system, according to the client's programming model and the client's capabilities. If the client performs data type conversions such as for numeric data, the result of the conversion performed by the client might differ from the result that was previously returned when the DB2 server performed the conversion.

Example 1: A stored procedure parameter is declared as an INTEGER data type, but the application CALL statement specifies a SMALLINT argument.

- Behavior before DB2 10: DB2 converts the INTEGER data and returns a 2-byte SMALLINT value, which maps to an Int16 .NET data type.
- New behavior in DB2 10: DB2 returns a 4-byte INTEGER value, which maps to an Int32 .NET data type.

Sample application impact (.NET): If the application uses an IBM Data Server .NET client, the stored procedure CALL statement now fails with an invalid conversion error because the Int16 and Int32 .NET data types are not compatible with each other. The application must be changed to specify an Int32 argument in the CALL statement as required by the .NET programming model. Applications that are coded with an Int32 argument and call stored procedures that have an INTEGER parameter require no change.

The .NET programming model enforces strong data typing. Thus, even though the argument type and the parameter type are compatible SQL types, the .NET driver enforces strong data type checking according to the .NET semantics. Refer to the .NET documentation for information about .NET strong type checking.

Example 2: A stored procedure parameter is declared as a REAL data type, but the application CALL statement specifies a DOUBLE argument.

- Behavior before DB2 10: DB2 converts the output data to a DOUBLE value before returning it to the client.
- New behavior in DB2 10: DB2 returns the data to the client as a REAL value.

Sample application impact (Java): If an application that uses an IBM Data Server Java client uses the registerOutParameter() method to register the output parameter as a DOUBLE value before calling a stored procedure that has a REAL parameter, the following behavior occurs if the Java application uses the getObject() method to retrieve the output value:

- Before Version DB2 10, DB2 converted the REAL value to a DOUBLE value and returned it to the client. The Java driver returned the DOUBLE value directly to the application as the result of the getObject() method.

- Beginning in DB2 10, DB2 returns the REAL value to the client. The Java driver converts the REAL value to a DOUBLE value and returns it to application as the result of the getObject() method. APAR IC80974 for LUW clients and APAR PM58951 (JCC 3.63.131) and PM58952 (JCC 4.13.136) for z/OS clients are required in the IBM Data Server Java driver to ensure that the conversion is completed by the getObject() method. If the APAR fix is not applied, the Java application gets a ClassCastException.

Example 3: A stored procedure parameter is declared as a TIMESTAMP data type, but the application CALL statement specifies a VARCHAR argument.

- Behavior before DB2 10: DB2 converts the fixed-length TIMESTAMP data and returns a 26-byte VARCHAR value, containing a TIMESTAMP value in ISO format (*yyyy-mm-dd-hh.mm.ss*[.*ffffffff*]). The VARCHAR value maps to a String .NET data type.
- New behavior in DB2 10: DB2 returns a 26-byte fixed-length TIMESTAMP value, containing a TIMESTAMP value in ISO format. The TIMESTAMP value maps to a DateTime .NET data type.

Sample application impact (Java): If the application uses an IBM Data Server Java client, then an application change might be required, depending on what method the Java application uses to retrieve the parameter data.

- Before DB2 10, the getString() method previously returned TIMESTAMP data in ISO format, while the getTimestamp() method returned TIMESTAMP data in Java format
- Beginning in DB2 10, both the getString() and getTimestamp() methods return the TIMESTAMP value in Java format (*yyyy-mm-dd hh*:*mm*:*ss*[.*ffffffff*]).

An application change might be required if the Java application uses the getString() method and the application depends on receiving TIMESTAMP values in ISO format. Applications that use the getTimestamp() method require no change.

Similar considerations apply for TIME parameters. Beginning in DB2 10, DB2 returns TIME parameters as TIME data values in ISO format (*hh.mm.ss*), regardless of the SQL type of the corresponding argument in the CALL statement. For Java clients, the getString() method now returns TIME data in Java format (*hh.mm.ss*). If a Java application uses the getString() method and requires TIME data in ISO format, the application must be examined for possible changes.

Changes to datetime built-in functions

Explanation

Many datetime functions allow arguments containing *string representations of datetime values*. Valid formats for those strings are described in the DB2 SQL Reference.

In DB2 Version 10 conversion mode (from Version 8 or Version 9.1), the following additional string formats are allowed in the specified limited contexts:

- A string value of seven characters representing a date is allowed as an argument to the DATE function only.
- A string value of 8, 13, or 14 characters representing a point in time is allowed as an argument to the TIMESTAMP function only.

Possible impact to your DB2 environment

After migration to Version 10, applications that provide a seven-character string argument to represent a date for built-in functions other than the DATE function will return an error. Applications that provide a string value of 8, 13, or 14 characters to represent a point in time as an argument for built-in functions other than the TIMESTAMP function also return an error.

Actions to take

To prepare for this change, use valid string formats that are described in String representations of datetime values (DB2 SQL) in arguments for all of your datetime functions.

Qualify user-defined function names

If you use a user-defined function that has the same name as a built-in function that has been added to Version 10, ensure that you fully qualify the function name. If the function name is unqualified and "SYSIBM" precedes the schema that you used for this function in the SQL path, DB2 invokes one of the built-in functions.

For a list of built-in functions, including those that have been added in Version 10, see Functions (DB2 SQL).

SQLCODE changes

Some SQLCODE numbers and message text might have changed in DB2 Version 10. Also, the conditions under which some SQLCODEs are issued might have changed.

SQL reserved words

Version 10 has several new SQL reserved words. Refer to Reserved words (DB2 SQL) for the list of reserved words, and adjust your applications accordingly.

GUPI

Utility release incompatibilities for migration from Version 9.1

When you migrate from DB2 Version 9.1 to Version 10, be aware of the utility release incompatibilities.

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changes to RECOVER utility

For point-in-time recovery, the RECOVER utility now enforces the recovery of a base table space together with its associated auxiliary LOB or XML table spaces.

Changes to COPY CHANGELIMIT

The default value of the COPY CHANGELIMIT option has changed from 1,10 to 10. This new default value means that COPY creates image copies as follows:

- If the percentage of changed pages is greater than 0 and less than 10, COPY creates an incremental image copy.

- If the percentage of changed pages is greater than or equal to 10, COPY creates a full image copy.
- If no pages have changed, no copies are made.

If this behavior is not the outcome that you want for any COPY statements with the CHANGELIMIT default option, change those statements to specify appropriate CHANGELIMIT values.

Changes to RUNSTATS INDEX

In DB2 Version 10, the following changes are made to the RUNSTATS INDEX utility:

- The default SHRLEVEL value is changed from REFERENCE to CHANGE.
- The KEYCARD behavior is always in effect, and the KEYCARD keyword is deprecated. If you specify KEYCARD in the utility job statement, no error message is issued to indicate that the keyword is deprecated.

Changes to DSN1PRNT

In DB2 Version 10, when you run DSN1PRNT with the FORMAT EXPAND option, and the input data sets constitute a full image copy, the input data sets must contain all pages of the original table space, including dictionary pages.

Changes to DSNJU004

The output of the DSNJU004 (print log map) utility has changed. The output of the system-level backup BSDS record now includes the z/OS level of the backup and the catalog copy option. This information is collected by the BACKUP system utility in DB2 Version 10.

Changes to LISTDEF

After you migrate to DB2 Version 10, the LISTDEF utility supports the DEFINED YES and DEFINED NO options. By default, DEFINED YES is used, which means that only defined objects are included or excluded from LISTDEF lists. To include or exclude undefined objects in the LISTDEF, add the DEFINED NO or DEFINED ALL option.

In previous versions, if a LISTDEF list contains only undefined objects, DB2 issues message DSNU185I for each undefined object, and LISTDEF returns return code 0. In Version 10, if a list returns only undefined objects, and DEFINED NO or DEFINED ALL is not specified, LISTDEF issues DSNU1008 and returns return code 4, and DSNU185I messages are not issued.

DB2 Version 10 also adds the HISTORY option, which changes the behavior of the BASE, LOB, XML, and ALL keywords. When one of these keywords is specified and the HISTORY keyword is also specified, non-history objects are filtered out of the resulting list. When one of these keywords is specified and the HISTORY keyword is not specified, history objects are filtered out of the resulting list.

MAX_UTIL_PARTS subsystem parameter removed

As of DB2 Version 10, the MAX_UTIL_PARTS subsystem parameter is removed. Because the LOAD utility no longer uses MAX_UTIL_PARTS, there is no longer a limit on the number of partitions of a compressed table space that LOAD can process when the LOAD utility control statement does not contain the PART parameter.

Change to reorganization of partition-by-growth table spaces

In DB2 Version 10, the REORG TABLESPACE utility might leave a table space in COPY-pending status if the following are true:

- The table space is a partition-by-growth table space.
- The table space contains LOB columns.
- Rows are moved to a different partition or a new partition is added to the table space.

Command release incompatibilities for migration from Version 9.1

When you migrate from Version 9.1 to Version 10, be aware of the command release incompatibilities.

GUPI

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Change to DISPLAY GROUP DETAIL command

In DB2 Version 10, the output of the DISPLAY GROUP DETAIL command includes information about subgroups, if any exist.

Related reference:

-DISPLAY GROUP (DB2) (DB2 Commands)

Change to REBIND PACKAGE command

In previous releases, the REBIND PACKAGE command did not preserve the old package copies by default. In Version 10, old package copies are saved when REBIND PACKAGE is used. To change this default behavior, change the value of the PLANMGMT subsystem parameter to OFF.

GUPI

Related reference:

REBIND PACKAGE (DSN) (DB2 Commands)

PLAN MANAGEMENT field (PLANMGMT subsystem parameter) (DB2 Installation and Migration)

Security release incompatibilities for migration from Version 9.1

When you migrate from DB2 Version 9.1 to Version 10, be aware of the security release incompatibilities.

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Separate security and system administrator authorities

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, users with SYSADM authority can no longer manage security objects such as trusted contexts and roles, issue grants, or revoke grants that are made by others. Also, users with SYSCTRL authority can no longer manage roles, issue grants, or revoke grants that are made by others.

If you set SEPARATE SECURITY to NO, which is the default, the SYSADM and SYSCTRL authorities have the same privileges as in the previous release.

Changes to the sample connection routine (DSN3SATH)

In previous versions of DB2, DSN3SATH retrieved the list of secondary authorization IDs from the address-space-level ACEE for connection processing for utilities that run in batch mode. DSN3SATH included code to retrieve the list of secondary authorization IDs from a task-level ACEE, but that code was commented out. In DB2 Version 10, DSN3SATH has been changed to use the task-level ACEE in this case. This change lets utilities that run under stored procedures retrieve the list of secondary authorization IDs from a task-level ACEE.

Restriction for SET CURRENT SQLID

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, there is an additional restriction for the SET CURRENT SQLID statement. If any of the authorization IDs of the process has SYSADM authority, then CURRENT SQLID must be set to one of the authorization IDs of the application process.

Restriction for BIND PACKAGE OWNER and BIND PLAN OWNER

If the SEPARATE SECURITY field on the DSNTIPP1 panel is set to YES, there is an additional restriction for the BIND PACKAGE OWNER and the BIND PLAN OWNER. If any of the authorization IDs of the process has SYSADM authority or SYSCTRL authority, then the OWNER must be set to one of the authorization IDs of the process.

Additional role name restrictions

Previous releases prohibited using SYSADM, SYSCTRL, or PUBLIC as role names. As of Version 10, role names cannot be ACCESSCTRL, DATAACCESS, DBADM, DBCTRL, DBMAINT, NONE, NULL, PACKADM, PUBLIC, SECADM, SQLADM, or any name beginning with "SYS."

Online-updatable subsystem parameters

In previous versions of DB2, users with SYSOPR or higher authorities can update the online-updatable subsystem parameters. In DB2 Version 10, the security-related subsystem parameters can only be updated by users that have SECADM or installation SYSADM authority. The subsystem parameters that are affected by this change are:

- AUTHCACH
- BINDNV
- DBACRVW
- EXTSEC
- TCPALVER

Other release incompatibilities for migration from Version 9.1

When you migrate from DB2 Version 9.1 to Version 10, be aware of the release incompatibilities.

Plan for the following changes in Version 10 that might affect your migration.

Release incompatibilities that were changed or added since the first edition of this Version 10 publication are indicated by a vertical bar in the left margin. In other areas of this publication, a vertical bar in the margin indicates a change or addition that has occurred since the Version 9.1 release of this publication.

Changed default values for some subsystem parameters and installation panel fields

The default values for some subsystem parameters and installation panel fields have changed. The new values are listed in the table below.

Table 36. Subsystem parameters and installation panel fields with new default values

Parameter	Field	Panel	Version 9.1 default value	Version 10 default value
none	BP8K0	DSNTIP2	1000	2000
none	DATABASES	DSNTIPE	100	200
CACHEPAC	PACKAGE AUTH CACHE	DSNTIPP	100K	5M
CACHERAC	ROUTINE AUTH CACHE	DSNTIPP	100K	5M
CHECK_FASTREPLICATION	FAST REPLICATION (none in Version 9.1)	DSNTIP6	PREFERRED	REQUIRED
CHKFREQ	MINUTES/CHECKPOINT	DSNTIPL1	500000 records	5 minutes
CONTSTOR	CONTRACT THREAD STG	DSNTIPE	NO	YES
DB2SORT	UT DB2 SORT USE (none in Version 9.1)	DSNTIP61	DISABLE	ENABLE
DLDFREQ	LEVELID UPDATE FREQ	DSNTIPL1	5	ON
DSMAX	DSMAX	DSNTIPC	9960	20000
EDM_SKELETON_POOL	EDM SKELETON POOL SIZE	DSNTIPC	5120K	10240K
EDMDBDC	EDM DBD CACHE	DSNTIPC	11700K	23400K
EDMPOOL	EDMPOOL LIMIT BELOW THE BAR	DSNTIPC	18142	0[1]
EDMSTMTC	EDM STATEMENT CACHE	DSNTIPC	56693K	113386K
FLASHCOPY_PPRC	FLASHCOPY PPRC (none in Version 9.1)	DSNTIP6	blank	REQUIRED
IRLMRWT	RESOURCE TIMEOUT	DSNTIPI	60	30
IRLMSWT	TIME TO AUTOSTART	DSNTIPI	300	120
LRDRTHLD	LONG-RUNNING READER	DSNTIPE	0	10
MAXRBLK	RID POOL SIZE	DSNTIPC	8000K	400000K
MINSTOR	MANAGE THREAD STORAGE	DSNTIPE	YES	NO
MONSIZE	MONITOR SIZE	DSNTIPN	256K	1M
NUMLKTS	LOCKS PER TABLE(SPACE)	DSNTIPJ	1000	2000
PARA_EFF	PARALLELISM EFFICIENCY (none in Version 9.1)	DSNTIP8	100	50
PCLOSEN	RO SWITCH CHKPTS	DSNTIPL1	5	10
PLANMGMT	PLAN MANAGEMENT (none in Version 9.1)	DSNTIP8	OFF	EXTENDED
PRIVATE_PROTOCOL	none	none	YES	NO

Table 36. Subsystem parameters and installation panel fields with new default values (continued)

Parameter	Field	Panel	Version 9.1 default value	Version 10 default value
RRULOCK	U LOCK FOR RR/RS	DSNTIPI	NO	YES
SEQCACH	none	none	BYPASS	SEQ
SEQPRES	none	none	NO	YES
SRTPOOL	SORT POOL SIZE	DSNTIPC	2000K	10000K
STATIME	STATISTICS TIME	DSNTIPN	5	1
STATROLL	STATISTICS ROLLUP	DSNTIP61	NO	YES
URCHKTH	UR CHECK FREQ	DSNTIPL1	0	5
URLGWTH	UR LOG WRITE CHECK	DSNTIPL1	0	10K
UTSORTAL	UT SORT DATA SET ALLOCATION	DSNTIP61	NO	YES

1. Although the default value of EDMPOOL is 0, if you are migrating from Version 9.1, you should use the same value for EDMPOOL that you used in Version 9.1. For more information about setting this parameter, see "EDM LIMIT BELOW THE BAR (EDMPOOL subsystem parameter)" on page 493.

If the values that you specified for these parameters are lower than the new default values, you might want to increase your values.

Changed maximum values for some subsystem parameters

The maximum values for some subsystem parameters have changed. The new values are listed in the table below.

Table 37. Subsystem parameters with new maximum values

Parameter	Version 9.1 maximum value	Version 10 maximum value
CACHEPAC	5M	10M
CACHERAC	5M	10M
CTHREAD	2000	20000
DSSTIME	1440	60
IDBACK	2000	20000
IDFORE	2000	20000
MAXDBAT	1999	19999
MAXOFILR	2000	20000
MONSIZE	16M	64M
STATIME	1440	60

Changed acceptable values for the DLDFREQ parameter

The values 0 through 32767 for the DLDFREQ subsystem parameter are no longer supported in Version 10. The acceptable values for DLDFREQ in Version 10 are ON and OFF. A value of 0 in previous releases corresponds to a value of OFF in Version 10. Values 1 through 32767 in previous releases correspond to a value of ON in Version 10.

Changed minimum value for the MONSIZE parameter

The minimum value for the MONSIZE subsystem parameter has changed from 256K to 1M.

Changed minimum value for the OUTBUFF parameter

The minimum value for the OUTBUFF parameter has changed from 40K to 400K. If you specify a value less than 400K, the installation process changes the value to 400K. Check the region size in the startup procedure for the DB2 system services address space (DSN1MSTR) to ensure that it is large enough to accommodate the increased output buffer size. The recommended setting for the REGION size of the address space is 0M.

Changes to STATIME and SYNCVAL parameters

In DB2 Version 10, the STATIME and SYNCVAL subsystem parameters apply only to IFCIDs 0105, 0106, and 0199. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME and SYNCVAL, and the corresponding trace records are written at fixed, one-minute intervals.

Changed maximum degree of parallelism for parallel groups

DB2 Version 10 uses a changed maximum degree of parallelism when the subsystem parameter PARAMDEG is set to 0. To avoid storage shortages, the maximum degree of parallelism has been reduced from ten times the number of CPUs to two times the number of CPUs. If an increase in elapsed time for parallel queries is a problem, specify a value for PARAMDEG that is greater than two times the number of CPUs.

Resource limit facility changes

Before DB2 Version 10, resource limit facility (RLF) limits applied to connections from remote users. These limits applied, regardless of whether the primary or secondary authorization IDs of the users had installation SYSADM or installation SYSOPR authority. Starting with DB2 Version 10, no RLF limits apply to primary or secondary authorization IDs that have installation SYSADM or installation SYSOPR authority.

Changed behavior for DDF to use getaddrinfo() instead of gethostid() to obtain IP addresses

DB2 Version 10 is an IPv6 system and cannot use gethostid() to obtain IP addresses because gethostid() does not support IPv6. Instead, DDF uses getaddrinfo() to obtain IP addresses. getaddrinfo() resolves the host name that is obtained from gethostname() to one or more IP addresses that are specified in the DNS or local host file. Therefore, it is important that the TCP/IP host name that is obtained from gethostname() matches the host name that is specified as part of the domain name in the DNS or local host file, so that an IP address can be obtained.

For details about other methods of specifying an IP address, see “Starting TCP/IP support” on page 929.

DB2 Version 10 requires IRLM 2.3

IRLM 2.3 is delivered with DB2 Version 10. You must use the DB2-supplied IRLM procedure. Although IRLM 2.3 is required in DB2 Version 10, you can use the DXRRL183 load module from IRLM 2.2.

Changes to table space types

The default table space type for new tables is now the partition-by-growth universal table space. The partitioned table space type is deprecated. In new-function mode, you can use an ALTER statement to change single-table table spaces to partition-by-growth universal table spaces.

You cannot use an ALTER statement to change multiple-table table spaces to universal table spaces.

Altered catalog and directory table spaces

During processing in enabling-new-function mode (from both Version 8 and Version 9.1), some catalog and directory table spaces are altered to use row-level locking and some tables use reordered-row format. Changes to the structure of the catalog also include new table spaces, new tables, and the use of new LOB columns. As a result, backup and recovery for the catalog is substantially changed.

DSNTIJIC uses SHRLEVEL CHANGE to copy the DB2 catalog and directory

In DB2 Version 10, job DSNTIJIC is modified to specify SHRLEVEL CHANGE rather than SHRLEVEL REFERENCE. This modification allows other tasks to update the DB2 catalog and directory while DSNTIJIC is executing. If you want to restrict other tasks to only access the catalog and directory when DSNTIJIC is executing, modify your customized copy of DSNTIJIC to use SHRLEVEL REFERENCE.

Change to job DSNTIJMV

In DB2 Version 10, job DSNTIJMV is modified to use the keyword parameter format for the subsystem parameter control statement in IEFSSN*xx* members. In previous versions, job DSNTIJMV used the positional parameter format. The two parameter formats cannot be used together in the same IEFSSN*xx* member.

Example of keyword parameter format:

```
SUBSYS SUBNAME(ssname) INITRTN(DSN3INI)
  INITPARM('DSN3EPX,cmd-prefix<,scope<,group-attach<<,subgroup-attach>>>')
```

Example of positional parameter format:

```
ssname,DSN3INI,'DSN3EPX,cmd-prefix<,scope<,group-attach<<,subgroup-attach>>>'
```

Changes to pending states for ALTER statements

Before Version 10, some ALTER statements produce immediate definition changes, and the table space or index defined in a statement is placed in a restrictive REBUILD-pending (RBDP) or restrictive REORG-pending (REORP) state. In Version 10 new-function mode, these same ALTER statements are pending definition changes, and the table space or index defined in a statement is placed in an advisory REBUILD-pending (ARBDP) or advisory REORG-pending (AREOR) state. Changes are applied when the table space or index is reorganized or the index is rebuilt, and data in the table space or index can be accessed or updated.

SYSDATE added as synonym for CURRENT TIMESTAMP(0)

As of Version 10, SYSDATE is a synonym for CURRENT TIMESTAMP(0). After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), if an object named SYSDATE already exists, all references to the object should specify the name as a delimited identifier ("SYSDATE") to distinguish the object from the CURRENT TIMESTAMP(0) synonym. References to SYSDATE as an ordinary identifier result in SQLCODE -4700.

Added precision variable for CURRENT TIMESTAMP

As of Version 10, an optional timestamp precision can be specified for the CURRENT TIMESTAMP special register. The precision is the number of digits of fraction seconds in the timestamp. If a user-defined function with the name CURRENT_TIMESTAMP already exists, all references to the function should specify the name as a delimited identifier ("CURRENT_TIMESTAMP") after migration to Version 10 conversion mode (from Version 8 or Version 9.1). References to CURRENT_TIMESTAMP in a function call result in SQLCODE -4700.

Catalog and directory data sets converted to SMS-controlled

In DB2 Version 10, all data sets for the DB2 catalog and directory must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating data sets in extended format and to use extended addressability (EA). See member DSNTIJSS of the Version 10 SDSNSAMP library for a sample SMS environment. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

During enabling-new-function mode (from both Version 8 and Version 9.1), some existing table spaces are converted to standalone table spaces. The obsolete table spaces are deleted in enabling-new-function mode (from both Version 8 and Version 9.1), at which time the data sets for those table spaces are also deleted. For details on the table space conversion, see "Job DSNTIJEN" on page 655.

SMF subtype field defined as a two-byte value

The SMF subtype field should be defined as a two-byte value. Previously, DB2 mapped the field SM10xSTF as a single-byte field that corresponds to the high half of the standard two-byte field location. In Version 10, SM10xSTF is changed to a two-byte field. This change results in the removal of unused field SM10xRI from the mapping. This change might require change to applications that parse SMF.

Changed default OP buffer size

The online performance (OP) buffer size in Version 10 defaults to 1 MB unless otherwise specified.

Change to processing of IEEE BFP values

When an underflow IEEE BFP value is encountered during a remote connection or local processing, SQLCODE -802 is issued. In Version 9.1, no SQLCODE was issued in either case.

Changed sizes and definitions of trace records

DB2 Version 10 introduces changes to the definitions and sizes of some IFCID fields. These changes might require modifications to your trace applications.

For more information, see Changed IFCIDs in Version 10 (DB2 for z/OS What's New?).

Changed format of the IFCID 0002, 0003, 0148, and 0306 trace records

For IFCID 0002, the format of the distributed data facility area, which is mapped by QLST fields, has changed substantially. For IFCID 0003, the format of the distributed data facility area, which is mapped by QLAC fields, has changed substantially. For IFCID 0148, the format of the distributed accounting records has changed substantially. Any application that reads and interprets location statistics must be modified to account for the Version 10 format of IFCID 0002, 0003, and 0148. For IFCID 0306, the definition of field QW0306AD has changed and the size of the area mapped by field QW0306OF has increased. The first eight bytes of field QW0306AD are now "V10 " to indicate that the data is from DB2 Version 10 and that the Version 10 mapping should be used.

Change to connection and sign-on routines

Explanation

In previous releases, the EXPLSITE value that is passed to the DSN3@ATH connection and DSN3@SGN sign-on exit routines, relative to an upstream DB2 for z/OS requester system partner, could be either a network address or the partner's actual location name. In DB2 Version 10, the connection and sign-on exit routines are passed an EXPLSITE value that reflects a network address that is relative to the DB2 for z/OS requester system.

Possible impact to your DB2 environment

Because the connection and sign-on exit routines might base authentication decisions on the EXPLSITE value, you might have different authentication results in Version 10 than in previous releases.

Actions to take

To prepare for this change, examine the logic of your connection and sign-on exit routines. Determine if they have dependencies on the EXPLSITE value being relative to remote upstream DB2 for z/OS requester partner location names. If such dependencies exist, before migration to Version 10, modify the exit routine logic to accept EXPLSITE values that are either the network address or the actual location name for upstream DB2 for z/OS requester partners. After migration to Version 10 new-function mode, you can optionally modify the exit routines to remove dependencies on the EXPLSITE value being an actual DB2 for z/OS requester partner location name.

Functions that are deprecated for systems migrating from Version 9.1

During migration from DB2 Version 9.1, be aware of the functions that are deprecated in Version 10. Although they are supported in Version 10, support for these functions might be removed in a later release. Avoid creating new dependencies that rely on these functions, and if you have existing dependencies on them, develop plans to remove these dependencies.

The following functions are deprecated in Version 10.

Some DB2-supplied stored procedures

The following DB2-supplied stored procedures are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- SYSPROC.DSNACCOR - Consider using the SYSPROC.DSNACCOX stored procedure instead.
- SYSPROC.DSNAEXP - Consider using the EXPLAIN privilege in place of DSNAEXP to explain statements for users that do not have the authority to execute EXPLAIN statements.
- SYSPROC.DSNTBIND - Consider using the SYSPROC.ADMIN_COMMAND_DSN stored procedure instead.
- SYSPROC.DSNUTILS - Consider using the SYSPROC.DSNUTILU stored procedure instead.
- SYSPROC.DSNWZP - Consider using the SYSPROC.ADMIN_INFO_SYSPARM stored procedure instead.

Some subsystem parameters

The following subsystem parameters are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- DISABSCL: In later DB2 releases, subsystems behave as if DISABSCL is set to NO.
- DPSEGSZ: In later DB2 releases, subsystems behave as if DPSEGSZ is set to 32.
- OJPERFEH: In later DB2 releases, subsystems behave as if OJPERFEH is set to YES.
- OPTIOWGT: In later DB2 releases, subsystems behave as if OPTIOWGT is set to ENABLE.
- OPTIXIO: In later DB2 releases, subsystems behave as if OPTIXIO is set to ON.
- PTCDIO: In later DB2 releases, subsystems behave as if PTCDIO is set to OFF.
- RETVLCFK: In later DB2 releases, subsystems behave as if RETVLCFK is set to NO. Use of non-padded indexes is recommended.
- SEQCACH: In later DB2 releases, subsystems behave as if SEQCACH is set to SEQ.
- SEQPRES: In later DB2 releases, subsystems behave as if SEQPRES is set to YES.
- SMSDCFL: In later DB2 releases, you can use the SMS data class parameter with CREATE STOGROUP and ALTER STOGROUP statements instead.
- SMSDCIX: In later DB2 releases, you can use the SMS data class parameter with CREATE STOGROUP and ALTER STOGROUP statements instead.
- STATCLUS: In later DB2 release, subsystems behave as if STATCLUS is set to ENHANCED.

Acceptable value for DDF_COMPATIBILITY subsystem parameter

The SP_PARMS_NJV option of the DDF_COMPATIBILITY subsystem parameter is deprecated. Although the option is supported in Version 10, it will be removed in a later release of DB2.

Some DB2-supplied user-defined functions

The following DB2-supplied user-defined functions are deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

- DB2XML.SOAPHTTPC
- DB2XML.SOAPHTTPV
- SYSFUN.DSN_XMLVALIDATE (replaced by the DSN_XMLVALIDATE built-in function).

  During migration to Version 10 conversion mode from Version 9.1 or new-function mode, job DSNTIJRT issues a dummy ALTER statement on SYSFUN.DSN_XMLVALIDATE. This action causes all packages that call the user-defined function to be rebound to reference the built-in function.

  In a non-data sharing environment, set the ABIND subsystem parameter to YES. In a data-sharing environment, set the parameter to YES or COEXIST. These settings can trigger automatic rebinds during fallback in non-data sharing environments or when a Version 9.1 member executes a Version 10 package in data-sharing environments.

EXPLAIN tables in Version 8 or Version 9 format

In DB2 Version 10, EXPLAIN tables that are in a Version 8 or Version 9 format are deprecated. These tables should be converted to the Version 10 format. Starting in DB2 conversion mode (from both Version 8 and Version 9.1), statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9 format, regardless of the encoding type. For more information about converting your EXPLAIN tables to Version 10 format, see "Migration step 24: Convert EXPLAIN tables to the current format and encoding type" on page 608.

NEWFUN SQL processing options and DECP values

The SQL processing options NEWFUN(YES) and NEWFUN(NO) are deprecated and NEWFUN(V8), NEWFUN(V9), and NEWFUN(V10) options are added in Version 10. Although these deprecated options are supported in Version 10, you should use NEWFUN(V10) instead of NEWFUN(YES) and use NEWFUN(V9) instead of NEWFUN(NO).

The *dsnhdecp* parameter values NEWFUN=YES and NEWFUN=NO are also deprecated. Although these values are supported in Version 10, you should use NEWFUN=V10 instead of NEWFUN=YES and use NEWFUN=V9 instead of NEWFUN=NO.

Password protection for active log and archive log data sets

Support for password-protected active log and archive log data sets is deprecated. Although they are supported in Version 10, they will be removed in a later release of DB2.

Functions that are no longer supported for systems migrating from Version 9.1

If you are migrating to DB2 Version 10 from Version 9.1, be aware of the functions that are no longer supported.

The following functions are not supported in Version 10.

BookManager format is being discontinued

IBM is discontinuing the BookManager format as part of an ongoing effort to deliver the best possible information experience to our customers. IBM is optimizing the DB2 for z/OS information for viewing in the information center.

The BookManager books will continue to be provided for the life cycle of DB2 UDB for z/OS Version 8 and DB2 Version 9.1 for z/OS; however, IBM is not delivering BookManager books beginning in DB2 Version 10 for z/OS. The Information Management Software for z/OS Solutions Information Center includes all of the information that you are used to seeing in BookManager and PDF, plus many improved features for finding and viewing the information that you need to do your job.

msys for Setup DB2 Customization Center

In DB2 Version 10, msys for Setup DB2 Customization Center is removed. Instead, use the installation panels. For more information about using the installation

panels, see Chapter 4, “Tailoring DB2 jobs to your environment using the installation CLIST,” on page 219.

Optimization Service Center

As of DB2 Version 10, Optimization Service Center (OSC) is no longer supported. The recommended no-charge query optimization and service tool is IBM Data Studio, which is a query serviceability tool for DB2 for z/OS. This tool is based and built on the foundation of Optimization Service Center and Optimization Expert.

DB2 XML Extender

DB2 XML Extender is no longer supported. In DB2 Version 10, you can use the XML data type instead.

DB2 Management Clients feature

DB2 Management Clients feature, which includes DB2 Administration Server, DB2 Control Center, and DB2 Development Center, is no longer supported. Instead, use IBM Data Studio to design, develop, deploy, and manage your data-driven applications.

Some DB2-supplied routines

The following DB2-supplied routines are removed in DB2 Version 10 and are unavailable to callers after migration to conversion mode (from Version 8 or Version 9.1). A report is added to the DSNTIJPM premigration job to detect occurrences of these routines on an existing subsystem or data sharing group, and to specify that these routines are not available in Version 10.

- DSN8.DSN8EXP
- AMI-based DB2 MQ functions
  - DB2MQ1C.GETCOL
  - DB2MQ1C.MQPUBLISH
  - DB2MQ1C.MQREAD
  - DB2MQ1C.MQREADALL
  - DB2MQ1C.MQREADALLCLOB
  - DB2MQ1C.MQREADCLOB
  - DB2MQ1C.MQRECEIVE
  - DB2MQ1C.MQRECEIVEALL
  - DB2MQ1C.MQRECEIVEALLCLOB
  - DB2MQ1C.MQRECEIVECLOB
  - DB2MQ1C.MQSEND
  - DB2MQ1C.MQSUBSCRIBE
  - DB2MQ1C.MQUNSUBSCRIBE
  - DB2MQ2C.GETCOL
  - DB2MQ2C.MQPUBLISH
  - DB2MQ2C.MQREAD
  - DB2MQ2C.MQREADALL
  - DB2MQ2C.MQREADALLCLOB
  - DB2MQ2C.MQREADCLOB

- DB2MQ2C.MQRECEIVE
- DB2MQ2C.MQRECEIVEALL
- DB2MQ2C.MQRECEIVEALLCLOB
- DB2MQ2C.MQRECEIVECLOB
- DB2MQ2C.MQSEND
- DB2MQ2C.MQSUBSCRIBE
- DB2MQ2C.MQUNSUBSCRIBE
- DB2MQ1N.GETCOL
- DB2MQ1N.MQPUBLISH
- DB2MQ1N.MQREAD
- DB2MQ1N.MQREADALL
- DB2MQ1N.MQREADALLCLOB
- DB2MQ1N.MQREADCLOB
- DB2MQ1N.MQRECEIVE
- DB2MQ1N.MQRECEIVEALL
- DB2MQ1N.MQRECEIVEALLCLOB
- DB2MQ1N.MQRECEIVECLOB
- DB2MQ1N.MQSEND
- DB2MQ1N.MQSUBSCRIBE
- DB2MQ1N.MQUNSUBSCRIBE
- DB2MQ2N.GETCOL
- DB2MQ2N.MQPUBLISH
- DB2MQ2N.MQREAD
- DB2MQ2N.MQREADALL
- DB2MQ2N.MQREADALLCLOB
- DB2MQ2N.MQREADCLOB
- DB2MQ2N.MQRECEIVE
- DB2MQ2N.MQRECEIVEALL
- DB2MQ2N.MQRECEIVEALLCLOB
- DB2MQ2N.MQRECEIVECLOB
- DB2MQ2N.MQSEND
- DB2MQ2N.MQSUBSCRIBE
- DB2MQ2N.MQUNSUBSCRIBE

• MQSeries publish and subscribe XML routines
- DB2MQ1C.PUBLISHXML
- DB2MQ2C.PUBLISHXML

• MQSeries XML routines
- DMQXML1C.DXXMQGEN
- DMQXML1C.DXXMQGENCLOB
- DMQXML1C.DXXMQINSERT
- DMQXML1C.DXXMQINSERTALL
- DMQXML1C.DXXMQINSERTALLCLOB
- DMQXML1C.DXXMQINSERTCLOB
- DMQXML1C.DXXMQRETRIEVE
- DMQXML1C.DXXMQRETRIEVECLOB

- DMQXML1C.DXXMQSHRED
- DMQXML1C.DXXMQSHREDALL
- DMQXML1C.DXXMQSHREDALLCLOB
- DMQXML1C.DXXMQSHREDCLOB
- DMQXML1C.MQPUBLISHXML
- DMQXML1C.MQRCVALLXMLCLOB
- DMQXML1C.MQREADALLXML
- DMQXML1C.MQREADALLXMLCLOB
- DMQXML1C.MQREADXML
- DMQXML1C.MQREADXMLCLOB
- DMQXML1C.MQRECEIVEALLXML
- DMQXML1C.MQRECEIVEXML
- DMQXML1C.MQRECEIVEXMLCLOB
- DMQXML1C.MQSENDXML
- DMQXML1C.MQSENDXMLFILE
- DMQXML1C.MQSENDXMLFILECLOB
- DMQXML2C.DXXMQGEN
- DMQXML2C.DXXMQGENCLOB
- DMQXML2C.DXXMQINSERT
- DMQXML2C.DXXMQINSERTALL
- DMQXML2C.DXXMQINSERTALLCLOB
- DMQXML2C.DXXMQINSERTCLOB
- DMQXML2C.DXXMQRETRIEVE
- DMQXML2C.DXXMQRETRIEVECLOB
- DMQXML2C.DXXMQSHRED
- DMQXML2C.DXXMQSHREDALL
- DMQXML2C.DXXMQSHREDALLCLOB
- DMQXML2C.DXXMQSHREDCLOB
- DMQXML2C.MQPUBLISHXML
- DMQXML2C.MQRCVALLXMLCLOB
- DMQXML2C.MQREADALLXML
- DMQXML2C.MQREADALLXMLCLOB
- DMQXML2C.MQREADXML
- DMQXML2C.MQREADXMLCLOB
- DMQXML2C.MQRECEIVEALLXML
- DMQXML2C.MQRECEIVEXML
- DMQXML2C.MQRECEIVEXMLCLOB
- DMQXML2C.MQSENDXML
- DMQXML2C.MQSENDXMLFILE
- DMQXML2C.MQSENDXMLFILECLOB

Some subsystem parameters

The following subsystem parameters are removed in DB2 Version 10:

- EDMBFIT
- INLIST_PTC

- LOGAPSTG
- MAX_UTIL_PARTS
- OPTHYBCST
- OPTIXOPREF
- OPTOIRCPF
- PARTKEYU
- PREVALKEEP
- SJMISSKY
- XMLTABJPD

EXPLAIN tables in pre-Version 8 format and EXPLAIN tables in EBCDIC encoding

In DB2 Version 10, EXPLAIN tables in a pre-Version 8 format are no longer supported. Before migration to Version 10, these tables must be converted to the format of the current (Version 8 or Version 9) release. Starting in DB2 10 conversion mode (from both Version 8 and Version 9), DB2 returns SQLCODE -20008 for statements or commands that invoke EXPLAIN processing if EXPLAIN tables are in a pre-Version 8 format.

Also in Version 10, EXPLAIN tables that are encoded in EBCDIC are no longer supported. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned. However, for transition purposes, EXPLAIN tables that are in Version 8 or Version 9.1 format can be encoded in EBCDIC. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9.1 format, regardless of the encoding type. For more information about converting your EXPLAIN tables, see "Converting EXPLAIN tables (before migration)" on page 64.

Private protocol

The private protocol remote data access method is no longer supported as of Version 10. DRDA protocol must be used instead. Packages and plans that were bound with DBPROTOCOL(PRIVATE) in a previous release can exist in Version 10 and might run in Version 10. If an application that executes packages or plans that were bound with DBPROTOCOL(PRIVATE) only accesses objects that are local to the DB2 subsystem, the application should run successfully. If an application that executes packages that were bound with DBPROTOCOL(PRIVATE) accesses a remote location, the application fails.

Annotated XML schema decomposition using XDBDECOMPXML and XDBDECOMPXML100MB

Using the XDBDECOMPXML and XDBDECOMPXML100MB DB2-supplied stored procedures for annotated XML schema decomposition is no longer supported. Also, calling the SYSPROC.XSR_COMPLETE stored procedure with the **issuedfordecomposition** parameter set to 1 is not supported.

Instead of using annotated XML schema decomposition, you should use the XMLTABLE function to decompose XML documents. However, if you need to use XML schema decomposition in Version 10 conversion mode from Version 9.1, make the XML schema decomposition stored procedures available by concatenating the Version 9.1 STEPLIB. Or, copy DSNNVDCP and its alias names, DSNNVDCA and DSNNVDCB, to the Version 10 *prefix*.SDSNLOAD.

XML schema decomposition does not work in Version 10 new-function mode.

REORG TABLESPACE SHRLEVEL NONE on LOB table spaces

In DB2 Version 10 new-function mode, running the REORG TABLESPACE utility with the SHRLEVEL NONE option on a LOB table space is no longer supported. If SHRLEVEL NONE is specified, the utility immediately completes with return code 0 without performing the reorganization. SHRLEVEL REFERENCE or SHRLEVEL CHANGE should be used instead.

Monitoring and optimizing queries with profile tables

In DB2 Version 10, the ability to use profile tables to monitor and optimize queries is no longer supported. The DSN_STATEMENT_RUNTIME_INFO and DSN_OBJECT_RUNTIME_INFO tables are no longer supported. The MONITOR, MONITOR CPUTIME, MONITOR SPIKE, and MONITOR RLF profile keywords are also no longer supported.

Preparing for DB2 data sharing

To plan for the data sharing function of DB2 for z/OS, coordinate your efforts with system hardware and software groups in your company.

About this task

If you are preparing to migrate a member of a data sharing group or distributed environment to a new release, remember the following key points:

- Before you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), read the information about release incompatibilities and make any necessary changes.
- Make a plan to minimize the amount of time that members operate at different version levels. For example, reduce the amount of time that you keep some members at the Version 9.1 level while others are in Version 10.
- Before you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), ensure that maintenance through the Version 10 fallback SPE is applied to all started DB2 members. If the fallback SPE is not on all active group members, Version 10 does not start but issues a message. If you have quiesced members in your data sharing group, you do not need to apply the fallback SPE to the quiesced member.
- Start only one DB2 member for migration processing. During the migration, other group members can be active. However, other active group members may experience delays or timeouts if they attempt to access catalog objects that are locked by migration or enabling-new-function mode (from Version 8 or Version 9.1) processing. After migration completes on the first member, you can migrate the other data sharing group members.
- To prepare for possible fallback from Version 10 conversion mode (from Version 8 or Version 9.1), keep the subsystem parameter load module that is used by the version that you are migrating from.
- After all members have migrated to Version 10 conversion mode (from Version 8 or Version 9.1), remain in that mode until your data sharing group has processed a full range of typical work. The amount of time that a data sharing group needs to remain in Version 10 conversion mode (from Version 8 or Version 9.1) depends on the size of the data sharing group and the complexity of its typical work.

Related concepts:

Chapter 6, “Installing, migrating, and enabling DB2 data sharing,” on page 669
“Changes that might affect your migration from Version 8” on page 65
“Changes that might affect your migration from Version 9.1” on page 114

Related tasks:

“Migrating an existing data sharing group to a new release” on page 711

DB2 data sharing in a Parallel Sysplex

If you plan to use DB2 data sharing in a z/OS Parallel Sysplex, your system must meet some additional software and hardware requirements and connectivity needs.

Related reference:

z/OS MVS Setting Up a Sysplex

Parallel Sysplex components and requirements

DB2 data sharing is dependent on the hardware and software components in the Parallel Sysplex.

Coupling facility

Install and define at least one coupling facility to z/OS before enabling the DB2 data sharing function. For high availability, more than one coupling facility is required.

Data sharing member names from the subsystem parameter load module (DSNZPARM) are used to connect members to the coupling facility at DB2 startup. The first connector causes the list structure to be allocated in a coupling facility based on the preference list in the active CFRM policy.

Cross-system coupling facility component of z/OS:

During startup, the members of a data sharing group join one cross-system coupling facility (XCF) group, and their associated internal resource lock managers (IRLMs) join another XCF group.

The z/OS cross-system extended services (XES) also join an XCF group implicitly on behalf of the IRLM connection to the lock structure. To join a particular group, the data sharing group members and the IRLMs use the names that you specify during DB2 installation.

DB2 uses the XCF for certain intersystem communications. Use both the coupling facility and channel-to-channel connections for XCF signaling. See z/OS MVS Setting Up a Sysplex for more information about configuring the XCF.

Coupling facility structures:

DB2 relies on areas of storage in the coupling facility called *structures*.

Three types of structures exist: lock, list, and cache. Each structure type has a unique function. The following figure shows a sample configuration of the coupling facility structures that are used by DB2.

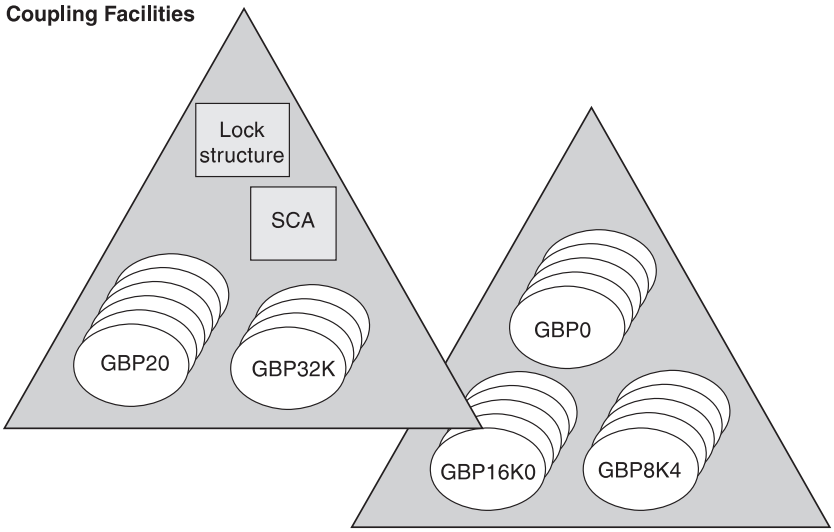


Figure 3. Coupling facility structures that are used by DB2. This is a sample configuration. The lock structure and list structure (SCA) do not need to be in the same coupling facility.

Members of a data sharing group use the following coupling facility structures to communicate and move data among themselves.

Lock structure

The lock structure protects shared DB2 resources (such as table spaces and pages) and enables concurrent access to those resources.

The system lock manager (SLM), a component of the z/OS cross-system extended services (XES), presents global lock information to the lock structure on behalf of each member's IRLM.

The lock structure consists of two parts: a coupling facility lock list table (called the *modified resource list*), and a coupling facility lock hash table (called the *lock table*). The modified resource list records locks that protect changed data, thereby protecting the data in case of failure. The lock table contains the lock status information and the owning members of those locks, and is used to provide global lock serialization.

List structure (SCA)

The list structure contains the DB2 shared communications area (SCA). Each member uses the SCA to pass control information to the rest of the members in the group. The SCA contains all database exception status conditions and other information that is necessary for recovery of the group.

Cache structures (DB2 group buffer pools)

The cache structures are used as group buffer pools (GBPs), caching shared data pages for the members. You have the following options for caching data of interest to more than one member of a group:

- Cache all data (read-only and updated)
- Cache only data that is updated
- Cache only system control pages (specific to LOB table spaces)
- Cache no data; use the group buffer pool only for cross-invalidation

Group buffer pools use a cross-invalidation mechanism to maintain data consistency across the buffer pools of group members. Shared data pages are registered in a group buffer pool directory in each cache structure, thus enabling the coupling facility control program to cross-invalidate the copies

of data pages that are held in individual member buffer pools. Cross-invalidation takes place when a member's own buffer pool does not contain the latest version of the data. In this case, the member must reread the pages from either the group buffer pool or disk, if that member needs to reference the pages again.

One group buffer pool exists for all member buffer pools of the same name. For example, each member must have a buffer pool 0 (BP0) that contains the catalog and directory table spaces. A group buffer pool 0 (GBP0) must exist on a coupling facility for that data sharing group.

Similarly, if a member creates table space X and associates it with buffer pool 1 (BP1), X is associated with BP1 for every member because there is only one definition of X in the catalog for the entire group. To share the data in X, you must define the cache structure, group buffer pool 1 (GBP1). If you do not define the group buffer pool, a single member can update X or more than one member can read X, but there can be no inter-DB2 read/write activity for X.

Recommendation: For data that is private to each member, such as work files or user data that only one member reads, define that data to a buffer pool for non-shared page sets. For example, assume that you want to associate all non-shared page sets with buffer pool 6 (BP6). If you want only member DB2A to access a non-shared table space Y, then define Y (and any indexes) to BP6. Define BP6 with a size of 0 (zero) and then you do not need to define the coupling facility structure for group buffer pool 6. By moving private data to buffer pools that are separate from buffer pools used by shared data, you can more easily monitor, and provide for more predictable performance of, private data.

Defining coupling facility structures:

Before you enable DB2 data sharing, you must define coupling facility structures.

About this task

You must define one lock structure, one list structure, and at least four cache structures:

- Group buffer pool 0
- Group buffer pool 8K0
- Group buffer pool 16K0
- Group buffer pool 32K

The lock structure and list structures (SCAs) do not need to be in the same coupling facility. Individual structures cannot span coupling facilities.

Recommendation: Use a Sysplex Failure Management (SFM) policy to define the availability characteristics of the coupling facility structures for lost connectivity failures, which includes a total failure of the coupling facility.

Procedure

To define coupling facility structures:

1. Determine the following characteristics of your DB2 coupling facility structures:

Initial size and maximum size of the structures
The structures can be dynamically resized from INITSIZE up to the value in SIZE.

Structure names
Determine the names that you will use for the coupling facility structures.

Availability characteristics
You must know the preference list (PREFLIST) for rebuilding or reallocating a structure, if the coupling facility fails.

2. Use the z/OS coupling facility resource management (CFRM) policies to define these structures to the Parallel Sysplex and define a SFM policy.

   A CFRM policy determines how and where the structure resources are allocated.

   See z/OS MVS Setting Up a Sysplex for information about how to create CFRM and SFM policies.

Example

A sample CFRM policy is shown in the following figure.

```
//POLICYX  JOB MSGCLASS=Z,REGION=2000K,CLASS=A,
//             MSGLEVEL=(1,1)
//STEP1  EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
 DATA TYPE(CFRM) REPORT(YES)
 DEFINE POLICY NAME(POLICYX) REPLACE(YES)
   STRUCTURE NAME(DSNDB0A_LOCK1)
             INITSIZE(32000)
             SIZE(64000)
             REBUILDPERCENT(5)
             PREFLIST(CF01,CF02)
   STRUCTURE NAME(DSNDB0A_GBP0)
             INITSIZE(50000)
             SIZE(100000)
             REBUILDPERCENT(5)
             DUPLEX(ALLOWED)
             PREFLIST(CF02,CF01)
   STRUCTURE NAME(DSNDB0A_GBP1)
             INITSIZE(50000)
             SIZE(100000)
             PREFLIST(CF02,CF01)
             DUPLEX (ENABLED)
   STRUCTURE NAME(DSNDB0A_SCA)
             INITSIZE(10000)
             SIZE(20000)
             REBUILDPERCENT(5)
             PREFLIST(CF01,CF02)

   CF NAME(CF01) TYPE(009674)
                 MFG(IBM)
                 PLANT(00)
                 SEQUENCE(000000040016)
                 PARTITION(1)
                 CPCID(00)
                 DUMPSPACE(1200)

   CF NAME(CF02) TYPE(009674)
                 MFG(IBM)
                 PLANT(00)
                 SEQUENCE(000000040029)
                 PARTITION(1)
                 CPCID(00)
                 DUMPSPACE(1200)
//
```

Figure 4. Sample CFRM policy

Related concepts:

"Coupling facility availability" on page 169

"Coupling facility structure names" on page 161

"Duplexed structures" on page 174

"Storage estimates for data sharing environments" on page 179

"How to change structure sizes" on page 190

"How structures are rebuilt when connectivity is lost" on page 172

Authorization for DB2 to access coupling facility structures:

Optionally, you can set up a facility class profile to limit access to the structures in the coupling facility.

If you do this, ensure that DB2 does have access by ensuring that the IDs that are associated with the DB2 address spaces have alter access authority to the coupling facility structures through RESOURCE(IXLSTR.*structure_name*) in SAF class CLASS(FACILITY).

If you do not create a facility class profile, the default allows any authorized user or program (supervisor state and program key mask allowing key 0-7) to issue coupling facility macros for the structure.

Sysplex timer:

Install at least one Sysplex Timer® in the Parallel Sysplex.

For high availability, more than one Sysplex Timer is required. The Sysplex Timer synchronizes the timestamps of the IBM System z9 or System z10® and z/Series servers for all members of the data sharing group. DB2 data sharing uses a value that is derived from the timestamp (as seen in the log) to recover data.

Common z/OS libraries:

DB2 supports a configuration with a SYS1.PARMLIB and SYS1.PROCLIB that is shared by all z/OS systems in the Parallel Sysplex. This configuration lets you add and modify systems more easily.

If you intend to have many members in the Parallel Sysplex, each DB2 and IRLM that you define to the z/OS system in the IEFSSN*xx* parmlib member **requires** a z/OS system linkage index (LX). The default number of these indexes that z/OS reserves is 165. If you place all of your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 165 LXs to start the members.

If you need more than 165 LXs, use the NSYSLX option on the z/OS IEASYS*xx* parmlib member to increase this number. See z/OS MVS Initialization and Tuning Reference for more information.

Related concepts:

"Naming recommendations for a DB2 data sharing environment" on page 162

Connectivity requirements in a Parallel Sysplex

DB2 data sharing requires that all DB2-related resources reside on shared disks. These resources include the DB2 catalog and directory and any user data, and the integrated catalog for DB2 data sets.

Also, all the members' logs and bootstrap data sets (BSDSs) must be on shared disks for recovery purposes. A member that performs recovery must have access to the logs of other members in the group.

Recommendation: Place work files on shared disks for the following reasons:

- For queries that are processed using Sysplex query parallelism, the placement of work files on shared disks is a requirement. Each assisting member writes to its own work file, and the coordinator can read the results from the assistants' work files.
- A member stays connected to its work file even if you need to restart the member on another processor.
- You can create or drop a work file table space from any other member in the data sharing group.

Ensure that you have physical connectivity by checking the following connections:

- Verify that one user-integrated catalog facility exists for cataloging the data sets of a data sharing group, and that you can access this catalog from each z/OS system in the Parallel Sysplex.
- Verify connectivity to the following entries from each system on which a member resides:
  - A set of DB2 target libraries
  - A single DB2 catalog
  - A single DB2 directory
  - All databases that are shared
  - All log data sets
  - All BSDS data sets
  - All coupling facilities used by the data sharing group
  - User integrated catalog facility catalogs for shared databases

Data sharing naming conventions

Some parts of a data sharing system require you to choose a valid name, while other names are generated during DB2 installation.

Carefully consider the naming convention you will use to name the various parts of the data sharing system. Assign names to both IRLM and data sharing groups, and to members within a group. One recommendation is to make names and prefixes unique within the Parallel Sysplex. Although this uniqueness is not required for all names, it helps you avoid problems with identifying and moving entities among z/OS systems in the Parallel Sysplex.

Related tasks:

"Renaming a member by performing an IPL" on page 740

"Renaming a member by stopping DB2" on page 741

Data sharing group names

Group names are shared by all members of the data sharing group. DB2 group names include the generic LU name, group attachment name, location name, and coupling facility structure names.

The following names are considered group names:

DB2 group name

The name that encompasses the entire data sharing group. The coupling facility structure names are based on this name.

The DB2 group name must be unique within the Parallel Sysplex. If you use this name as a basis for the location name, the DB2 group name must be unique within the network.

This name can be up to eight characters long, it must begin with an alphabetic character, and it can consist of the characters A-Z, 0-9, $, #, and @. An example of a DB2 group name is DSNDB0A.

Restrictions: To avoid names that IBM uses for its XCF groups, do not begin DB2 group names with the letters A-I unless the first three characters are DSN. Do not use the string SYS as the first three characters, and do not use the string UNDESIG as the group name.

Important: Never reuse a DB2 group name, even if a data sharing group that previously used the name no longer exists. Some data sharing information, such as the DB2 group name, is retained in the Parallel

Sysplex couple data set (CDS). To determine what DB2 group names exist, execute the z/OS command DISPLAY XCF,GROUP.

ICF catalog alias
The name of the Integrated Catalog Facility (ICF) catalog alias that you must place in the z/OS master catalog. This name can be up to 8 characters long, and it should be the same as the DB2 group name. An example of an ICF catalog alias name is DSNDB0A.

Group attachment name
The name that is used by the TSO/batch attachment, the call attachment facility (CAF), DL/I batch, utilities, and the Resource Recovery Services attachment facility (RRSAF) as a "generic" attachment name. This name can be up to four characters long. An example of a group attachment name is DB0A.

Subgroup attachment name
The name that is used by TSO, CAF, RRSAF, JDBC, ODBC, and DB2 utilities connections requests to identify which subgroup attachment to use. This name can be up to four characters long. An example of a subgroup attachment name is SBG1.

To specify subgroup attachment names, you must use Version 10 early code.

Catalog and directory database names
All DB2 subsystems in the group share the same catalog and directory. Examples of the catalog and directory database names are *catalias*.DSNDB01 and *catalias*.DSNDB06.

Application defaults module
The name of the module that contains the application programming defaults. This module resides in SDSNEXIT. The name can be up to eight characters long. DSNHDECP is an example module name.

Target libraries
Target libraries can be shared among DB2 group members or can be replicated. Example names for target libraries are *catalias*.SDSNCLST, *catalias*.SDSNLINK, *catalias*.SDSNLOAD, and *catalias*.SDSNEXIT.

Network Address
The group DVIPA and port number. It can be an IPv4 address, an IPv6 address, or both.

DRDA port number
The port number that is used by the distributed data facility (DDF) to accept incoming SQL connection requests, if the data sharing group uses TCP/IP. 446 is the recommended DRDA port.

Secure DRDA port number
The secure port identifies the TCP/IP port number that is used by DDF to accept inbound secure DRDA connection requests.

Location name
The name that is used when the data sharing group processes distributed requests. The group is treated as a single location by remote requesters. Requesters use this name in their SQL applications. This name can be up to 16 characters long.

Generic LU name
This name lets remote requesters configure their systems to treat the data sharing group as a single LU. This name can be up to eight characters long.

Parallel Sysplex domain name
This name lets you take advantage of workload balancing for TCP/IP connections.

Group IPv4 address
GRPIPV4 identifies a constant IPv4 address to be associated with the data sharing group for which this DDF is a member.

Group IPv6 address
GRPIPV6 identifies a constant IPv6 address to be associated with the data sharing group for which this DDF is a member.

Related concepts:

"Group attachment names and subgroup attachment names"

Communicating with data sharing groups (DB2 Data Sharing Planning and Administration)

Group access (DB2 Data Sharing Planning and Administration)

Specifying the DRDA port number (DB2 Data Sharing Planning and Administration)

Group attachment names and subgroup attachment names

Group and subgroup attachments are used by TSO, CAF, RRSAF, JDBC, ODBC, and DB2 utilities connections to find active DB2 subsystems.

A group attachment name acts as a generic name for all the members of a data sharing group. Subgroup attachment names can be used to specify a subset of members within a group attachment and provide more organization and control over connection requests. To specify subgroup attachment names, you must use Version 10 early code.

Specify the group attachment name and subgroup attachment name for a subsystem in member IEFSSN*xx*. You can let the DB2 installation process do this for you by specifying the names on installation panel DSNTIPK, or you can update the member yourself. If you use the DSNTIPK panel, the installation CLIST places the specified names in the IEFSSN*xx* member and in the DSNHDECP load module for the group. Specify the group attachment name and subgroup attachment name at a convenient time (during a planned IPL, for example).

The following rules apply to group attachment and subgroup attachment names:

- A subgroup attachment cannot have the same name as a group attachment.
- Neither the group attachment nor the subgroup attachment should have the same name as a DB2 member.
- If you specify a subgroup attachment name, you must also specify a group attachment name. A subgroup attachment must belong to one, and only one group attachment.
- A member can belong to one subgroup attachment at most and does not need to belong to a subgroup attachment.

The group attachment name and subgroup attachment name are included in the output for the command DISPLAY GROUP.

Even if you have not yet enabled data sharing, the group attachment name and subgroup attachment name is active after you IPL the system. An active group attachment name and subgroup attachment name in a non-data sharing environment is not a problem. Until you are ready to move to data sharing, continue to specify the DB2 subsystem name in your TSO and batch jobs. When you are ready to move to data sharing, you can change those jobs to specify a group attachment name and subgroup attachment name without the need for an IPL.

How DB2 chooses a subsystem name

When you submit a job on a z/OS system, DB2 treats the name that you specified on the DB2 connection request as a subsystem name, a group attachment name, or a subgroup attachment name. DB2 first assumes that the name is a subsystem name and attaches to that subsystem if either of the following is true:

- The subsystem is started.
- The subsystem is not started, and NOGROUP was specified in the DB2 connection request. NOGROUP indicates that group attachment or subgroup attachment processing is not to be considered. If RETRY was specified in the command, DB2 tries to attach the subsystem again in 30 seconds. The value of RETRY determines the number of times that DB2 re-attempts to attach.

DB2 assumes that the name on the DB2 connection request is a group attachment name or subgroup attachment name if no qualifying subsystem is found and either of the following is true:

- No subsystem with the name in the command is defined.
- A subsystem with that name is not started, the group attachment name or subgroup attachment name is the same as its subsystem name, and NOGROUP was not specified in the DB2 connection request.

When DB2 assumes that the name is a group attachment name or subgroup attachment name, it performs the following actions:

- Constructs a list of DB2 subsystems that are defined to this z/OS.

  To create the list, DB2 adds each subsystem during subsystem initialization. At IPL time, subsystems are initialized in the order in which they appear in member IEFSSN*xx*. If you add a subsystem with the z/OS SETSSI command, that subsystem is added to the list at that time.
- Tries to attach to the subsystems in the list in random order until it finds one that is started on this z/OS. However, if RANDOMATT=NO has been specified for a DB2 subsystem, that subsystem is excluded from the random search.
- Searches for any available DB2 subsystem within the group or subgroup, including those that specify RANDOMATT=NO, if the random search finds no available DB2 subsystems or if RANDOMATT=NO is specified for all subsystems. Search order is defined according to the order of z/OS subsystem initialization.

If the name on the DB2 connection request is not a group attachment name or subgroup attachment name, a “not started” message is returned.

If you need to prevent a batch workload from running on a DB2 subsystem that is intended for high-performance online transactions, specify RANDOMATT=NO for the online subsystems and RANDOMATT=YES for the batch workload subsystems. In that case, the batch jobs use the group attachment name or subgroup attachment name, and the online subsystem does not.

Alternatively, to achieve non-randomized group attachments or subgroup attachments, as in Version 8 of DB2, specify RANDOMATT=NO for all members of the group.

When a subsystem and a group attachment name or subgroup attachment name are the same

When you begin moving to data sharing, ensure that your IEFSSN*xx* definitions are correct and DB2 connection requests are coded to get the results that you intend. You should especially ensure that the definitions are correct when the group attachment name or subgroup attachment name is the same as a subsystem name. Incorrect IEFSSN*xx* definitions might be troublesome if you have inactive subsystems that are still defined but not used.

Assume you have the following subsystem definitions on a z/OS system:

```
DB2A,DSN3INI,'DSN3EPX,-DB2A,S'  ←Inactive subsystem
DB1A,DSN3INI,'DSN3EPX,-DB1A,S,DB2A'  ←Active subsystem
```

The jobs that are submitted on this z/OS system try to connect to the name DB2A. DB2 tries to connect to subsystem DB2A before considering group attachment processing. However, because DB2A is not started and it lacks a group attachment name, DB2 does not invoke group attachment processing and find DB1A as you might have intended. To avoid this situation, include the group attachment name in the definition of DB2A, or remove entry IEFSSN*xx* for subsystem DB2A if it is obsolete. With DB2A defined as the group attachment name for subsystem DB2A, DB2 tries to attach to DB1A after it discovers that DB2A is not started.

Alternatively, you might want a job to connect to a specific subsystem, but it connects to another subsystem in the group instead.

Assume that you have the following subsystem definitions on a z/OS system:

```
DB1A,DSN3INI,'DSN3EPX,-DB1A,S,DB1A'.  ←Inactive subsystem
DB2A,DSN3INI,'DSN3EPX,-DB2A,S,DB1A'.  ←Inactive subsystem
DB3A,DSN3INI,'DSN3EPX,-DB3A,S,DB1A'.  ←Active subsystem
```

Notice that DB1A is specified as a subsystem and also as the group attachment name for all three subsystems.

The jobs are submitted on this system with a DB2 connection request that specifies the name DB1A and omits the NOGROUP keyword. DB2 tries to connect to subsystem DB1A, but does not. Instead, DB2 invokes group attachment processing and eventually connects to DB3A, the first active subsystem with the group attachment name. You might have intended that the job connect only to DB1A. To ensure that DB2 connects to subsystem DB1A and no other subsystem, specify NOGROUP in the DB2 connection request to disable group attachment processing. Specify the RETRY keyword in the request to indicate the number of times, at 30-second intervals, that DB2 will retry to connect to DB1A.

Related tasks:

"Registering the command prefixes, member group attachment name, and subgroup attachment name" on page 195

Member names

Member names include the individual DB2 member (subsystem) name and its associated z/OS subsystem name, procedure names, and BSDS and log names. Member names must be unique within the data sharing group or, in certain cases, the z/OS Parallel Sysplex.

The following names are considered member names:

Member name
The name of an individual member of a data sharing group. A member name can be up to eight characters long, and it can consist of the characters A-Z, 0-9, $, #, and @. DB2 uses this name to form its z/OS cross-system coupling facility (XCF) member name. This name must be unique within the data sharing group. If the member name is also used as the high level qualifier for the member's data sets (BSDS, logs, and so on), member names must be unique within the z/OS Parallel Sysplex. This is because the z/OS Parallel Sysplex can have a shared master catalog. An example of a member name is DB1A.

DB2 subsystem name
The name of a DB2 subsystem that is a member of a data sharing group. This name is used by all the attachment interfaces. A DB2 subsystem name can be up to four characters long, and it must be unique within the Parallel Sysplex. The member name and the DB2 subsystem name should be the same. An example of a DB2 subsystem name is DB1A.

Location alias name
This name can represent one, several, or all members of a data sharing group. A location alias name enables you to define subsets of data sharing group members. Subsetting gives you the ability to limit the members to which DRDA requesters can connect.

Member LU name
The network name of an individual member of a data sharing group. A logical unit (LU) name must be unique both within the group and within the network. This name can be up to eight characters long.

Member domain name
This name lets DB2 handle indoubt thread resolution for TCP/IP connections.

Network Address
The member DVIPA and port number. It can be either an IPv4 address, an IPv6 address, or both. The TCP/IP port number that is used by the distributed data facility (DDF) to accept incoming DRDA two-phase commit resynchronization requests, if the data sharing group uses TCP/IP. Each member of a data sharing group listens on a unique resynchronization port.

Member IPv4 Address
A constant IPv4 address to be associated with the data sharing member.

Member IPv6 Address
A constant IPv6 address to be associated with the data sharing member.

Command prefix
The prefix that directs commands entered at a z/OS console to a particular member of a data sharing group. The command prefix can be up to eight characters long and the first character must be a special character. The default prefix is the concatenation of the hyphen character (-) with the DB2 subsystem name.

This string is specified as a parameter on the IEFSSN*xx* subsystem definition. An example of a command prefix is -DB1A. You can have blanks between the command prefix and the command.

Do not assign a command prefix that is used by another DB2 subsystem or that can be interpreted as belonging to more than one DB2 subsystem or z/OS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix starting from the first character. For example, it is invalid to assign a hyphen (-) to one DB2 subsystem and '-DB2A' to another DB2 subsystem. Similarly, it is invalid to assign '?DB2' to one DB2 subsystem and '?DB2A' to another DB2 subsystem. It is valid, for example, to assign '-DB2A' and '-DB3A' to different DB2 subsystems.

Work file database
The name of the work file database that is associated with a member of the data sharing group. Each member has its own work file database. In a non-data sharing environment, a DB2 subsystem's work file database is called DSNDB07. In a data sharing environment, although one member of the group can have a work file database with the name DSNDB07, you might want to create one with a more meaningful name, such as WRKDB1A, for member DB1A. This name can be up to eight characters long.

Work file data sets have names of the format *catalias*.DSNDBC.*mworkdb*.DSN*kknn*.*y*0001.A001, where *y* can be either I or J.

Restriction: You cannot specify a work file database name that begins with DSNDB unless the name is DSNDB07.

Load module for subsystem parameters
The name of the load module that contains a member's subsystem parameters. Each member has its own subsystem parameters. The load module resides in SDSNEXIT. The name is specified as a parameter on *mssn*MSTR procedure.

The load module name can be up to eight characters long. An example name is DSNZP01A, a naming convention that you can use to associate member DB1A with DSNZP01A.

Choosing names for member data sets: When choosing names for member data sets, remember that data set names beginning with *membname* must have a master catalog alias to point to the catalog where the data sets are cataloged. The DB2 installation process does not create this catalog alias. One way to handle this is to begin member data set names with *catalias* and a member-related qualifier. For example, member data set names could take the form *catalias.membname.xxxxx*. This format eliminates the need to have a master catalog alias for *membname*.

Member BSDS names
The names of bootstrap data sets (BSDS) belonging to a member of a data sharing group. The catalog alias can be added as the first qualifier. These names can be up to 33 characters long. Example BSDS names are DSNDB0A.DB1A.BSDS01 and DSNDB0A.DB1A.BSDS02.

Active log data set prefixes
The prefixes of active log data sets belonging to a member of a data sharing group. The catalog alias can be added as the first qualifier. These prefixes can be up to 30 characters long. Example active log data set prefixes are DSNDB0A.DB1A.LOGCOPY1 and DSNDB0A.DB1A.LOGCOPY2.

Archive log data set prefixes
The prefixes of archive log data sets belonging to a member of a data sharing group. The catalog alias can be added as the first qualifier. These prefixes can be up to 35 characters long unless you want the data sets timestamped. If they are timestamped, these prefixes can be up to only 19 characters long. Use the TIMESTAMP ARCHIVES parameter of installation panel DSNTIPH to specify whether you want the date and time qualifiers to be added to the archive log data set prefix.

Example archive log data set prefixes are DSNDB0A.DB1A.ARC1 and DSNDB0A.DB1A.ARC2.

SCA and group buffer pool connection names
The connection name is generated by DB2. You see it only in certain commands (z/OS D XCF,STRUCTURE and the DB2 connection list display of DISPLAY GROUPBUFFERPOOL). The connection name can be up to 16 characters long. An example connection name is DB2_*membername*.

Procedure names
The procedure names are generated from the member subsystem name at installation. The names can be up to eight characters long. Example procedure names are *mssn*MSTR, *mssn*DBM1, *mssn*DIST, and *mssn*SPAS.

XCF member name
This is the same as the member name for DB2; thus, a maximum of eight characters is used.

Related concepts:

Group access (DB2 Data Sharing Planning and Administration)

Related tasks:

"Choosing names for the local subsystem" on page 885

"Registering the command prefixes, member group attachment name, and subgroup attachment name" on page 195

Related reference:

"COMMAND PREFIX field" on page 423

IRLM names

Each member of a data sharing group has its own IRLM. The IRLM group name, subsystem name, and member ID are parameters on the IRLM startup procedure.

This means that every IRLM in the group must have a separate IRLM procedure. The following figure shows the relationship between DB2 group names and IRLM group names.

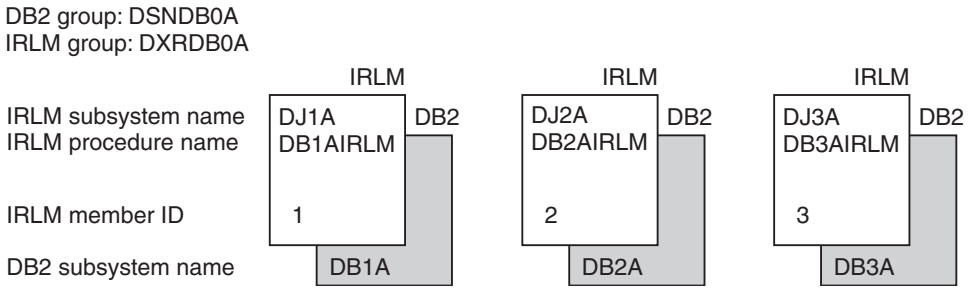


Figure 5. Relationship between DB2 and IRLM group names

You must choose the following IRLM names before installing DB2:

IRLM group name

The name that encompasses the entire IRLM group. This name can be up to eight characters long and can consist of the characters A-Z, 0-9, $, #, and @. The group name must begin with an alphabetic character. The name must be unique across the Parallel Sysplex.

Restriction: To avoid names that IBM uses for its XCF groups, do not begin with the letters A-I unless the first three characters are DXR. Do not use the string SYS as the first three characters, and do not use the string UNDESIG as your group name.

The IRLM group name is a parameter (IRLMGRP=) on each member's IRLM procedure. An example IRLM group name is DXRDB0A, a naming convention that you can use to associate member DB0A with DXRDB0A.

IRLM subsystem name

Each IRLM must have a subsystem name that must be unique within the data sharing group and can be up to four characters long and. This name is a parameter on the IRLM procedure.

A sample subsystem name is DJ1A. The “1A” characters indicate that this IRLM is paired with the member DB1A.

IRLM procedure name

Each member of a data sharing group knows its IRLM by the procedure and subsystem names saved in that member's installation parameter load module. The IRLM procedure name can be up to eight characters long.

Recommendation: Use the member subsystem name followed by “IRLM.” An example IRLM procedure name is DB1AIRLM.

IRLM member ID

This ID uniquely names an IRLM within a group. It is a parameter on the IRLM procedure and must be unique within the data sharing group. It is a number between 1 and 255 (inclusive).

The following names also apply to IRLMs:

IRLM XCF member name

This name can be up to 16 characters long. An example IRLM XCF member name is *xxxxxxxx$issnNNN*, where *xxxxxxxx* is the IRLM group name, *issn* is the IRLM subsystem ID, and *NNN* is the IRLM member ID. Dollar signs ($) are used as padding. This name is generated at startup time.

Lock structure contention name

The connection name is the same as the IRLM XCF member name. It is generated by IRLM. You see it only as the output of certain commands (such as z/OS D XCF,STRUCTURE). In some cases, the connection name can be of the format *xxxxxxxx#issnNNN*.

Related reference:

START irlmproc (z/OS IRLM) (DB2 Commands)

Coupling facility structure names

Names for coupling facility structures must conform to a strict naming convention that is based on the DB2 group name.

Sample names are shown in the following table:

Table 38. Sample names of coupling facility structures

Structure type	Example name
Cache structure (group buffer pools)	DSNDB0A_GBP0
Lock structure	DSNDB0A_LOCK1
List structure (shared communications area)	DSNDB0A_SCA

Group buffer pool names

Use a name of the following format on the CFRM policy to define the group buffer pool to the coupling facility. The name can be up to 16 characters long.

*groupname*_GBP*xxxx*

Where GBP*xxxx* is the name of the group buffer pool; the following restrictions apply:

- 4 KB group buffer pools are named GBP0, GBP1, ..., GBP49
- 8 KB group buffer pools are named GBP8K0, GBP8K1, ..., GBP8K9
- 16 KB group buffer pools are named GBP16K0, GBP16K1, ..., GBP16K9
- 32 KB group buffer pools are named GBP32K, GBP32K1, ..., GBP32K9

Requirement: You must define GBP0, GBP8K0, GBP16K0, and GBP32K. You must also define any other group buffer pools that you need for your particular data sharing setup.

When DB2 duplexes a group buffer pool structure, the same structure name is used for both the primary and secondary structures. A duplexed structure requires a single CFRM policy structure definition, with one structure name.

Lock structure name

Use a name of the following format on the CFRM policy to define the lock structure to the coupling facility. The name can be up to 16 characters long.

*groupname*_LOCK1

Shared communications area

Use a name of the following format on the CFRM policy to define the SCA to the coupling facility. The name can be up to 16 characters long.

*groupname*_SCA

Naming recommendations for a DB2 data sharing environment

Because you must name a large number of items, and you might have to add new members or move existing members to different systems in the future, managing all these items is easier if you choose and maintain a consistent naming convention.

You control the names that you assign to entities during the DB2 installation process. **After installation, you cannot change some names, such as the group name and the member names.**

If you are enabling the originating member of the group from an existing DB2 subsystem, you can build a naming scheme around existing names to reduce the disruption to existing applications. However, before enabling data sharing, you might want to change some names to lay the foundation for a solid naming convention.

Another name to choose carefully is the catalog alias for the group. It is very difficult to change that name. To change the catalog alias for the group, you must bring the entire group down and perform the single-system procedure for every member of the group.

Configuration assumptions

DB2 data sharing does not require that the Parallel Sysplex has a specific configuration. However, a specific configuration is assumed during the installation process. DB2 data sharing naming recommendations support and the installation process assumes the following z/OS Parallel Sysplex configuration:

- One SYS1.PARMLIB shared by all z/OS systems
- One SYS1.PROCLIB shared by all z/OS systems
- One integrated catalog facility master catalog shared by all z/OS systems

If the z/OS Parallel Sysplex is configured differently, you must customize the installation process. For example, if you use different SYS1.PARMLIBs, make sure that the DB2 data sets are in the APF list in each PARMLIB. If you use different PROCLIBs, modify the JCL to point to the correct libraries during installation. And, if you are using more than one integrated catalog facility (IFI) master catalog, put the DB2 catalog alias in each master catalog.

Naming convention suggestions

Consider the following suggestions when naming various DB2 entities:

- Name subsystems first.

  Subsystem names are limited to four characters, making them the shortest names. After you select subsystem names, use them as the basis for creating other names.

  Four-character names are also needed for the following entities:

  – DB2 group name; one per data sharing group
  – Member name; one for each DB2 subsystem in the group
  – IRLM subsystem name; one for each member of the group

  One possible naming convention is a Parallel Sysplex-wide approach to naming. This convention assigns subsystem names of the form "*ctmg*" where:

  c Denotes a particular collection of logically related applications or subsystems.

  t Denotes a type of resource, such as "B" for DB2 or "J" for IRLM.

  m Denotes a particular member within a data sharing group or an IRLM group. This identifier also associates a subsystem with its z/OS system for recovery when you use automatic restart.

  g Denotes a particular DB2 group.

  This information uses "A" as the group identifier. Therefore, the naming scheme has a subsystem name of DB1A for the first member of the data sharing group. The second member is DB2A, and so on. The following table shows how the member identifier relates to a particular z/OS system and how the group identifier associates members across a Parallel Sysplex system. In each column are the names that are associated with a particular z/OS recovery group (z/OS2, for example). In each row are the names that are associated with the primary components of the data sharing group, such as DB2 subsystems and IRLM subsystems.

Table 39. Subsystem names in a Parallel Sysplex

z/OS1	z/OS2	z/OS3	z/OS4
DA11 (CICS)	DA21 (CICS)	DA31 (CICS)	DA41 (CICS)
DB1A (DB2)	DB2A (DB2)	DB3A (DB2)	DB4A (DB2)
DJ1A (IRLM)	DJ2A (IRLM)	DJ3A (IRLM)	DJ4A (IRLM)
DD11 (DBCTL)	DD21 (DBCTL)	DD31 (DBCTL)	DD41 (DBCTL)

- If you enable an existing DB2 subsystem to take advantage of data sharing, and existing applications already access that DB2 subsystem, consider using the name of the existing DB2 subsystem as the group attachment name. This strategy allows existing applications continue to use the data sharing member. It does not require any changes to those applications in terms of specifying a new subsystem name.
- Use the same name for the DB2 group name, the DB2 location name, and the DB2 integrated catalog alias.

  A data sharing group uses the same catalog alias name. This catalog alias is used as the high-level qualifier for the DB2 directory (DSNDB01), catalog (DSNDB06), default database (DSNDB04), and work file database VSAM data sets.
- Use the same name for a member's member name and DB2 subsystem name.
- Accept the default prefix for a member's command prefix.

Example of names for a DB2 data sharing environment

The following table shows example names for a 12-member data sharing group. This example uses the subsystem naming convention that is described above. A zero in place of the member identifier is used to identify a group-wide resource, such as the group name or the catalog alias (DSNDB0A). The letter 'A' indicates that this naming convention is for data sharing group A. You can establish similar naming conventions for other data sharing groups, for example, group DSNDB0B. In the following table, the '#' character is used to denote a character from the set {1- 9,A,B,C}. These characters are used to denote a particular member's name.

Table 40. DB2 data sharing naming example

Entity	Name
DB2 group name	DSNDB0A
ICF catalog alias	DSNDB0A
Group attachment name	DB0A
Subgroup attachment name	SBG1
Catalog and directory database names	DSNDB0A.DSNDB01 and DSNDB0A.DSNDB06
Application defaults module	DSNHDECP
Target libraries	DSNDB0A.SDSNCLST DSNDB0A.SDSNLINK DSNDB0A.SDSNLOAD DSNDB0A.SDSNEXIT
Network Address	DOMAIN DSNDB0A.svl.ibm.com AND PORT 446
Group DRDA Port	446
Group Secure DRDA Port	448
Location name	STLEC1

Table 40. DB2 data sharing naming example (continued)

Entity	Name
Generic LU name	USIBMSY.SYEC1GLU
Parallel Sysplex domain name	DSNDB0A.svl.ibm.com
Group IPv4 Address	1.1.1.1
Group IPv6 Address	1.1.1.1.1.1.1.1
Member name	DB#A
DB2 subsystem name	DB#A
Location alias name	ALIAS1
Member LU name	LUDB2LA
Member domain name	DSNDB0A.svl.ibm.com
Network address	DOMAIN DSNDB0A.svl.ibm.com AND PORT 5001
Resync Port	5001
Member IPv4 Address	2.2.2.2
Member IPv6 Address	2.2.2.2.2.2.2.2
Command prefix	-DB#A
Work file database	WRKDB#A
Load module for subsystem parameters	DSNZP0#A
Member BSDS names	DSNDB0A.DB#A.BSDS01 DSNDB0A.DB#A.BSDS02
Active log data set prefixes	DSNDB0A.DB#A.LOGCOPY1 DSNDB0A.DB#A.LOGCOPY2
Archive log data set prefixes	DSNDB0A.DB#A.ARC1 DSNDB0A.DB#A.ARC2
SCA and group buffer pool connection names	DB2_DB#A
Procedure names	DB#AMSTR DB#ADBM1 DB#ADIST DB#ASPAS
XCF member name	DB0A
IRLM group name	DXRDB0A
IRLM subsystem name	DJ#A
IRLM procedure name	DB#AIRLM
IRLM member ID	Number 1-12 corresponding to #
IRLM XCF member name	DXRDB0A$$DJ#A12
Lock structure contention name	DXRDB0A$$DJ#A12
Cache structure (group buffer pools)	DSNDB0A_GBP0
Lock structure	DSNDB0A_LOCK1
List structure (shared communications area)	DSNDB0A_SCA

Distributed naming conventions

The example above does not include names for distributed processing. Those naming conventions will probably be part of a much broader convention. See your network administrator for more information about choosing names for distributed access.

Related tasks:

"Renaming a member by performing an IPL" on page 740

"Renaming a member by stopping DB2" on page 741

Planning for availability in a data sharing environment

When planning your data sharing configuration for the highest availability, the physical protection of the coupling facility and the structures within the coupling facility, as well as the highest network connectivity, are the primary concerns. The SCA and lock structure are both necessary for the group to function.

About this task

To plan for the highest availability:

Procedure

1. Duplex the group buffer pool to allow DB2 to switch to the secondary structure if the primary structure fails.

   Although the loss of a group buffer pool does not require a group restart, availability for users and important applications requires that data in a group buffer pool be available as quickly as possible after failure. Group buffer pools have several availability options, depending on the type of failure that occurs. Duplexing group buffer pool structures assures minimal impact on performance. Duplexing can also help you to avoid hours of recovery time, which are often required to recover simplexed group buffer pool structures.

   If a simplexed group buffer pool structure fails, the group buffer pool can be recovered automatically from data in the DB2 logs. If members lose connectivity to the group buffer pool, the group buffer pool can be rebuilt in another coupling facility to which the members can connect. Recovering data from the DB2 logs and rebuilding structures in another coupling facility can be very time-consuming.
2. Configure a simplexed SCA and lock structure in failure-isolated coupling facilities. (When duplexed, the SCA and lock structure do not need to be in failure-isolated coupling facilities.)

   Duplexing the SCA and lock structure offers marginally faster recovery times compared to dynamically rebuilding a simplexed SCA and lock structure. Duplexing also enables you to use internal coupling facilities (ICFs), which provide cost savings over stand-alone coupling facilities without compromising availability. However, before you decide to duplex the SCA and lock structure, you should consider that:

   - Duplexing the SCA and lock structure does not add significant availability, because the structures can be rebuilt very quickly from in-memory data.
   - Duplexing the SCA and lock structure can cause a significant impact on performance. In most cases, the overhead that is incurred by duplexing these structures outweighs the availability benefits of duplexing.
3. Define a dynamic virtual IP address (DVIPA) for the DB2 group and a DVIPA for each DB2 member in the group. DVIPA addressing gives you the ability to assign a specific virtual IP address to a data sharing group and to each member

of the group. This address is independent of any specific TCP/IP stack within the Parallel Sysplex. Even if a member is moved to another z/OS system, as in the case of a failure or maintenance, the member remains accessible and retains the same virtual IP address.

- To achieve the highest level of application availability at the database level, route connections using the DB2 Universal JDBC Driver or DB2 Connect Server with connection concentrator and Sysplex Workload Balancing support enabled. To route all connections through a DB2 Connect Server with Sysplex support enabled, you must first configure the DB2 group and each DB2 member to use DVIPA. The DB2 Connect Server then manages the connections to the group using the DVIPAs.
- To support JTA and JTS distributed transactions, the DB2 Universal JDBC Driver and the WebSphere Application Server can be used to coordinate commits against a data sharing group. If using the DB2 Universal JDBC Driver, group and member DVIPA need to be configured in order to recover indoubt threads after a system failure.

Automatic restart of z/OS

The purpose of automatic restart is to reduce the amount of time that a particular system is down.

When DB2 or IRLM stops abnormally, the surviving z/OS systems analyze the situation to determine whether z/OS failed too, and where DB2 or IRLM should be restarted. If z/OS did not fail, then DB2 or IRLM is restarted on the same z/OS image. If z/OS failed, then DB2 or IRLM is restarted on another z/OS image.

Automatic Restart Manager (ARM) is the component of z/OS that manages automatic restarts.

Advantage of automatic restart:

When DB2 and IRLM restart automatically, locks that are held by failed members are released quickly.

Applications that are running on other members can then access data for which the failed member is holding incompatible locks. Consider using automatic restart in conjunction with the RETAINED LOCK TIMEOUT option of installation panel DSNTIPI.

You must install DB2 with a command prefix scope of “started” to take advantage of automatic restart.

Related concepts:

“The scope and uniqueness of DB2 subsystem parameters” on page 671

Related tasks:

“Registering the command prefixes, member group attachment name, and subgroup attachment name” on page 195

Automatic restart policies:

You control how automatic restart works by using z/OS automatic restart policies.

When the automatic restart function is active, the default action (for both sharing and non-sharing DB2 subsystems and IRLMs) is to restart the subsystems when they fail. If this default action is not what you want, you must create a policy that defines the action that you want to be taken.

Creating an automatic restart policy:

If the default action of restarting DB2 and IRLM is the action that you want, you do not need to create an automatic restart policy.

Before you begin

If you do want to change the default restart behavior of DB2 or IRLM, you need to know the automatic restart element name.

In a data sharing group, the DB2 element name is the DB2 group name concatenated with the member name (such as DSNDB0ADB1A).

For IRLM, the element name is the IRLM group name concatenated with the IRLM subsystem name and three-character member ID (such as DXRDB0ADJ1A001).

You can also specify a pattern-matching character (such as DSNDB0A*) if you want to use a single policy statement for all members in the group.

About this task

DB2 startup can be a little faster when Automatic Restart Manager (ARM) restarts IRLM because this activity is done in parallel. DB2 does not have to start IRLM and wait. However, if you do not want IRLM to automatically restart after a failure, you can modify the IRLM element.

You can also tailor the restart policy to use restart light. Restart light enables DB2 to restart with a minimal storage footprint to quickly release retained locks and then terminate normally. It is not recommended for a restart in place, but it is recommended for a cross-system restart in the event of a failed z/OS system. It is primarily intended to restart DB2 temporarily on another z/OS system that does not have the capacity to sustain a DB2 and IRLM pair. ARM will not restart this member again when it comes down after performing a restart light.

Procedure

To create an automatic restart policy:

Create an ARM policy with one or both of the following specifications as needed for your system.

- To specify that DB2 or IRLM is not to be restarted after a failure:

  Include RESTART_ATTEMPTS(0) in the policy for that DB2 or IRLM element. For IRLM, you can also use the MODIFY irlmproc,ABEND command, as shown below, to stop IRLM and unregister it from ARM when it comes down. Unregistering prevents IRLM from automatically restarting after you bring it down.

  ```
  MODIFY irlmproc,ABEND,NODUMP
  ```

  However, if a member's AUTO START option of installation panel DSNTIPI has a value of YES, and if z/OS restarts DB2 automatically, DB2 restarts IRLM, too.
- To have DB2 restarted in a light mode (restart light):

  Modify the ARM policy for the DB2 group to specify LIGHT(YES) or LIGHT(NOINDOUBTS) within the RESTART_METHOD(SYSTERM) keyword for the DB2 element name. For example:

  ```
  RESTART_METHOD(SYSTERM,STC,'cmdprfx STA DB2,LIGHT(YES)')
  ```

Coupling facility availability

For high availability, you should have at least two coupling facilities, and at least one of them should be non-volatile.

With multiple coupling facilities, you can specify that structures be allocated in the secondary coupling facility, if the primary coupling facility is damaged. You can also consider duplexing SCA, lock, and group buffer pool structures. With duplexing, a secondary structure is always on standby in another coupling facility. This secondary structure is ready to take over if the primary structure fails or if a connectivity failure occurs. If you have three or more coupling facilities, you can even maintain duplexing while performing maintenance on one of the coupling facilities.

Duplexing the group buffer pool structures is a good idea because the performance overhead of duplexed group buffer pools are negligible in most cases, and the availability benefits are very high. Duplexing the SCA and lock structure is not as important for high availability because these structures can be rebuilt dynamically on an alternate coupling facility, if the coupling facility that contains the SCA and lock structure fails.

If the SCA and lock structure are not duplexed, the coupling facility that contains these structures should be failure isolated for the highest availability. A *failure-isolated CF* resides in a central processor complex (CPC) that does not also contain a data sharing member that is connected to structures in that coupling facility. If the SCA and lock structure reside in a *non-failure-isolated coupling facility* (a coupling facility that contains the SCA and lock structure and resides in a CPC that also contains a member of that data sharing group), the CPC becomes a single point of failure. If the CPC fails, the entire data sharing group comes down. Duplexing the SCA and lock structure, or keeping the SCA and lock structure in a failure-isolated coupling facility, avoids this single point of failure.

Related concepts:

"Duplexed structures" on page 174

Related reference:

Achieving the Highest Levels of Parallel Sysplex Availability

z/OS Parallel Sysplex Configuration Overview

Recommendations for placement of coupling facilities:

Physically separate the coupling facilities from each other and from the rest of the z/OS images that use those coupling facilities.

If you separate the SCA and lock structures from the systems that use them, you can minimize the chances of performing a lengthy group restart after a lengthy outage.

Quick recovery of the group buffer pools uses information in the lock structure and SCA to determine which databases must be recovered. This is known as *damage assessment*. Consider putting the lock structure and SCA in a coupling facility that does not contain important cache structures (such as group buffer pool 0). You are less likely to lose the SCA, lock structure, and the group buffer pool at the same time if you carefully separate these structures by placing them in different coupling facilities.

If you lose the lock structure or SCA at the same time as one or more group buffer pools, DB2 waits until the lock structure and SCA are rebuilt before doing damage assessment.

Preparations for coupling facility failures:

If structure failures occur, DB2 can recover resources in the structures.

The major types of coupling facility failures are structure failures, connectivity failures, and channel failures:

- A structure failure occurs when a group buffer pool, lock, or SCA structure no longer operates.
- A connectivity failure is a total failure of the coupling facility, such as a power failure to the coupling facility or some problem with the coupling facility control code.
- A channel failure occurs when a channel connecting a CPC to a coupling facility no longer operates.

Recommendation: To ensure high availability when preparing for structure and connectivity failures, configure the group buffer pool structures in duplex mode and the SCA and lock structures in simplex mode. To prepare for channel failures, consider using dual channels between each CPC and a coupling facility.

Duplexed group buffer pool structure failures:

If a duplexed group buffer pool structure fails, the failing structure is deallocated, and processing continues with the running structure.

This recovery is usually fast and unnoticeable. In case both structures fail, use checkpointing.

Related concepts:

"Simplexed group buffer pool structure failures"

How DB2 gathers checkpoint information (DB2 Data Sharing Planning and Administration)

Simplexed group buffer pool structure failures:

Simplexed group buffer pools can be recovered from the log when they fail.

Recovery from the log can occur manually, as the result of a START DATABASE command, or it can occur automatically because the group buffer pool is defined with the AUTOREC(YES) option. In either case, to reduce the time that is needed for group buffer pool recovery, use the ALTER GROUPBUFFERPOOL command to make group buffer pool checkpoints more frequent. However, weigh the benefit of faster recovery with the processing resources that are used for the checkpointing. (You can reduce checkpoint costs considerably when z/OS is at the appropriate level of maintenance.

Ensure that you specify one or more alternate coupling facilities in the CFRM preference list for the group buffer pools because a group buffer pool can be allocated in an alternative coupling facility when a new connection is made to it.

Related concepts:

How DB2 gathers checkpoint information (DB2 Data Sharing Planning and Administration)

Problem: Loss of group buffer pool structure (non-duplexed) (DB2 Data Sharing Planning and Administration)

Automatic recovery requirements:

Automatic recovery is faster than manual recovery because DB2 can optimize internal processing of the recovery.

For automatic recovery to be initiated for a group buffer pool, all of the following conditions must be true:

- AUTOREC (YES) is specified for the group buffer pool.
- At least one member is actively connected at the time that the group buffer pool failed. This member must have successfully completed damage assessment.

DB2 never initiates automatic recovery during restart.

For a duplexed group buffer pool, DB2 can use automatic recovery if both instances of the group buffer pool are damaged. Automatic recovery is not needed for group buffer pools that are defined as GBPCACHE (NO).

You can enable automatic recovery with the AUTOREC option of the ALTER GROUPBUFFERPOOL command.

Simplexed SCA and lock structure failures:

To recover a simplexed SCA and lock structure, DB2 uses information that is contained in its virtual storage to quickly rebuild the structures.

This recovery has a minimal impact on performance. DB2 can rebuild a simplexed SCA and lock structure in the same coupling facility or in an alternate coupling facility, assuming that the following conditions are true:

- You specified the alternate coupling facility in the CFRM policy preference list.
- You allocated enough storage in the alternate coupling facility to rebuild the structures there.

If DB2 fails to rebuild the SCA and lock structure from virtual storage, all active members in the group terminate abnormally, and you must perform a group restart to recover the necessary information from the logs.

Duplexed SCA and lock structure failures:

If a duplexed SCA or lock structure fails, the failing structure is deallocated, and processing continues on the running structure.

Important: To enable DB2 to switch to the secondary SCA and lock structure, the CFRM policy for the structures must indicate that duplexing is allowed, and the SCA and lock structure must currently be running in duplexed mode.

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

Preparations for connectivity failures:

DB2 and z/OS interpret a total failure of the coupling facility as a connectivity failure.

Connectivity failures include such issues as power failures to the coupling facility or a problem with coupling facility control code.

Related concepts:

Problem: loss of coupling facility (CF) (DB2 Data Sharing Planning and Administration)

Preparations for coupling facility channel failure:

To prevent a failure to DB2 that is caused by a coupling facility channel failure, consider using dual channels between each CPC and a coupling facility.

Without dual channels (sometimes called *links*), a channel failure is more likely to occur than a failure in the coupling facility. Losing connectivity to the SCA or lock structure can bring that particular member down, unless you specify duplexing or an alternative coupling facility in the CFRM policy preference list.

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

How structures are rebuilt when connectivity is lost:

As with structure failures, DB2 can recover quickly from connectivity failures if group buffer pool structures are duplexed, and if the SCA and lock structure are simplexed.

Recovering from connectivity failures with duplexed structures

DB2 recovers from connectivity failures with a duplexed group buffer pool, SCA, and lock structure by switching to the structure with good connectivity.

Recovering from connectivity failures with simplexed structures

DB2 recovers from connectivity failures with a simplexed group buffer pool, SCA, and lock structure as described below. Be aware that recovery of a simplexed SCA and lock structure is fast and has little or no impact on performance, but recovery of simplexed group buffer pool structures is very disruptive to the system.

When connectivity is lost, DB2 rebuilds simplexed structures on the alternate coupling facility that is specified in the CFRM policy. In rebuilding these structures, DB2 attempts to allocate storage on the alternate coupling facility. DB2 uses the current size of the structure for the initial size of the structure on the alternate coupling facility. If DB2 cannot allocate the storage for the SCA or lock structure, the rebuild fails. If z/OS cannot allocate the storage for the group buffer pools, the changed pages are written to disk.

Related concepts:

Failure scenarios (DB2 Data Sharing Planning and Administration)

Problem: a subset of members have lost connectivity (non-duplexed) (DB2 Data Sharing Planning and Administration)

Specifying when structure rebuilds occur after connectivity is lost:

The *rebuild threshold* is a percentage that you specify for the REBUILDPERCENT parameter in the CFRM policy. z/OS uses the REBUILDPERCENT value to determine whether to initiate a structure rebuild in the case of a loss of connectivity to the coupling facility that contains the structure.

About this task

The percentage is based on the Sysplex failure management (SFM) weights of all the systems that have active connections to a structure at the time. (A connection is considered active even if connectivity is lost.) You also specify weights on the SFM policy.

z/OS calculates the total weight of:

- All systems that have active connections but that lost connectivity to the structure (A)
- All systems that have active connections to the structure (B)

z/OS then divides A by B and compares the result to the REBUILDPERCENT value. (In the case of multiple connections to a structure from a single z/OS system, that system is counted only once. Multiple connections from a single z/OS system exist, for example, when two members reside on the same z/OS system.)

Procedure

To control when a rebuild occurs:

Specify REBUILDPERCENT values in your CFRM policy.

Recommendation: If you have high availability requirements, specify a small REBUILDPERCENT value, and allow the system to rebuild the structure. Specify a larger REBUILDPERCENT value only if you prefer to have a member lose the use of a structure rather than temporarily disrupt all of the members that were using the structure so that the structure can be rebuilt.

Example

Example: All z/OS systems in an eight-member Parallel Sysplex are of equal weight (10), and six of the systems have active connections to the group buffer pool. *A* equals the total weight of all systems that lose connectivity to the group buffer pool. *B* equals the total weight of all systems that have active connections to the group buffer pool. If two of the six systems lose connectivity, the value of *A* is 20, and the value of *B* is 60.

z/OS determines whether to initiate a rebuild as follows:

- If $(A/B) \times 100$ is greater than or equal to the REBUILDPERCENT value, DB2 rebuilds the structure.
- Otherwise, DB2 does not rebuild the structure. Instead, the affected members disconnect from the group buffer pool.

In this example, $(20/60) \times 100$ is equal to 33. If the value of REBUILDPERCENT is 35, z/OS does not initiate a rebuild.

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

Rebuild events:

You can monitor a performance class 20 trace (IFCIDs 0267 and 0268) to determine how long a rebuild of a structure takes and the reason for the rebuild.

You can also examine the messages that are returned to the console to monitor a rebuild. The reasons for a rebuild can include:

- Lost connectivity
- Operator command
- Duplexing is being established

Coupling facility volatility:

When a coupling facility is in a volatile state, data in the coupling facility is not saved in the event of a power failure.

DB2 issues a warning message if allocation occurs in a volatile coupling facility. A change in volatility after allocation does not affect existing structures.

If the coupling facility is configured to be non-volatile (using the proper power backup, such as a battery backup), volatility is generally a transient state that might occur, for example, if you take the battery out. If you lose power to a non-volatile coupling facility, the coupling facility enters power save mode and saves the data that is contained in the structures. When power is returned, you do not need to do a group restart nor recover the data from the structures. For systems that require high availability, non-volatile coupling facilities are recommended.

Related concepts:

When the coupling facility structure fails (z/OS MVS Setting Up a Sysplex)

Duplexed structures

Running some or all of the SCA, lock, and group buffer pool structures in duplex mode is one way to achieve high availability for these structures across many types of failures, including lost connections and damaged structures.

Recommendation: Although the SCA and lock structure can run in duplex mode, you should run them in simplex mode. When duplexed, the SCA and lock structure can cause a significant impact on performance. Also, because the SCA and lock structure can be quickly rebuilt, duplexing them does not add significant availability.

Important: If you decide to duplex the SCA and lock structure, you must duplex both structures to achieve the availability benefits. Duplexing only the SCA or only the lock structure does not provide any benefit.

How group buffer pool duplexing works:

With a duplexed group buffer pool structure, two allocations of the same structure use one connection from each member.

Each structure allocation must be in a different coupling facility. z/OS prefers to place the structures in coupling facilities that are failure-isolated from one another.

Recommendation: Ensure that at least one of the group buffer pool structures is in a non-volatile coupling facility. If power is lost to both coupling facilities and both coupling facilities are volatile, you must recover the group buffer pool from the logs.

Characteristics of primary and secondary structures:

z/OS commands let you stop and start duplexing, and let you choose which of the duplexed structures is the primary and which is the secondary.

The *primary structure* is the one from which changed data is cast out to disk. DB2 also uses the primary structure to read data, keep track of page-registration, and perform cross-invalidation of pages in the buffer pools of individual members. When changed data is written to the primary structure, it is also written to the *secondary structure*. DB2 uses the secondary structure as a backup. After data is cast out to disk from the primary structure, the data is deleted from the secondary structure.

From a z/OS perspective, duplexing is really an extended rebuild, so z/OS documentation and commands sometimes call the primary structure the *old* structure, and refer to the secondary structure as the *new* structure.

Coupling facility storage considerations for duplexing:

When planning for storage, make the primary and secondary group buffer pool structures the same size.

If your system is properly configured for high availability using a simplexed group buffer pool, you usually do not need extra coupling facility storage for duplexing. For simplexed structures, you must reserve enough spare capacity in the coupling facilities to be able to absorb the structures of any failed coupling facility. With duplexed structures, instead of reserving storage in case of a coupling facility failure, that storage is used by the secondary group buffer pool.

Example: Assume that you have 2 coupling facilities, each with 1 GB of memory, for a total of 2 GB of memory. Ensure that the total size of the structures across the 2 coupling facilities does not exceed 1 GB (50% of the total coupling facility storage).

If you configure three or more coupling facilities, you might need an additional coupling facility for duplexing.

Duplexing requirements:

To achieve high availability for SCA, lock, and group buffer pool structures across many types of failures, three conditions must be met.

For duplexing to work, the following conditions must be true:

- The CFRM policy preference list for the group buffer pool structure that you want to duplex must contain at least two coupling facilities with a CFLEVEL of 5 or higher. All members of the data sharing group must have physical connectivity to both coupling facilities in which the primary and secondary structures reside.

If you are going to do automatic reduplexing, you need three coupling facilities that are physically connected to members of the data sharing group.
- At least one member must be actively connected to the group buffer pool structure.
- The group buffer pool must be defined with GBPCACHE(YES), the default.

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

Options for duplexing:

Three options on the CFRM policy allow, enable, or disable duplexing.

You have three options on the CFRM policy for duplexing.
- DUPLEX(ENABLED) automatically starts duplexing.
- DUPLEX(ALLOWED) is not automatic, and you must issue a command to start duplexing.
- DUPLEX(DISABLED) disables duplexing.

Related tasks:

Starting duplexing for a structure (DB2 Data Sharing Planning and Administration)

Stopping duplexing for a structure (DB2 Data Sharing Planning and Administration)

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

Performance implications of duplexing:

The process of establishing duplexing can be somewhat disruptive because access to the group buffer pool structure is quiesced while the secondary structure is allocated. Also, changed pages are copied from the primary structure to the secondary structure (or cast out to disk).

Transactions that need access to the group buffer pool structure during this process are suspended until the process is complete. Because of this disruption, you need to establish duplexing at a time of low activity on the system. How long the process takes depends on how many pages are copied to the secondary group buffer pool structure.

In general, more processor and elapsed time is needed to do duplexed group buffer pool structure writes and castout processing than to do simplexed group buffer pool structure writes and castout processing. Workloads that are more update-intensive will probably experience a slight increase in host CPU usage when duplexing is activated. In most cases, the majority of the CPU increase occurs in the DB2 address space. Duplexing can cause a slight increase in the transaction elapsed time. Read performance is unaffected by duplexing.

The statistics and accounting trace classes contain information about structure duplexing.

Estimating the effect of duplexing on CPU usage:

When you enable duplexing, you will see an increase in the CPU usage in the coupling facility that contains the secondary structure. Duplexing should have little or no impact on the CPU usage in the coupling facility that contains the primary structure.

Procedure

To estimate approximately how much the coupling facility CPU usage will increase when you establish duplexing:

1. Determine the amount of coupling facility CPU usage that the simplexed primary structure consumes.
2. Divide the result in half to determine how much coupling facility CPU usage the duplexed secondary structure will consume.

Duplexing rebuild status:

When a group buffer pool structure is duplexed, it is considered to be in an extended rebuild status called a duplexing rebuild.

This activity is reported in IFCIDs 0267 and 0268, along with other reasons for rebuilding.

DB2 resource availability considerations

Several options for the catalog and directory, for data in group buffer pools, and for DB2 restart can be used for data that requires high-availability.

DB2 resource availability considerations for a data sharing group are basically the same as for a single DB2 subsystem.

Critical DB2 data

Critical DB2 data should be placed behind high-availability storage controllers, such as the IBM Enterprise Storage Server®. Consider placing the catalog and directory behind a 3990 control unit with dual-write capability for hardware duplexing. Another possibility is to use storage controllers with high-availability characteristics (such as controllers that use RAID technology) for critical DB2 data.

Group buffer pool data

Assign data that require high availability to group buffer pools that reside on non-volatile coupling facilities. Use group buffer pool duplexing, which has minimal impact on performance and can avoid hours of recovery time.

Data availability at restart

For faster restarts, take more frequent checkpoints. Checkpoint frequency is the most important factor in DB2 restart time.

If your installation sometimes has problems with units of recovery (URs) that take a long time to back out after a failure, make plans to reroute work to other members of the group. Another solution is to postpone backout processing for those long-running URs until DB2 is up and receiving new work.

Related concepts:

Postponed backout in a data sharing environment (DB2 Data Sharing Planning and Administration)

Implications of distributed systems on data sharing

WebSphere Application Server and the JCC Driver can be used to coordinate distributed transactions using XA protocols supported by DRDA.

In this configuration, you are required to set up dynamic virtual IP addresses. A dynamic IP address needs to be configured for the DB2 group, and one address must be set up for each DB2 member in the group. For XA recovery to be processed on any member after failure, information on indoubt XIDs must be obtained from the SCA. The SCA contains the list of indoubt XIDs for the entire group, as well as each member's DVIPA and the resync port of the member that owns the indoubt thread. XA recovery returns the list of all prepared or heuristically committed threads for the entire group back to the XA Transaction Manager (TM). The Java Universal Driver uses the information to send the commit decision to the owning member, when the XA TM recovers, and then commits or rolls back indoubt XIDs.

For the recovery process to function properly, you must set up the group DVIPA. the following figure illustrates how the XA recovery process, that is described above, works. After a system failure, XA recover requires the list of indoubt transactions identified by XA XIDs, to be returned to the group. In this example, if member DB1G fails, DB1G will leave an indoubt thread. The state of the thread, as well as related XIDs, is contained in the log of the DB1G member. The DB1G indoubt thread XID is also contained in the group's XA indoubt list, in the SCA. When WebSphere tries to recover from the system failure, the IBM Data Server Driver for JDBC and SQLJ sends the XA Recover command to any member of the group. The member then returns the group's list of indoubt XIDs from reading the XA indoubt list in the SCA. When WebSphere commits or rollbacks the indoubt thread, the IBM Data Server Driver for JDBC and SQLJ sends the decision to member DB1G, which resolves the indoubt thread.

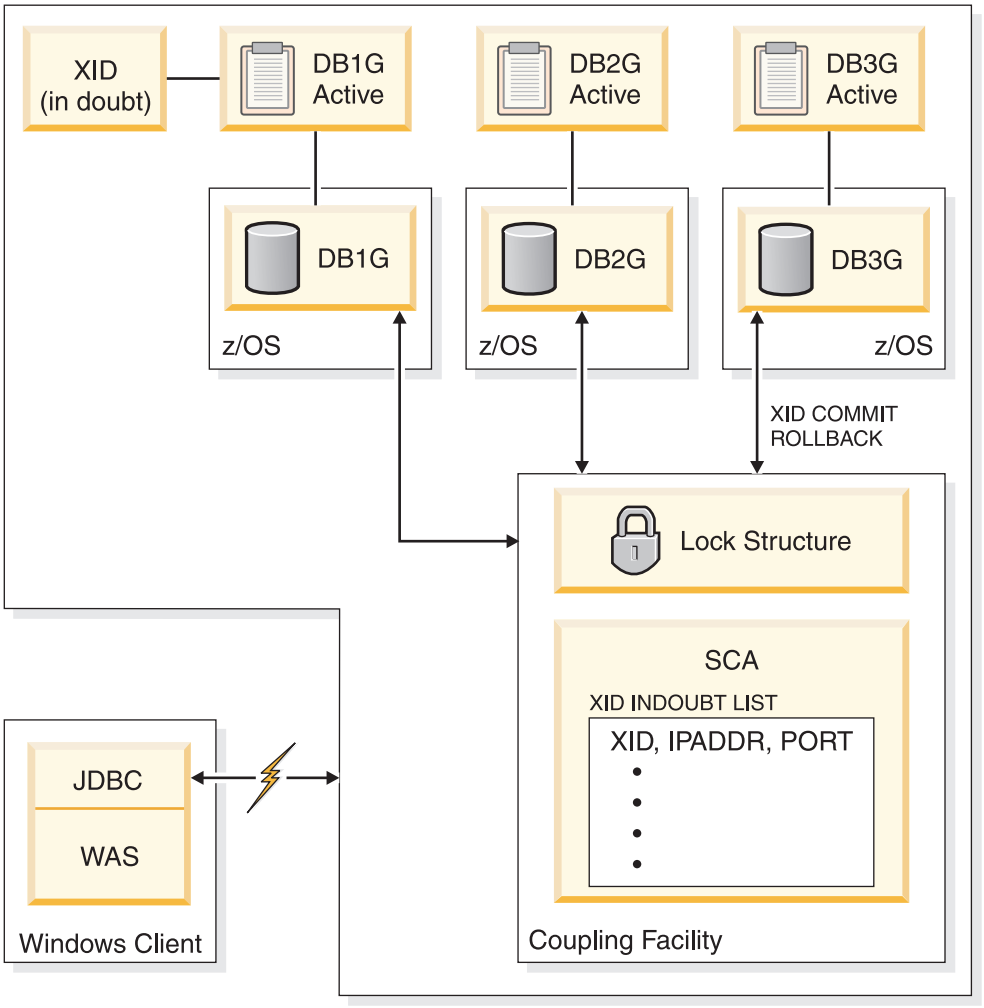


Figure 6. The XA recovery process

Related concepts:

Communicating with data sharing groups (DB2 Data Sharing Planning and Administration)

Storage estimates for data sharing environments

As part of planning for a data sharing environment, you should estimate the storage requirements for coupling facility structures and for DB2 resources.

Precise estimates for the sizes of coupling facility structures are hard to provide. Every environment is different, and storage allocation is affected by the processor model and the level of coupling facility control code. Use the information in this section to estimate initial size (INITSIZE) values. Depending on how much your work load varies, consider specifying a larger value for SIZE.

For duplexed structures, the SIZE and INITSIZE parameter values apply to both instances of the structure.

The information in this section assumes that all page sets in a particular group buffer pool are defined with the same GBPCACHE attribute. You can put page sets with different GBPCACHE attributes in the same group buffer pool, but you must adjust the formulas accordingly.

Coupling facility structure size allocation

When a new coupling facility structure is allocated for a data sharing group, its size is usually taken from the value of the INITSIZE parameter in the CFRM policy.

After the structure is allocated, you can dynamically change its size with the z/OS command SETXCF START,ALTER,SIZE=*newsize*,STRNAME=*strname*. The new SIZE value cannot be greater than the SIZE value in the CFRM policy, but it can be smaller than the INITSIZE value. If the coupling facility has enough space, z/OS increases (or decreases) the size of the structure to the new size; the INITSIZE value of the policy remains unchanged.

In most cases, DB2 uses the new SIZE value for any subsequent allocations of the structure instead of the INITSIZE value in the CFRM policy. Any of the following subsequent allocations can use the new SIZE value:

- A group buffer pool or SCA is deallocated and then reallocated
- A secondary structure is allocated for a duplexed group buffer pool, SCA, or lock structure, if duplexing is started after the size of the primary structure was dynamically changed
- Any structure is rebuilt with the z/OS command SETXCF START,REBUILD,STRNAME=*strname*

The new size is recorded across a restart of DB2 and is used for all subsequent allocations until one of the following events occurs:

- A CFRM policy is started, and the policy has a different INITSIZE value than the size of the structure that was dynamically changed with the SETXCF START,ALTER command
- Another SETXCF START,ALTER command is issued to dynamically change the size of the structure

Exception: If a lock structure is deallocated and all the members are down, the INITSIZE value is used. This is a consideration for disaster recovery or for situations where data sharing groups are cloned. During normal operation of a data sharing group, you are unlikely to encounter this situation.

Related concepts:

"How to change structure sizes" on page 190

Coupling facility structure sizer:

The coupling facility structure sizing tool (CFSizer) simplifies the task of estimating coupling facility structure storage size by asking you questions about your existing or planned configuration.

If you are not sure how to answer the questions, the help functions direct you to the source of the data on your system.

After you decide what your structure sizes are, include those values in the CFRM policy definition. See z/OS MVS Setting Up a Sysplex for more information about creating CFRM policies.

Related tasks:

Determine the size of each coupling facility structure (z/OS MVS Setting Up a Sysplex)

Recommendations for coupling facility sizes:

Coupling facility structures contain some static control structures. When a structure is initially allocated, these static structures are allocated to accommodate the potential size of the coupling facility structure.

In other words, the size of the static structures is proportional to the maximum size (the value of the SIZE parameter) of the coupling facility structure. If the SIZE value is much larger than the INITSIZE value, a large percentage of the initial structure's size might be used for these static structures, leaving you with little usable storage space in the structure.

Recommendation: In general, specify a SIZE value that is larger than the INITSIZE value, but limit the SIZE value in the following ways:

- Limit the SIZE value to no more than two to three times the INITSIZE value for the SCA and lock structure.
- Limit the SIZE value to 1.2 - 1.5 times the INITSIZE value for group buffer pools. The SIZE value should not be more than two times the INITSIZE value.

For example, if the INITSIZE value for a group buffer pool is 100 MB, specify a SIZE value of 200 MB or less.

Group buffer pool sizes

The two parts of a group buffer pool are data pages and directory entries. You can specify the ratio of data pages to directory entities in a group buffer pool, or you can use the DB2 default ratio.

Data pages

Data pages (sometimes called *data entries*) reside in the group buffer pool. The size of a data page is the same as the page size supported by the corresponding DB2 buffer pools (4 KB, 8 KB, 16 KB, or 32 KB).

If you are caching changed data only, you need enough space to cache changed data plus extra space for pages that are frequently referenced. By caching those frequently referenced pages in the group buffer pool, you can decrease the amount of time it takes for any member to refresh that page in its member buffer pool because you avoid the disk I/O.

If you choose GBPCACHE NONE or GBPCACHE SYSTEM, no user data pages are actually stored in the group buffer pool. However, with GBPCACHE SYSTEM, space map pages for LOBs are cached in the coupling facility.

Directory entries

A directory entry specifies the location and status of a page image somewhere in the data sharing group, whether the image is in the group buffer pool or in one of the member buffer pools. Only one directory entry exists for any given page, no matter how many places that page is cached.

The size of a directory entry is approximately 200 bytes, but it varies somewhat based on the size of the data pages and the CFLEVEL you are using.

Specifying a ratio

The amount of space that is allocated for a group buffer pool is determined by the ratio of directory entries to data pages. When you originally define a structure in the CFRM policy for a group buffer pool, you specify its total size. For DB2, the ratio defaults to five directory entries per data page. Later, you can change the ratio with the ALTER GROUPBUFFERPOOL command. The new value that you define takes effect when the group buffer pool is rebuilt or reallocated.

For group buffer pools that are defined with GBPCACHE(NO), ratios are ignored because no data is actually stored in the group buffer pool.

When possible, both a formula and a general guideline are provided to help you estimate the initial sizes and ratios of your group buffer pools. (The exception is for GBPCACHE ALL group buffer pools, for which only a general guideline is provided.)

The formula is not too complex and is likely to be more accurate, assuming that you are fairly confident of the values for the variables in the formulas. Otherwise, use the general guidelines and then adjust your values from there.

Related concepts:

Determining the correct size and ratio of group buffer pools (DB2 Data Sharing Planning and Administration)

Related reference:

zSeries Processor Resource/Systems Manager Planning Guide

Storage estimate for group buffer pools that cache changed data:

The size of a group buffer pool is related to the amount of sharing and the amount of updating. You can use a formula to estimate the storage size of group buffer pools that cache changed data.

An estimate must be based on the total amount of member buffer pool storage multiplied by a percentage based on the amount of update activity. As data sharing and updating increases, more pages must be cached in the group buffer pool, and more directory entries are needed to track inter-DB2 interest.

Formula

The following is a formula for estimating storage for group buffer pools that cache changed data:

```
Data_entries = U * D * R
Data(MB)     = Data_entries * P / 1024
Dir_entries  = Data_entries + (U * VP)
Dir(MB)      = 1.1 * Dir_entries * 0.2 / 1024
GBP(MB)      = Data(MB) + Dir(MB)
RATIO        = Dir_entries / Data_entries
```

Where:

U The estimated degree of data sharing:

- **1** A high amount of sharing with a lot of update activity
- **0.7** A moderate amount of sharing with a moderate amount of update activity

0.5 A low amount of sharing with a low amount of update activity

D The number of data pages written to disk per second for all members during peak activity. Do not use the number of pages written to the group buffer pool; D must be a count of distinct pages. To determine this value, use the field QBSTPWS from IFCID 0002 (the PAGES WRITTEN field of the buffer pool section of the OMEGAMON Statistics report.)

R The average page residency time in the group buffer pool, in seconds. This value is application-dependent, but you can assume that the typical range is 30 to 180 seconds. If you have no information about residency time, use 120.

General guideline: Make *R* large enough so that other members can refresh an invalidated local copy of a changed page. When a changed page is written to the group buffer pool, it invalidates local copies of the page in other members. The changed page needs to remain resident in the group buffer pool long enough for other members to refresh the page.

P The page size (4 KB, 8 KB, 16 KB, or 32 KB).

VP The number of data pages defined for the virtual pool (the sum across all of the members).

0.2 The approximate size of a directory entry, in KB.

1.1 The additional storage needed for coupling facility control structures.

Example: Assume that you have a two-member data sharing group for which you have determined the following information:

- The degree of data sharing is very high (U=1).
- 500 disk writes per second occur across both members (D=500).
- No information exists about the page residency time (R=120).
- The page size is 4 KB (P=4).
- Member 1 is configured with a virtual pool of 80000 buffers.
- Member 2 is configured with a virtual pool of 40000 buffers (VP=120000).

The calculation is as follows:

```
Data_entries = 1 * 500 * 120 = 60000
Data(MB)     = 60000 * 4 / 1024 = 234 MB
Dir_entries  = 60000 + (1 * 120000) = 180000
Dir(MB)      = 1.1 * 180000 * 0.2 / 1024 = 38 MB
GBP(MB)      = 234 MB + 38 MB = 272 MB
RATIO        = 180000  /60000 = 3.0
```

The preceding calculation indicates that the group buffer pool should be defined with an INITSIZE of 272 MB. Use the ALTER GROUPBUFFERPOOL command to change RATIO to 3.

General guideline: For installation planning purposes, you should use the following general guideline as an initial estimate for the size of a DB2 group buffer pool for table spaces, indexes, or partitions that cache only changed data (GBPCACHE(CHANGED)):

Add the local buffer pool storage for this buffer pool number across all the members of the group. Then, multiply this amount by one of these workload factors:

Factor	Condition
10%	For light sharing with a low amount of updating activity
20%	For medium sharing with a moderate amount of update activity
40%	For a high amount of sharing with a lot of update activity

You can run a trace for IFCID 0002 to obtain an estimate of the amount of data sharing in your system. Calculate the "degree of data sharing" by dividing QBGLGG, the number of get pages for group buffer pool-dependent objects, by QBSTGET, the number of get pages. A value that is less than 25% is considered to be light data sharing, a value between 25% and 75% is medium data sharing, and a value greater than 75% is high data sharing.

Remember that the type of workload you run can influence the amount of storage you use. For example, if you have “hot spots” in which updates to a single page are frequent rather than spread throughout the table space, you might need less storage for caching.

Example: Assume that the total buffer pool storage for all the members of the group is 400 MB, and you expect a medium amount of read/write sharing in the environment. The following calculation is an estimate for the size of a group buffer pool when GBPCACHE(CHANGED):

```
400 MB x 20% = 80 MB
```

Storage estimate for caching all data:

For installation planning purposes, you can use an initial size estimate of a group buffer pool that caches read-only pages with changed pages (GBPCACHE(ALL)).

Calculate the sum of the local buffer pool storage for this buffer pool number (virtual only) across all members in the group. Then, multiply this amount by one of these workload factors:

Factor	Condition
50%	Less than half of the table spaces, indexes, or partitions specify GBPCACHE ALL
75%	Half of the table spaces, indexes, or partitions specify GBPCACHE ALL
100%	75% of more of the table spaces, indexes, or partitions specify GBPCACHE ALL for a high sharing environment

Example: Assume that the local buffer pool storage on all members of the group adds up to 200 MB. Half of the page sets that come into the pool are defined as GBPCACHE(ALL). The following calculation is an estimate for the size of a group buffer pool when GBPCACHE(ALL):

```
200 MB x 75% = 150 MB
```

Storage estimate for caching no data:

For planning purposes, you can use a formula to estimate the size of group buffer pools that cache no data.

The following is a formula for estimating storage for group buffer pools that cache no data (GBPCACHE(NO)):

```
Dir_entries  = U * VP
Dir(MB)      = 1.1 * Dir_entries*0.2/1024
GBP(MB)      = Dir(MB)
RATIO        = n/a
```

If the group buffer pool itself is defined with GBPCACHE(NO), the ratio is ignored.

In summary, the variables are:

U The estimated degree of data sharing.

P The page size (4 KB, 8 KB, 16 KB, or 32 KB).

VP The number of data pages defined for the virtual pool (the sum across all the members).

Example: Assume that you have a two-member data sharing group for which you have determined the following information:

- The degree of data sharing is very high (U=1).
- Member 1 is configured with a virtual pool of 80000 buffers.
- Member 2 is configured with a virtual pool of 40000 buffers (VP=120000).

The calculation is as follows:

```
Dir_entries  = 1 * 120000 = 120000
Dir(MB)      = 1.1 * 120000 * 0.2/1024 = 25 MB
GBP(MB)      = 25 MB
```

The preceding calculation indicates that the group buffer pool should be defined with an INITSIZE of 25 MB. Use the command ALTER GROUPBUFFERPOOL to change the GBPCACHE attribute to NO. If you put GBPCACHE NONE page sets in a GBPCACHE(YES) group buffer pool, the calculation becomes more complicated. You do this because the RATIO is observed and you are probably going to waste a lot of space on unneeded data entries.

Related concepts:

"Storage estimate for group buffer pools that cache changed data" on page 182

Storage estimate for caching LOB space maps (GBPCACHE SYSTEM):

For planning purposes, you can use a formula to estimate storage for group buffer pools that cache LOB space map data.

The following is a formula for estimating storage for group buffer pools that cache LOB space map data:

```
Data_entries = (U * D / 10) * R
Data(MB)     = Data_entries * P / 1024
Dir_entries  = Data_entries + (U * VP)
Dir(MB)      = 1.1 * Dir_entries * 0.2 / 1024
GBP(MB)      = Data(MB) + Dir(MB)
RATIO        = MIN(Dir_entries / Data_entries, 255)
```

In summary, the variables are:

U The estimated degree of data sharing.

D The number of data pages that are written to disk per second for all

members during peak activity. Do not use the number of pages that are written to the group buffer pool; D must be a count of distinct pages. To determine this value, use the field QBSTPWS from IFCID 0002 (the PAGES WRITTEN field of the buffer pool section of the OMEGAMONStatistics report).

10 An estimate of the LOB system pages that are written for every LOB data page.

P The page size (4 KB, 8 KB, 16 KB, or 32 KB).

R The average page residency time in the group buffer pool in seconds.

VP The number of data pages that are defined for the virtual pool (the sum across all the members).

Example: Assume that you have a two-member data sharing group for which you have determined the following information:

- The degree of data sharing is moderate (U=.7).
- 10 disk writes per second occur across both members during peak activity (D=10).
- The space map page is resident in the group buffer pool page for 120 seconds (R=120).
- The page size is 32 KB (P=32).
- Member 1 is configured with a virtual pool of 20000 buffers.
- Member 2 is configured with a virtual pool of 10000 buffers (VP=30000).

The calculation is as follows:

```
Data_entries = ((.7 * 10)/ 10) * 120 = 84
Data_(MB)     = 84 * 32 / 1024 = 2.6 MB
Dir_entries  = 84 + (.7 * 30000) = 21084
Dir_(MB)      = 1.1 * 21084 * 0.2 / 1024 = 4.5 MB
GBP(MB)      = 2.6 MB + 4.5 MB = 7.1 MB
RATIO        = MIN (21084 / 84, 255)  = 251
```

The preceding calculation indicates that the group buffer pool should be defined with an INITSIZE of 7.1 MB. The ratio is less than the maximum value, so use the command ALTER GROUPBUFFERPOOL to change the ratio to 251.

Related concepts:

"Storage estimate for group buffer pools that cache changed data" on page 182

Related reference:

-ALTER GROUPBUFFERPOOL (DB2) (DB2 Commands)

PR/SM formulas for calculating sizes of group buffer pools:

You can use Processor Resource/System Manager (PR/SM™) formulas to calculate the sizes of group buffer pools.

You can also calculate group buffer pool sizes by using the coupling facility allocation formulas for cache structures found in zSeries Processor Resource/Systems Manager Planning Guide. The following table contains information that is used in those formulas. The size of cache structures in DB2 can vary greatly based on the amount of data for which inter-DB2 read/write interest exists at any given time. You probably need to monitor the use of the group buffer pools and adjust their sizes accordingly.

Table 41. Information for calculating cache structure sizes

Parameter	DB2 Value	Explanation
MSC	**1**	Maximum storage class
MCC	**1024**	Maximum castout class
MDAS	Dependent on page size being cached: **1** for 4 KB pages **2** for 8 KB pages **4** for 16 KB pages **8** for 32 KB pages	Maximum data area size
DAEX	**4**	Data area element characteristic
AAI	**0**	Adjunct assignment indicator
R_de	Set on ALTER GROUPBUFFERPOOL. Default is **5**. See the table below for more information on determining this value.	The directory portion of the directory-to-data ratio
R_data	Set on ALTER GROUPBUFFERPOOL command. See the table below for more information on determining this value.	The data object portion of the target directory-to-data ratio

The following table contains information about the formulas used to determine R_data and R_de. *N* is the RATIO entered on the ALTER GROUPBUFFERPOOL command. If *N* has a decimal point, all digits after the first decimal place are ignored. For example, 5.67 is treated as 5.6.

Table 42. Formulas for determining R_data and R_de.

Page Size	N has no decimal point	N has decimal point
4 KB R_de	N	N×10
4 KB R_data	1	10
8 KB R_de	N	N×10
8 KB R_data	2	20
16 KB R_de	N	N×10
8 KB R_data	4	40
32 KB R_de	N	N×10
32 KB R_data	8	80

Related reference:

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

Lock structure size

The coupling facility lock structure contains two parts: a lock entry table and a list of update locks.

The *lock entry table* is used to determine if inter-DB2 read/write interest exists on a particular hash class. *Hash classes* are resources that hash to a particular place in the lock table. The second part is a list of the update locks that are currently held (sometimes called a *modify lock list* or *record list table*). You can control the division

of the lock structure storage between these two components by using the IRLMPROC or by using an IRLM MODIFY command. If you do not specify how the structure is to be split, IRLM attempts to divide it with a 1:1 ratio between lock table entry (LTE) and record list entry (RLE) storage.

The total size of the lock structure must be large enough to limit hash contention, which prevents performance problems. The lock structure must also be large enough to prevent failures that result from a lack of record table storage to write a MODIFY entry (RLE). Proper specification for the number of LTEs can help avoid hash contention.

IRLM reserves 10% of the record table entries for “must complete” functions (such as rollback or commit processing), so that a shortage of storage does not cause a member to fail. However, if storage runs short in the record table, there can be an impact on availability (transactions are terminated), response time, and throughput.

The z/OS system can automatically alter the size of a lock structure in the coupling facility if it needs storage space. The ALLOWAUTOALT option in the CFRM policy specifies whether system-initiated automatic alters are to be allowed for a structure.

If no competing resource is using the coupling facility, consider specifying ALLOWAUTOALT(YES) to enable the lock structure to change as the workload grows. However, if you use this setting, other resources can force the IRLM lock structure size to decrease. This situation can cause an IRLM out-of-storage condition in the lock table and an abend.

If you need the lock structure to be stable, specify ALLOWAUTOALT(NO). If you do not specify ALLOWAUTOALT, the default value of NO is used.

Related concepts:

Avoiding false contention (DB2 Data Sharing Planning and Administration)

Auto Alter capabilities (DB2 Data Sharing Planning and Administration)

Changing the size of the lock structure (DB2 Data Sharing Planning and Administration)

Ways to monitor DB2 locking activity (DB2 Data Sharing Planning and Administration)

Lock entry size parameter:

The LOCK ENTRY SIZE parameter on installation panel DSNTIPJ determines the amount of space required for lock contention control information (that is, individual entries in the lock table).

The lock entry size and the number of lock table entries of the first IRLM to join the group determines the whole group's storage size for the lock table and the lock table entry width. The default is 2 bytes, which is probably the size you want, unless you immediately create a data sharing group of seven or more members. By restricting each lock entry to 2 bytes, you maximize the amount of RLE space available from the define structure size.

Storage estimate for the lock structure:

For installation planning purposes, the initial size of the lock structure is based on how much updating you do.

The following table gives you initial size value recommendations.

Recommendation: If you do not specify a value for the LTE= parameter in the IRLMPROC, choose a value for INITSIZE that is a power of 2. This enables IRLM to allocate the coupling facility storage so that half is used for lock table entries and the remainder is used for record table entries. If a 1:1 split occurs and total size is not a power of 2, you might experience severe shortage of space for the record table entries, resulting in DB2 or possible IRLM failures. A failure will occur because the number of lock table entries requested on CONNECT must be a power of 2. The record table is susceptible to storage shortages if the structure is too small or if the allocation of the lock table leaves too little storage for the record table.

When specifying a value for the LTE= parameter in the IRLMPROC, or when issuing the MODIFY irlmproc SET,LTE= command, you should monitor XES contention rates to determine the optimum value for your normal operating environment. If the contention rates appear to be too high, increase the LTE= parameter value to the next power of 2. Remember, any increase in the size of the lock table will cause a corresponding decrease in the record table, unless the structure size is also increased.

If you have little contention and want more storage available for record table entries, decrease the LTE= parameter value by a power of two. Anytime the number of lock table entries is decreased, you should monitor contention rates for a period of time.

Because structure allocation is done at CONNECT, any change that is made to the LTE= parameter value does not take effect unless the group is terminated, structure forced and the group restarted, or a REBUILD is done. Also, the LTE= parameter value of the first IRLM to CONNECT dictates the coupling facility structure sizes that are used by the group.

Table 43. Recommendations for lock structure size

INITSIZE	SIZE	Condition
16 MB	32 MB	For light sharing with a low amount of updating, or for a single-member data sharing group
32 MB	64 MB	For medium sharing with a moderate amount of update activity
64 MB	128 MB	For a high amount of sharing with a lot of update activity

SCA size

The shared communications area (SCA) is a list structure in the coupling facility. Your storage estimate for your data sharing group should include an estimate of SCA size.

The SCA contains the following information:

- Member names
- BSDS names
- Database exception status conditions about object in the database
- Recovery information, such as log data set names and the list of indoubt XA transactions

You can use the coupling facility structure sizing tool (CFSizer) to help you calculate structure sizes. The following table shows how to estimate the size of the SCA. The SCA size can be specified in 1 KB increments.

Table 44. Estimating storage for the SCA

Site size	Databases	Tables	INITSIZE	SIZE
Small	50	1000	16 MB	32 MB
Medium	200	4000	32 MB	64 MB
Large	400	10,000	64 MB	128 MB
Extra Large	600	50,000	128 MB	256 MB

Running out of space in the SCA can cause DB2 to fail. Because much of the space in the SCA is taken up with exception information, you can reclaim space by correcting database exception conditions. Each indoubt XID entry takes approximately 250 bytes. The additional SCA storage required to be reserved is dependent on the total number of database access threads (DBATs) that can be active for the data sharing group.If the data sharing group supports XA transactions, be sure to reserve additional space in the SCA for the indoubt transaction ID (XID) list. (DRDA XA protocol support enables DB2 servers to participate in WebSphere coordinated two-phase commit transactions that have multiple connections to more than one data source.) After a failure, indoubt XA transactions are resolved by WebSphere, or you can reclaim space by resolving indoubt XA transactions manually.

Related tasks:

Recovering indoubt threads (DB2 Administration Guide)

Determine the size of each coupling facility structure (z/OS MVS Setting Up a Sysplex)

How to change structure sizes

You can change the size of individual structures by changing the CFRM policy.

Then you should rebuild the structures by using the z/OS command SETXCF START,REBUILD,STRNAME=*strname*. (Be aware that this command does not work on structures that are actively being duplexed.)

z/OS attempts to reallocate a new instance of the structure in the same coupling facility if that coupling facility has enough storage space. If there is not enough space, z/OS looks at the preference list and uses the specified alternate coupling facility. After the space is allocated, DB2 rebuilds the information into the new structure. Any transactions that need the structure must wait until the rebuild is complete. It is best to plan a rebuild when other activity in the system is low.

Dynamic changes to structure sizes:

If the affected structure is allocated in a coupling facility with CFLEVEL greater than 0, you can dynamically increase the structure sizes up to a maximum limit specified on the CFRM policy by using the z/OS command SETXCF START,ALTER.

The advantages to this method are:

- DB2 can access the structures while a change is taking place.

- Less coupling facility storage is required because DB2 does not need to allocate enough space for a whole new structure. It dynamically adds or deletes storage from the existing structure.
- The command works on duplexed structures.

Related concepts:

"Storage estimates for data sharing environments" on page 179

Changing the size of the group buffer pool (DB2 Data Sharing Planning and Administration)

Problem: storage shortage in the SCA (DB2 Data Sharing Planning and Administration)

Related tasks:

Changing the lock structure size dynamically (DB2 Data Sharing Planning and Administration)

Related reference:

Updating a CFRM Policy(z/OS MVS Setting Up a Sysplex)

Automatic changes to structure sizes:

z/OS can automatically alter a coupling facility structure when the structure reaches an installation-defined or default percent-full threshold, as determined by structure full monitoring.

The alter process can change the size of the structure, reapportion the objects within the structure, or both. This function works the same in data sharing environments as it does in non-data sharing environments.See z/OS MVS Setting Up a Sysplex for complete information about allowing structures to be altered automatically.

IRLM storage size

For data sharing, plan for additional IRLM storage to accommodate data sharing-specific locks called *P-locks*.

P-locks are held on open page sets and on database descriptors (DBDs), skeleton cursor tables (SKCTs), and skeleton package tables (SKPTs). Unlike transaction locks, storage for P-locks is held even when no transaction activity exists; therefore they consume storage even with no transaction activity.

Plan also for the extra storage that IRLM needs to build retained locks in case other members fail. The following table shows the variables that you need to take into account.

Table 45. Variables that are used to estimate additional IRLM storage

Variable	Description	Calculation
X	P-locks [1]	N = (MAX_OPEN_DATA_SETS × 500) X = (N + (N × .40)
Y	Ability to hold update-retained locks for a failed member	Depends on the update intensity of the workload. Start with the following: Y = .25(X)(X)

Table 45. Variables that are used to estimate additional IRLM storage (continued)

Variable	Description	Calculation
Note:		
1. The formula is based on the assumption that more than one P-lock might be held on a page set occasionally (such as for castout activity), and an estimate of about 40% for P-locks on the EDM pool objects and for short-lived page P-locks. If you know that the EDM pool has relatively few objects in it, you can use a lower percentage for that value. Estimate the maximum number of open data sets or use the value that is specified for the subsystem parameter DSMAX.		

Related concepts:

"IRLM address space (IRLMPROC) storage requirements" on page 47

How DB2 determines the initial value of DSMAX (DB2 Performance)

Physical locks in data sharing (DB2 Data Sharing Planning and Administration)

Related reference:

"DSMAX field (DSMAX subsystem parameter)" on page 491

IRLM priority setting:

If IRLM priority is too low, storage might not be freed as quickly, and IRLM might run out of storage.

The IRLM address space priority, though it should remain high, should be lower than XCFAS.

Related tasks:

Determining z/OS Workload Manager velocity goals (DB2 Performance)

IRLM storage use:

Use the z/OS command MODIFY irlmproc,STATUS,STOR to see how much storage IRLM is using.

Related reference:

MODIFY irlmproc,STATUS (z/OS IRLM) (DB2 Commands)

Calculating storage for the coordinator and the assistants:

Sysplex query parallelism uses IRLM notify messages to pass data between the assistants and the coordinator.

About this task

The coordinator uses notify messages to communicate with the assistants. The assistants use notify messages to communicate with the parallelism coordinator. An *assisting DB2* is any member that receives query work from another member of the data sharing group. Before you use Sysplex query parallelism, ensure that you have enough ECSA to handle the notify messages.

Procedure

To estimate the amount of extra storage that IRLM requires for coordinating and assisting members:

1. Estimate approximately 200 KB of extra storage for any member that can be a coordinator. This storage is needed for messages that are sent to the assistants.
2. Estimate the storage that is required for an assisting member:
   a. Estimate the number of parallel tasks that can run concurrently on the assisting member.
   b. Divide that number by the number of members that are sharing the query work.
   c. Multiply the result by 32 KB.

   ```
   (numbers of queries × max concurrent tasks) / number of members) × 32 KB
   ```

Example

Assume that you have a data sharing group in which all four members participate in processing parallel queries. If you have a total of 10 queries executing concurrently, and the highest number of parallel tasks is approximately 40, the calculation is:

```
(10 × 40) / 4 = 100
100 × 32 KB = 3 MB of extra storage on assistant
```

Storage estimate for the EDM pool in a data sharing environment

If you are planning to use data sharing, you should consider additional storage requirements for the EDM pool.

General guideline

You can use the storage estimates for your capacity planning.

For data sharing, you might need to increase your storage estimate by about 10% because of the way DB2 cross-invalidates database descriptors (DBDs). This percentage is just an estimate; the actual amount of the increase depends on how often you create, drop, and alter objects in the data sharing group.

Cross-invalidating items in the EDM pool

DB2 does not have a backup EDM pool in the coupling facility for invalidating objects in the EDM pool (DBDs, cursor tables, and more) because these objects are modified less frequently than database data. So, there is one EDM pool for each member. When a DBD changes, DB2 uses XCF messages to notify other members which are also using that DBD that new transactions should use the new DBD, which is read into the EDM pool. Thus, it is possible that one transaction is using the new DBD while other, currently running transactions are still using the old DBD. In other words, more than one copy of a DBD can exist in the EDM pool.

Reducing the storage impact

For CREATE, ALTER, or DROP statements, the DBD is not modified until a COMMIT is issued. You can significantly reduce the number of EDM versions by issuing CREATE, ALTER, or DROP statements within a single COMMIT scope. However, the exclusive lock on the DBD is held until the COMMIT.

Related concepts:

"Default database storage requirements" on page 44

Related tasks:

"Calculating EDM pool size" on page 56

Before you enable DB2 data sharing

You have several options for enabling DB2 data sharing. After you enable data sharing, disabling data sharing is a very difficult process and one that you should avoid.

You can move to DB2 data sharing in one of three ways:

- Install Version 10 as a new subsystem, and then enable data sharing.
- Migrate from a Version 8 or a Version 9.1 non-data sharing environment to Version 10, and then enable data sharing.
- Migrate from a Version 8 or a Version 9.1 data sharing environment to Version 10.

Related concepts:

"Disabling and re-enabling DB2 data sharing" on page 736

Deadlock detection and resource timeouts in data sharing environments (DB2 Data Sharing Planning and Administration)

Related tasks:

"Enabling DB2 data sharing" on page 679

"Migrating an existing data sharing group to a new release" on page 711

Connecting CICS to DB2 (optional)

Before you enable data sharing, connect CICS to each member. The CICS attachment facility command, DSNC STRT, lets you override the subsystem name on startup.

About this task

To connect DB2 to CICS, you must regenerate several CICS tables with additional entries. Coordinate this connection with your CICS support group.

Related reference:

CICS attachment facility commands (DB2 Commands)

Connecting DB2 to IMS (optional)

Before you enable data sharing, attach IMS to every member that runs IMS applications. IMS must include a separate member SSM for every member DB2.

About this task

Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

1. Make DB2 load modules available to IMS.
2. Define DB2 to IMS.
3. Define new programs and transactions to IMS.

4. Prepare IMS applications for DB2.
5. Depending on your site, you might also need to:
   - Define DB2 plans for IMS applications.
   - Generate a user language interface.

Registering the command prefixes, member group attachment name, and subgroup attachment name

Before you enable data sharing, register the command prefix for a member, the group attachment name, and the subgroup attachment name for the group.

Before you begin

To specify subgroup attachment names, you must use Version 10 early code.

Procedure

To register the command prefixes, member group attachment name, and subgroup attachment name:

Use parameter library member IEFSSN*xx*. For example:

```
SUBSYS SUBNAME(ssname) INITRTN(DSN3INI)
   INITPARM('DSN3EPX,cmd-prefix<,scope<,group-attach<,subgroup-attach>>>')
```

By default, the subsystem definition does not include the subgroup attachment name. If you specify a value for SUBGRP NAME on installation panel DSNTIPK, the installation CLIST appends the name to the subsystem definition.

Recommendation: When you register the 1- to 8-character command prefix in parameter library member IEFSSN*xx* and specify the scope of the prefix, choose a scope of started (S). This specification lets all z/OS systems in a single PARMLIB member IEFSSN*xx* use all z/OS systems in the Parallel Sysplex. It also simplifies the task of moving a member from one system to another; you can stop DB2 on one z/OS system and start it on another. You do not need to re-IPL the system.

Results

The following sample definitions are examples of what might appear in the shared parameter library SYS1.PARMLIB:

```
DB1A,DSN3INI,'DSN3EPX,-DB1A,S,DB0A'
DB2A,DSN3INI,'DSN3EPX,-DB2A,S,DB0A,SBG1'
DB3A,DSN3INI,'DSN3EPX,-DB3A,S,DB0A,SBG1'
DB4A,DSN3INI,'DSN3EPX,-DB4A,S,DB0A,SBG2'
```

What to do next

If you need to change the command prefix parameters, you must change entry IEFSSN*xx* and re-IPL the host system. For example, if you want to change the command prefix scope from system-wide to Parallel Sysplex-wide, and you want to register the prefix at DB2 startup, change the M in the entry to S before you re-IPL.

If you want to use multiple-character command prefixes, make sure that your automation programs can handle multiple-character prefixes in messages before you change the prefixes.

Related concepts:

"Group attachment names and subgroup attachment names" on page 155

Related tasks:

"Migration step 13: Define DB2 Version 10 to z/OS: DSNTIJMV" on page 593

Increasing the size of the BSDS

Data sharing causes additional records to be written to the BSDS for member information.

About this task

Recommendations: To prevent the BSDS from expanding into secondary extents, change the size of the primary space allocation to 850 records. This is necessary only for members that have followed the IBM-recommended migration path from Version 8 or Version 9 without altering that record size. New installations and member installations already do this for you.

Procedure

To increase the space allocation for the BSDS:

1. Rename existing BSDSs.
2. Define larger BSDSs with the original names.
3. Copy the renamed BSDSs into the new BSDSs.

Results

You can do this using access method services. To see the definition that is used for the BSDSs, see the installation job DSNTIJIN.

Increasing the size of the SYSLGRNX table space

The SYSLGRNX directory table space contains the relative byte address (RBA) ranges showing when data sets are open for updating.

About this task

Because more members open and close data sets in a data sharing configuration, this table space is likely to grow with the addition of each new member to the data sharing group. And if you choose to copy indexes, this table space can grow even more.

Consider increasing the frequency with which you remove rows from this table space, or increasing the size. To see the definition used for SYSLGRNX, see installation job DSNTIJIN.

Procedure

To increase the space allocation for SYSLGRNX, use access method services:

1. Stop the table space.
2. Rename the existing SYSLGRNX data set.
3. Define a larger SYSLGRNX data set with the original name.
4. Using only DSN1COPY, copy the contents of the renamed data set into the new SYSLGRNX data set.
5. Restart the table space.

Database design recommendation

You should partition table spaces as much as possible and use as much separation of I/O as possible.

Define table spaces and indexes with GBPCACHE CHANGED: It is unlikely that members that are processing a large query will repeatedly read the same pages, so those pages do not need to be cached in the group buffer pool. Define the relevant table spaces and indexes with GBPCACHE CHANGED (the default).

Related tasks:

Partitioning for optimal parallel performance (DB2 Performance)

Application design planning

If you plan to develop applications for use in DB2 data sharing environments, keep several considerations in mind.

CACHE option of CREATE SEQUENCE statement

The CACHE option of the CREATE SEQUENCE statement is a performance and tuning option that directs DB2 to preallocate a specified number of sequential values in memory.

GUPI Sequence objects provide unique, sequential, and numeric values to DB2 applications. Applications can use sequence numbers for a variety of purposes, including the avoidance of concurrency and performance problems that can result when applications generate their own sequence numbers. Unlike application-generated sequences, DB2 sequence objects allow multiple transactions to concurrently increment the sequence number, and DB2 guarantees that each sequential value is unique. DB2 does not wait for a transaction that has incremented a sequence to commit before allowing the sequence to be incremented by another transaction. No retained locks are held to prevent access to the sequence.

Every time a cache is allocated, the SYSIBM.SYSSEQUENCES table is updated and a forced log record is written for the update. Using ORDER or NO CACHE will result in a SYSIBM.SYSSEQUENCES table update and the associated forced log write every time a new value is generated by DB2. Specifying CACHE provides faster access to the sequence since values are assigned from the cache. It also reduces the number of SYSIBM.SYSSEQUENCES updates and the associated forced log records. The SYSIBM.SYSSEQ table space is defined with the MAXROWS(1) attribute in order to minimize page P-lock contention in a data sharing environment as the SYSIBM.SYSSEQUENCES table gets updated.

DB2 always generates sequence numbers in order of request. However, when a sequence is shared across multiple members of a data sharing group, each DB2 member allocates its own cache of unique consecutive numbers for the sequence. Therefore, in situations where transactions from different members are requesting the next sequence number from the same sequence, values assigned for the sequence across multiple DB2 members may not be in strict numeric order.

Example: Assume that members DB2A and DB2B share a sequence named SEQ1 that starts with 1, increments by 1, and has cache = 20. If the transaction that is associated with DB2A makes the first request for a sequence number, DB2A allocates a cache of 20 values (from 1 to 20) and the value of 1 is provided to the application. If the transaction that is associated with DB2B makes the next request for a sequence number, DB2B allocates its own cache of 20 values (from 21 to 40)

and the value of 21 is provided to the application. Assuming that sequence number requests continue to arrive from the transactions that are associated with members DB2A and DB2B in this manner (one from DB2A and then one from DB2B), the values assigned for that sequence are 1, 21, 2, 22, 3, 23, and so on. Although the numbers are in sequence within each DB2, the numbers assigned across multiple DB2 members are not in strict numeric sequence.

In a data sharing environment, using the ORDER or NO CACHE option ensures that the values assigned to a sequence which is shared by applications across multiple DB2 members are in strict numeric order. In a data sharing environment, if ORDER is specified, then NO CACHE is implicit even if CACHE n is specified.

In a non-data sharing environment, the numbers are always assigned in strict numerical order, even if NO ORDER is specified; so specifying ORDER or NO CACHE is not necessary.

Important: Specifying ORDER or NO CACHE in a data sharing environment, or NO CACHE in a non-data sharing environment, is not recommended because it will result in a SYSIBM.SYSSEQUENCES table update and the associated forced log record for each value generated by DB2. This can lead to an excessive amount of log write I/Os, which can drastically reduce the log write bandwidth.

GUPI

Related reference:

CREATE SEQUENCE (DB2 SQL)

Applications using CICSPlex System Manager

CICSPlex® System Manager (CICSPlex SM) is a system-management tool that lets you manage several Customer Information Control System (CICS) systems as if they were one.

The dynamic transaction routing program that is supplied with CICSPlex SM balances the enterprise workload dynamically across the available application owning regions. CICSPlex SM lets you manage a variable workload without operator intervention and maintains consistent response times. It can do this because it routes transactions away from busy regions and from those that are failing or likely to fail, which improves throughput and conceals problems from end users.

Storm-drain effect:

In some situations, your DB2 applications must be sensitive to a resource-unavailable condition.

For example, assume that a database is stopped for planned maintenance, and that the application receives SQLCODE -904 from the system and ends normally. If your application ignores the SQLCODE -904 message, CICSPlex SM might continue to route work to the stopped system because it appears to complete its work rapidly. This is sometimes called the *storm-drain* effect.

When both of the following conditions are true, the storm drain effect can occur:
- The CICS attachment facility is down.
- You are using INQUIRE EXITPROGRAM to avoid AEY9 abends.

Again, because there has not been an abend, it appears as if work completes rapidly at that subsystem.

How a CICS exit routine can avoid the storm-drain effect:

You can write a resource manager interface program exit routine, XRMIOUT, to avoid the storm drain effect caused by SQLCODE -904 (resource unavailable). This exit routine does not avoid the storm-drain problem caused by using INQUIRE EXITPROGRAM to avoid AEY9 abends.

About this task

GUPI Using XRMIOUT, you can intercept the return from the resource manager. The exit routine can determine whether:

- The resource manager is DB2.
- SQLCODE -904 is in the SQL communication area (SQLCA).

If these conditions exist, abend the transaction instead of ending the transaction normally.

To determine if DB2 is the resource manager, compare 'DSNCSQL' with the value stored at the address that is included with the UEPTRUEN parameter that is passed to XRMIOUT, as shown in the following figure.

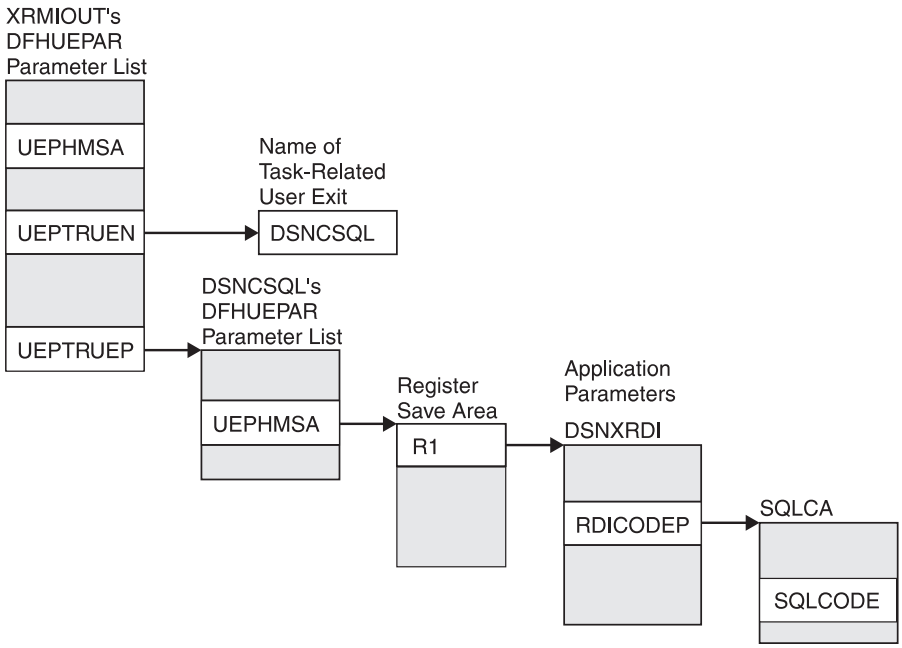


Figure 7. Pointers to resource manager name and SQLCODE

Procedure

To find the SQLCODE:

1. Find UEPTRUEP in the DFHUEPAR parameter list that is passed to XRMIOUT. UEPTRUEP contains the address of the DFHUEPAR parameter list that is passed to DSNCSQL.
2. Find UEPHMSA in DSNCSQL's DFHUEPAR parameter list. UEPHMSA points to the register save area that contains the application's registers.

3. Find register 1 in the register save area. Register 1 contains the address of the application parameters.

   The DSNXRDI macro maps the application parameters that are passed by the precompiler to DB2. The mapping macro is contained in the data set library prefix.SDSNMACS.
4. Find RDICODEP in the DSNXRDI structure. RDICODEP contains the address of the SQL communication area (SQLCA).
5. Find the SQLCODE. GUPI

Related concepts:

SQL control statements for external SQL procedures (DB2 SQL)

Related reference:

Exit XRMIOUT (CICS Transaction Server for z/OS)

A CICS enhancement that avoids the storm-drain effect:

The CICS Transaction Server for z/OS helps you avoid the storm-drain effect. With the CICS Transaction Server for z/OS, you do not need to use XRMIOUT to check for resource unavailable conditions.

CICS Transaction Server for z/OS also lets you benefit from the INQUIRE EXITPROGRAM command without causing the storm-drain effect.

Related information:

INQUIRE EXITPROGRAM (CICS Transaction Server for z/OS)

Order-dependent transactions

In limited cases, one transaction, called an originating transaction, updates DB2 data with INSERT, UPDATE, or DELETE. Before completing phase 1 of the commit process, the originating transaction spawns a second transaction that is dependent on the updates of the first transaction. These types of transactions are referred to as *order-dependent transactions*.

Description of the problem

In some situations, a dependent transaction can encounter a “row not found” condition that does not occur in non-data sharing environments. Even in non-data sharing environments, dependent transactions need to tolerate the “row not found” condition in cases where the originating transactions roll back.

When the problem might occur

The “row not found” condition can occur if all of the following conditions are true:

- The originating transaction spawns a dependent transaction before it completes phase 1 of the commit process.
- The dependent transaction runs on a different member than the member on which the originating transaction runs.

  Because each member has its own local buffer pools, the uncommitted buffered pages that are updated by the originating transaction are not immediately visible to the dependent transaction.
- The dependent transaction attempts to access a row that was updated by the originating transaction.

- The dependent transaction is not bound with an isolation level of repeatable read.

Preventing the problem

To prevent these periodic "row not found" situations, consider using the IMMEDWRITE(YES) option of the BIND and REBIND commands for plans or packages that spawn dependent transactions that might run on other members. You can also use the IMMEDWRI subsystem parameter. IMMEDWRITE(YES) means that DB2 writes the page as soon as the buffer update completes. DB2 writes the data to one of the following structures or devices:

- The group buffer pool
- Disk storage for GBPCACHE NO group buffer pools
- Disk storage for GBPCACHE NONE page sets
- Disk storage for GBPCACHE SYSTEM page sets

The IMMEDWRI parameter (IMMEDIATE WRITE field on installation panel DSNTIP8) can override the value of the IMMEDWRITE bind option on a data sharing member. The following table illustrates the implied hierarchy when using the IMMEDWRI subsystem parameter and the IMMEDWRITE option of the BIND and REBIND commands.

Table 46. The implied hierarchy of the immediate write option

IMMEDWRITE bind option	IMMEDWRI subsystem parameter	Value at run time
NO	NO	NO
NO	YES	YES
YES	NO	YES
YES	YES	YES

Note: YES always has precedence whether it is defined by the subsystem parameter or the bind option. Updated pages that are group buffer pool-dependent are written at, or before, phase 1 of the commit process.

The following alternatives can help solve the order-dependent transaction problem:

- Ensure that the originating transaction does not schedule the dependent transaction until the originating transaction has completed phase 1 of commit.
- Run the dependent transaction with an isolation level of repeatable read.
- If the dependent transaction is currently running with an isolation level of cursor stability AND CURRENTDATA(NO), changing it to use CURRENTDATA(YES) can sometimes solve the problem.
- Add statement retry logic to handle the return of a "row not found" condition.
- Run the dependent transaction on the same member as the originating transaction.

Related concepts:

"The scope and uniqueness of DB2 subsystem parameters" on page 671

Binds of plans and packages that move to a new machine

You do not need to rebind a plan or package to run it on a data sharing group.

However, if you are moving to a new machine that has different performance characteristics (moving from a System/390® microprocessor to a z/Series

microprocessor cluster, for example), it is to your advantage to rebind plans and packages on the machine on which they will be running.

Related concepts:

Access path selection in a data sharing group (DB2 Data Sharing Planning and Administration)

Release coexistence considerations

DB2 Version 8 or Version 9.1 can coexist with Version 10 in a data sharing environment and in a distributed environment.

In a data sharing or distributed environment, coexistence is limited to two releases at a time. A Version 10 member can coexist with other Version 10 members and Version 8 members, or with other Version 10 members and Version 9.1 members.

Release coexistence in a distributed environment

DB2 Version 10 communicates in a distributed data environment with Version 8 and Version 9.1 of DB2 by using DRDA access.

You can exchange data of newer types with other database systems by using the appropriate level of DRDA. The newer data types and the level of DRDA that they require are as follows:

BIGINT

The minimum SQL Application Manager (SQLAM) level is 6. DRDA specifies that DECIMAL(19,0) is to be used as the compatible data type to which a sender converts BIGINT data before transmission to a DRDA partner at a lower level. DB2 Version 7 and Version 8 convert BIGINT as follows:

- Accept BIGINT input from a DRDA requester by converting the input from BIGINT to DECIMAL(19,0).
- Accept BIGINT output from a DRDA server by converting the output from BIGINT to DECIMAL(19,0).

Beginning with Version 9 new-function mode, DB2 can process BIGINT directly and does not convert BIGINT to decimal before processing it.

BINARY and VARBINARY

The minimum SQL Application Manager (SQLAM) level is 8. DRDA defines a compatible mapping for use with DRDA partners at a lower SQLAM level for output data only. The compatible mapping for BINARY is CHAR FOR BIT DATA and the compatible mapping for VARBINARY is VARCHAR FOR BIT DATA.

DB2 issues SQLCODE -352 for BINARY and VARBINARY data to be input to servers at lower SQLAM levels.

DB2 downgrades BINARY and VARBINARY outputs to CHAR FOR BIT DATA and VARCHAR FOR BIT DATA types if the statement being executed is dynamic. Otherwise, DB2 issues SQLCODE -351 for BINARY and VARBINARY data to be output to a downlevel requester if the statement is static.

This behavior supports middleware products that perform SELECT * against servers at higher SQLAM levels. It is expected that these dynamic applications will issue DESCRIBE to understand the type and size of the

output data and will set up appropriate application data types to receive the output data. By choosing the correct type and size, the applications avoid padding problems.

TIMESTAMP WITH TIME ZONE

The minimum SQL Application Manager (SQLAM) level is 9.

DB2 issues SQLCODE -352 when servers at lower SQLAM levels (level 8 or below) request TIMESTAMP WITH TIME ZONE data as input.

If the precision of TIMESTAMP WITH TIME ZONE output data is greater or less than 6, DB2 sends the data to requesters at lower SQLAM levels after converting the data to TIMESTAMP. If the precision is greater than 6, the data is truncated to TIMESTAMP(6). If the precision of the output data is less than 6, the TIMESTAMP data is padded with zeros.

This behavior supports middleware products that perform SELECT * against servers at higher SQLAM levels. It is expected that these dynamic applications will issue DESCRIBE to understand the type and size of the output data and will set up appropriate application data types to receive the output data. By choosing the correct type and size, the applications avoid padding problems.

Release coexistence in a data sharing environment

Coexistence considerations are similar to those that you need to understand for the fallback environment. For example, objects that are frozen in fallback are generally not accessible from a down-level member (either Version 8 or Version 9.1) that is still running under a prior release of DB2.

In a data sharing group, DB2 can support members in Version 8 and members in Version 10 conversion mode from Version 8. Or, DB2 can support members in Version 9.1 and members in Version 10 conversion mode from Version 9.1. To support two releases, you must first apply the fallback SPE to all Version 8 or Version 9.1 members of the group. Release coexistence begins when you migrate the first data sharing member to Version 10. You must successfully migrate the first data sharing member to Version 10 before attempting to migrate the other data sharing members.

For the best availability, you can migrate the members to Version 10 one member at a time. When developing your migration plan, remember that most new functions that are introduced in Version 10 are not available to any members of the group until all members are migrated to Version 10 new-function mode.

Version 10 migration modes

The DB2 Version 10 migration process is designed to be accomplished in phases.

- Migrating a member to DB2 Version 10 initially puts that member into conversion mode (from Version 8 or Version 9.1). In this mode, no new Version 10 function is available for use.
- To make new function available, you must put the group through the enabling new-function mode process. Before attempting this process, confirm that all members are running DB2 Version 10. The enabling new-function mode process can only begin if all members are running Version 10.
- When the enabling new-function mode process completes successfully, the group enters Version 10 new-function mode, in which all new Version 10 function is available for use.

From either Version 10 enabling-new-function mode (from Version 8 or Version 9.1) or from Version 10 new-function mode, members cannot return to Version 10 conversion mode (from Version 8 or Version 9.1), nor fall back to the previous release. The only way to recover to a previous catalog level is to perform a point-in-time recovery of both the catalog and directory.

TSO, CAF, and RRSAF logon procedures

You can attach to either release of DB2 with your existing TSO, CAF, or RRSAF logon procedures without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you **must** update those procedures and jobs to point to the latest level of DB2 load libraries. If you forget to update those procedures and jobs before migrating to any release subsequent to Version 10, those procedures and jobs can no longer work in that subsequent release.

Private protocol

Other pre-Version 10 DB2 subsystems or data sharing groups can continue using private protocol to access information that is managed by a data sharing group if there is at least one Version 8 or Version 9.1 member of the group. If a package or plan is bound with DBPROTOCOL(PRIVATE), it can be processed successfully by a Version 8 or Version 9.1 member of the group. If a Version 10 member of the data sharing group attempts to run a package or plan that is bound with DBPROTOCOL(PRIVATE) and accesses a remote location, the application fails with SQLCODE -904. If a package or plan is manually rebound on a Version 10 member, it can only be rebound with the DBPROTOCOL(DRDA) bind option. If the rebound package or plan is then subsequently used on a Version 8 or Version 9.1 member, the object will be automatically rebound and it will retain the DBPROTOCOL(DRDA) setting.

After all members are migrated to Version 10 conversion mode (from Version 8 or Version 9.1), the group can no longer use private protocol objects or accept private protocol requests from other data sharing groups or subsystems.

Access paths of dynamic SQL queries

DB2 Version 10 can capture access paths of dynamic SQL queries in new catalog tables. However, in data sharing environments, Version 8 and Version 9.1 members are not able to use the stabilized access paths.

Database metadata stored procedures

In DB2 Version 10, the DB2 database metadata stored procedures have new Unicode load modules.

Table 47. Database metadata stored procedures and corresponding Unicode load modules

Stored procedure name	Unicode load module
SYSIBM.SQLCOLPRIVILEGES	DSNACPRU
SYSIBM.SQLCOLUMNS	DSNACOLU
SYSIBM.SQLFOREIGNKEYS	DSNAFNKU
SYSIBM.SQLFUNCTIONCOLS	DSNAFCOU
SYSIBM.SQLFUNCTIONS	DSNAFUNU
SYSIBM.SQLGETTYPEINFO	DSNATYPU

Table 47. Database metadata stored procedures and corresponding Unicode load modules (continued)

Stored procedure name	Unicode load module
SYSIBM.SQLPRIMARYKEYS	DSNAPRKU
SYSIBM.SQLPROCEDURECOLS	DSNAPCOU
SYSIBM.SQLPROCEDURES	DSNAPRCU
SYSIBM.SQLSPECIALCOLUMNS	DSNASPCU
SYSIBM.SQLSTATISTICS	DSNASTAU
SYSIBM.SQLTABLEPRIVILEGES	DSNATBPU
SYSIBM.SQLTABLES	DSNATBLU
SYSIBM.SQLUDTS	DSNAUDTU
DSNACLRU DLL (for remote procedure)	DSNACLRU

If you use the DB2 database metadata stored procedures in a coexistence environment, you need to make the Version 10 load modules that are listed above available to your Version 8 or Version 9.1 members.

To make your Version 10 load modules available to your Version 8 or Version 9.1 members, complete one of the following tasks:

- Copy the Version 10 load modules that are listed above into the Version 8 or Version 9.1 *prefix*.SDSNLOAD libraries.
- Include the Version 10 *prefix*.SDSNLOAD library in the STEPLIB of the WLM startup procedure for the database metadata stored procedures.

From V8

REORG SHRLEVEL REFERENCE

Version 8 does not support REORG SHRLEVEL REFERENCE for LOB table spaces. To use REORG SHRLEVEL REFERENCE for LOB table spaces in a coexistence environment with Version 8 members, direct utility jobs to the Version 10 member of the data sharing group. From V8

From V8

REORG TABLESPACE SHRLEVEL CHANGE PART x job

If you submit a REORG TABLESPACE SHRLEVEL CHANGE (or REFERENCE) PART x job on a Version 8 member, you might get an error message if a REORG TABLESPACE SHRLEVEL CHANGE (or REFERENCE) PART y job is running on another partition of the same table space that is on a Version 10 member. From V8

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Related tasks:

"Preparing for DB2 data sharing" on page 146

Related reference:

REORG TABLESPACE (DB2 Utilities)

IRLM service and release coexistence

As you apply IRLM service to members of a data sharing group, some members run with the newer service level, and some run with the older service level.

A mix of service levels can raise issues that you must consider.

Related tasks:

"Determining the function level of an IRLM group in coexistence" on page 725

Chapter 3. Loading the DB2 libraries

IBM distributes DB2 on tapes or cartridges, depending on which feature you order. To load the DB2 libraries, use System Modification Program Extended (SMP/E). SMP/E processes the installation tapes or cartridges and creates DB2 distribution libraries, DB2 target libraries, and SMP/E control data sets.

For information on using SMP/E, see DB2 Program Directory.

Related tasks:

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Editing the SMP/E jobs

Before running any of the SMP/E jobs, you must edit them. Some of the items that you might want to edit include link list options and naming conventions for the DB2 library data sets.

Creating JOB statements

The SMP/E jobs do not include JOB statements. Although JOB statements are often built automatically, creating your own JOB statements that are correct for your site is usually easier than editing the provided JOB statements.

Procedure

To create JOB statements:

1. If you are using ISPF to edit and submit the SMP/E jobs, edit a member that contains the JOB statement. Delete all text except the JOB statement. Then use the ISPF COPY command to copy the member into each job before submitting it.
2. If you are using TSO to submit the SMP/E jobs, edit a JOB statement and submit that JOB statement with each job. For example, data set JCL.CNTL(J) might contain the following code:

   ```
   //DB2INST   JOB ACCT,NAME,
   //          MSGCLASS=A,MSGLEVEL=(1,1),
   //          TIME=(10),USER=SYSADM,PASSWORD=xxxxxxxx
   /*JOBPARM   ....
   /*ROUTE     PRINT ....
   ```

   When you are ready to submit a job, use a command like the following command:

   ```
   SUBMIT (JCL(J) JCL(DSNTIJxx))
   ```

   In this command, *xx* represents the last two characters of the SMP/E job name. This command submits the JOB statement along with the job.

© Copyright IBM Corp. 1982, 2012

Choosing link list options

Before you run the SMP/E jobs to load the DB2 libraries, you can choose your link list options.

Procedure

To choose your link list options:

Consider and choose your link list options based on the following information about the load module libraries:

prefix.**SDSNLINK**

Contains modules that you must place in the link list because they are loaded at subsystem initialization during IPL. For Version 10, the load module library SDSNLINK contains modules that are called early (ERLY) code. Version 8 early code with the required maintenance is upward compatible with Version 10. Version 9.1 early code is downward compatible with Version 8 and upward compatible with Version 10. The Version 10 early code is downward compatible with Version 8 and Version 9.1. The Version 10 early code is incompatible with Version 7.

If you are migrating, maintenance to early code or an installation of new early code requires that you activate the early code, which can be done in a couple of ways. One option to activate the code is to IPL z/OS. An IPL is required if Version 8 early code is in use for that subsystem. Another option to activate maintenance to early code is to issue the REFRESH DB2,EARLY command. If you use this command, the subsystem that is having its early code refreshed must be stopped when the command is issued. To initialize a new DB2 subsystem, you can issue the SETSSI z/OS command, which will execute the early code initialization routine and update the subsystem vector table (SSVT).

You should schedule a z/OS IPL before or during a migration to a new release of DB2 This IPL is necessary because migration job DSNTIJMV makes changes to SYS1.PARMLIB that are not recognized by z/OS until the next IPL. Alternatively, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands to define the subsystem to z/OS until the next IPL. Changes that DSNTIJMV makes to the SYS1.PARMLIB might affect the following libraries:

- New subsystem definitions in IEFSSN*xx*
- New APF libraries in IEAAPF*xx*
- New load module libraries in LNKLST*xx*

prefix.**SDSNLINK**

- Contains early code
- Can be shared by multiple subsystems and releases of DB2
- Is APF-authorized

prefix.**SDSNLOAD**

- Is a main load module repository
- Can be shared by multiple subsystems at the same release level
- Allows only DB2 to modify code
- Holds default exit routines
- Is APF-authorized
- Must be a PDSE

prefix.**SDSNLOD2**

Contains a PDSE data set, which contains JDBC and SQLJ DLLs.

prefix.**SDSNEXIT**

- Holds the subsystem parameter module, DSNHDECP, and user-written exit routines
- Is modified by user
- Is APF-authorized

IRLM link list requirement

You must add the IRLM load module DXRRL183 to the link list. This requires that you copy the module into another library if the IRLM load module is not in the link list. Although IRLM 2.3 is required in DB2 Version 10, you can use the DXRRL183 load module from IRLM 2.2. After you apply maintenance to IRLM that affects DXRRL183, remember to copy the updated module to the link list.

Integrated Cryptographic Services Facility link list requirement

If you plan to use encryption with your DB2 subsystem, you must include in the link list the Integrated Cryptographic Services Facility library that contains the SCSFMOD0 load module. The SQL statement SET ENCRYPTION PASSWORD and the following built-in functions require this support:

- ENCRYPT
- DECRYPT_BIT
- DECRYPT_DB
- DECRYPT_CHAR
- GETHINT

Supporting one DB2 subsystem

You can choose among several methods of maintaining a single DB2 subsystem. The following steps describe what is probably the easiest method for most sites:

1. Change the SMP/E procedure DSNALLOC to assign all load modules to *prefix*.SDSNLOAD. You can do this by changing the data set name for DDDEF (SDSNLINK) from *prefix*.SDSNLINK to *prefix*.SDSNLOAD.
2. Remove the allocation for *prefix*.SDSNLINK from the allocation job DSNALLOC.
3. Include *prefix*.SDSNLOAD (instead of *prefix*.SDSNLINK) in the LNKLST*xx* member of SYS1.PARMLIB.

Supporting multiple DB2 subsystems

Supporting multiple subsystems can mean several things. You can have two or more DB2 subsystems at the same release and service level (for example, two DB2 Version 10 subsystems). Create separate libraries for DSNHDECP and user-written exit routines of each DB2 subsystem.

You can also have two or more DB2 subsystems at the same release level, but at different service levels. For example, you can have a DB2 Version 10 production subsystem and a DB2 Version 10 test subsystem at different service levels. Alternately, you can have two DB2 subsystems at different release levels. For example, you can have a Version 10 subsystem and a Version 9.1 subsystem.

In either of these cases, you can use an INCLUDE statement in JCL or a started task to identify the necessary load modules.

The DB2 subsystem must use the appropriate release level of *prefix*.SDSNLOAD, but the application attachment code (for example, CICS,

CAF, or TSO attachment facilities) can use code that is either one release level down or one release level up from that of *prefix*.SDSNLOAD. To use application attachment code that is either one level down or one level up from that of *prefix*.SDSNLOAD, place the attachment code in a different STEPLIB data set from the STEPLIB data set that DB2 executes.

Related tasks:

"Preparing for DB2 data sharing" on page 146

Accessing the correct DB2 program library

If you do not place *prefix*.SDSNLOAD in the LNKLST*xx* member of SYS1.PARMLIB, you must provide JOBLIB or STEPLIB statements for it in certain types of programs and procedures.

About this task

The installation and migration jobs that are provided with DB2 Version 10 already contain the necessary JOBLIB or STEPLIB statements. In addition, the startup procedures that DB2 provides include STEPLIB statements for their respective program libraries, *prefix*.SDSNLOAD, and *prefix*.SDSNEXIT.

Procedure

To access the correct program library:

Provide STEPLIB or JOBLIB statements for the following types of programs and procedures if you do not place *prefix*.SDSNLOAD in the LNKLST*xx* member of SYS1.PARMLIB.

- **TSO or batch jobs** that access DB2 services require JOBLIB or STEPLIB statements for *prefix*.SDSNLOAD. These jobs include TSO logon procedures and batch jobs that access the DSN command and subcommands, the DB2 precompiler, and DB2 utilities.
- **IMS control, message, and batch processing jobs** also require JOBLIB or STEPLIB statements for *prefix*.SDSNLOAD. You must specify the DB2 load library in the startup procedure for each IMS region (IMS control, message processing program (MPP), batch message processing (BMP), and Fast Path region) that can communicate with DB2. You can do this in two ways:
  - If all the data sets that the JOBLIB or STEPLIB statement refers to for an IMS region are APF-authorized, add the DD statement for *prefix*.SDSNLOAD to the JOBLIB or STEPLIB statement. If you are using the DYNAM option of COBOL, the IMS RESLIB DD statement must precede the reference to *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statement.
  - If any of the data sets that the JOBLIB or STEPLIB statement refers to for the IMS region are not APF-authorized, add the DFSESL DD statement for *prefix*.SDSNLOAD. All libraries that are specified on the DFSESL DD statement must be APF-authorized. The DFSESL DD statement is not required by the DB2 DL/I batch support. IMS requires that an IMS RESLIB DD statement also be referred to by the DFSESL DD statement, as in the following example:

    ```
    //DFSESL     DD      DSN=ims_reslib,DISP=SHR
    //           DD      DSN=prefix.SDSNLOAD,DISP=SHR
    ```
- **CICS procedures**, including the CICS initialization JCL, also need to include DB2 libraries.

Related tasks:

CICS startup JCL requirements for connection to DB2

Performance implications of installation libraries

The performance of your system might be degraded by including modules in the libraries that are included in the link list and you might want to consider the strategies below. These general suggestions might not match the specific needs of your site.

Adding many modules to the libraries that are included in the link list can reduce system performance. However, adding only a few modules to the libraries requires additional STEPLIB or JOBLIB statements. Because these STEPLIB or JOBLIB statements must be searched before the link list is searched, this approach can also reduce system performance. The approach that produces the best performance for your site depends on the environment in which you use DB2. Regardless of which attachment facilities you use, the modules in *prefix*.SDSNLINK must always be in the link library list.

If you are using DB2 with IMS, you should probably include *prefix*.SDSNLINK, not *prefix*.SDSNLOAD, in the LNKLST*xx* member of SYS1.PARMLIB because both the IMS RESLIB and *prefix*.SDSNLOAD have the DSNHLI alias. Place the needed STEPLIB or JOBLIB statements in the IMS procedures.

If you are using DB2 with IMS and you want *prefix*.SDSNLOAD (in addition to *prefix*.SDSNLINK) in the LNKLST*xx* member of SYS1.PARMLIB, ensure that the library concatenation for *prefix*.SDSNLOAD and the IMS RESLIB are correct for your site because both libraries have the DSNHLI alias.

If you are using DB2 with CICS, you should probably put *prefix*.SDSNLINK, not *prefix*.SDSNLOAD, in the LNKLST*xx* member of SYS1.PARMLIB. Then, place the needed STEPLIB or JOBLIB statements in the CICS procedures.

If you use the TSO attachment facility, RRS attachment facility, and call attachment facility, you might need to handle placement of load modules somewhat differently, as follows:

- If you use the DSN command and its subcommands infrequently, place **only** *prefix*.SDSNLINK in the LNKLST*xx* member of SYS1.PARMLIB. Provide the necessary STEPLIB or JOBLIB statements in your TSO logon procedures or in your JCL if you are using batch.
- If you use the DSN command and its subcommands frequently, you might also want to move the TSO attachment facility load modules to a library that is defined in the LNKLST*xx*. The TSO attach modules are DSNECP00, DSNECP10, DSNESM00, and DSNELI.
- If you use the call attachment facility (CAF) frequently, move the CAF load modules (DSNACAB, DSNACAF, and DSNALI) to a library that is defined in the LNKLST*xx*.
- If you use the RRS attachment facility (RRSAF) frequently, move the RRSAF load modules (DSNARRS and DSNRLI) to a library that is defined in the LNKLST*xx*.
- If you use the CAF, RRSAF, or the DSN command and its subcommands frequently, you can move the eligible load modules to a library that is defined in the link pack area (LPA), in IEALPA*xx* member of SYS1.PARMLIB. The CAF and DSN load modules must reside below the 16 MB line of z/OS virtual storage.

- The TSO load modules that you can place in the LPA are DSNECP00, DSNECP10, DSNESM00, and DSNELI. If you include these modules in the LPA, remember to include the appropriate aliases for DSNECP00 (DSN) and DSNELI (DSNHLI).
- The CAF load modules that you can place in the LPA are DSNACAF and DSNALI. If you include these modules in the LPA, remember to include the appropriate alias for DSNALI (DSNHLI2). Do not include DSNACAB in the LPA because it is a data-area-only, non-executable load module.
- The RRSAF load modules that you can place in the LPA are DSNARRS and DSNRLI. If you include these modules in the LPA, you must include the appropriate alias for DSNRLI (DSNHLIR).

Attention: If modules are moved or copied from one library to another, you must make changes to SMP/E control data to reflect the movement. If you do not make these changes, future service or changes to the modules will not be processed correctly.

Naming conventions for DB2 library names

You need to modify the DB2 library data set names in the SMP/E jobs.

These data sets are listed in Table 1 of "What you produce" on page 216. Their names are composed of three parts:
- A user-defined prefix
- A fixed base name: for example, SDSNLOAD
- An optional user-defined suffix

The Version 10 default prefix (*prefix*) is used in this information; the default suffix is null. You need to edit each of the DB2 SMP/E jobs and follow the directions in the header notes of each job to specify the names of the SMP/E data sets. If you want to add a suffix, edit the SMP/E procedures and allocation jobs. The prefix cannot exceed 18 characters. The suffix cannot exceed 17 characters, minus the length of the prefix. In addition, any data set names that exceed eight characters must be in groups of no more than eight characters, separated by periods. The qualified data set name cannot exceed 44 characters.

You can also change the base name of these libraries or load them into another data set. If you do this, however, you might need to do additional editing of the installation or migration jobs. The DSNTINST CLIST, which you use later to tailor the installation and migration jobs, uses the following default data set names:

prefix.ADSNLOAD	*prefix*.SDSNMACS
prefix.SDSNCLST	*prefix*.SDSNSAMP
prefix.SDSNEXIT	*prefix*.DBRMLIB.DATA
prefix.SDSNLINK	*prefix*.RUNLIB.LOAD
prefix.SDSNLOAD	*prefix*.SRCLIB.DATA
prefix.SDSNDBRM	*prefix*.SDSNIVPD
prefix.SDXRRESL	*prefix*.SDSNC.H

Recommendation: Use the supplied naming convention.

Document any changes you make to the library names in the SMP/E jobs. You must specify these library names again during the ISPF tailoring session.

Specifying SMP/E data set options

You have several options regarding how you establish and use SMP/E data sets.

About this task

You must decide whether you want DB2 and IMS to share SMP/E data sets. You must also decide whether you need an additional set of SMP/E data sets. An additional set of SMP/E data sets is required if you support more than one release of DB2.

Related information:

SMP/E for z/OS: User's Guide

Sharing SMP/E data sets with IMS

DB2 and z/OS cannot share SMP/E data sets because some module names and macro names are common to both products. Under certain conditions, however, DB2 can share SMP/E data sets with IMS.

About this task

If you do not share SMP/E data sets with IMS, skip this step.

The allocation job DSNALLOC defines a new set of SMP/E data sets that DB2 and IMS are to share.

Procedure

To define sharing of SMP/E data sets between DB2 and IMS:

You must modify your allocation job for either of the following situations:

- **Situation 1**: You decide to have separate SMP/E data sets for DB2 and IMS. In certain situations, DB2 and IMS cannot share SMP/E data sets. You must have separate SMP/E data sets if you want to have two IRLMs.

  Even if you are not required to have separate SMP/E data sets, you might want to keep them separate. If DB2 and IMS share the SMP/E data sets, you need to accept or reapply DB2 corrective service to these data sets to allow IMS SYSGENs.

  To establish separate SMP/E data sets for DB2 and IMS, change the data set prefix that your allocation job uses to a value other than the prefix that you use for your current IMS SMP/E data sets. The allocation jobs use the prefix “IMS”. Changing this prefix prevents the allocation job from replacing your current SMP/E data sets and still allows it to create new SMP/E data sets.
- **Situation 2**: You decide to share SMP/E data sets between DB2 and IMS, but you want to use the SMP/E data sets that already exist for IMS. To do this, remove the data set allocation and initialization statements from your allocation job. When you run the job, no SMP/E data sets are created, and DB2 then shares the existing SMP/E data sets with IMS.

SMP/E data sets for two releases

A single set of SMP/E zone structures can record only one release of DB2. Maintaining separate zone structures for both Version 10 and your previous release is strongly recommended until you are sure that you will not need to fall back.

The SMP/E jobs that are provided with DB2 assume that you will allocate a new set of SMP/E data sets for the new release. When you run your allocation job

(DSNALLOC), it creates a set of SMP/E data sets. If you choose to reuse your Version 8 or Version 9.1 zone structure, you can run job DSNTIJUD to delete SMP/E data for Version 8 or Version 9.1. However, after you run this job, you cannot fall back to your previous release.

You can create an additional set of SMP/E data sets either by copying them from a prior release of DB2 or by allocating a new set. Allocating a new set is faster because no data must be deleted.

Recommendation: Copy a prior set so that you can then perform service regression checking.

Specifying a new SMP/E prefix for IRLM

The SMP/E prefix in the SMP/E jobs is the same for the new IRLM as for the old IRLM. Consequently, if you do not change the SMP/E prefix, the jobs overwrite your old IRLM.

Procedure

If you do not want the jobs to overwrite your old IRLM:

Edit the SMP/E jobs accordingly.

DB2 utilities packaging

Several utilities are included with DB2 at no extra charge. Other utilities are available as a separate product.

The following utilities are core utilities, which are included (at no extra charge) with Version 10 of DB2 for z/OS:
- CATENFM
- CATMAINT
- DIAGNOSE
- LISTDEF
- OPTIONS
- QUIESCE
- REPAIR
- REPORT
- TEMPLATE
- All DSN stand-alone utilities

All other utilities are available as a separate product called the **DB2 Utilities Suite** (5655-V41, FMID JDBAA1K), which includes the following utilities:
- BACKUP SYSTEM
- CHECK DATA
- CHECK INDEX
- CHECK LOB
- COPY
- COPYTOCOPY
- EXEC SQL
- LOAD
- MERGECOPY
- MODIFY RECOVERY
- MODIFY STATISTICS
- REBUILD INDEX

- RECOVER
- REORG INDEX
- REORG TABLESPACE
- RESTORE SYSTEM
- RUNSTATS
- STOSPACE
- UNLOAD

All DB2 utilities operate on catalog, directory, and sample objects, without requiring any additional products.

SMP/E jobs for DB2 utility products

To load the DB2 utility products, use System Modification Program Extended (SMP/E). SMP/E processes the installation cartridges and creates DB2 distribution target libraries.

DB2 provides several jobs that invoke SMP/E. These jobs are on the cartridge that you received with the utility product. The job prologues in these jobs contain directions on how to tailor the job for your site. Follow these directions carefully to ensure that your DB2 Utilities Suite SMP/E process works correctly.

The SMP/E RECEIVE job, DSNRECVK, loads the DB2 Utilities Suite Version 10 program modules, macros, and procedures into temporary data sets (SMPTLIBs). If these jobs fail or abnormally terminate, correct the problem and rerun the jobs.

The SMP/E APPLY job, DSNAPPLK, copies and link-edits the program modules, macros, and procedures for the DB2 Utilities Suite Version 10 into the DB2 target libraries.

The SMP/E ACCEPT job, DSNACCPK, copies the program modules, macros, and procedures for the DB2 Utilities Suite Version 10 into the DB2 distributed libraries.

Related information:

DB2 for z/OS Program Directories

Operation of DB2 utilities in a mixed-release data sharing environment

The utilities batch module, DSNUTILB, is split into multiple parts: a release-independent module called DSNUTILB, a release-dependent module DSNUT101, and utility-dependent load modules.

To operate in a mixed-release data sharing environment, you must have the release-dependent modules from both releases and all applicable utility-dependent modules available to the utility jobs that operate across the data sharing group. For information about sharing utility modules in a mixed-release data sharing environment, seeLoad module names for running purchased utilities in coexistence (DB2 Installation and Migration).

Related reference:

Load module names for running purchased utilities in coexistence (DB2 Installation and Migration)

What you produce

During SMP/E processing, DB2 is loaded into the distribution and target libraries. DB2 uses the distribution libraries to maintain DB2 and contain the master copy of all elements for your DB2 subsystem.

The target libraries contain the various DB2 components. DB2 target libraries are updated when you apply corrective service.

The following table describes all the DB2 distribution libraries. The distribution libraries contain the master copy of all elements for your DB2 system.

Table 48. DB2 distribution libraries

Distribution libraries	Description
prefix.ADSNBASE	This library contains all jobs that are required to complete SMP/E installation.
prefix.ADSNENU or ADSNDKF	This library contains the DB2 English or Kanji task panels, respectively.
prefix.ADSNLOAD	This library contains an individual object module for every DB2 load module. It contains the IRLM load modules if you choose to install IRLM into the same distribution libraries as DB2. This library must be a PDSE data set.
prefix.ADSNLOD2	This library contains a PDSE data set, which contains JDBC and SQLJ DLLs.
prefix.ADSNHFS	This library contains the data that is to be copied into z/OS UNIX System Services.
prefix.ADSNIVPD	This library contains the IVP input data and expected output for sample applications.
prefix.ADSNMACS	This library contains the DB2 macros, sample programs, sample data, initialization data, TSO CLISTs, ISPF panels, and ISPF messages.
prefix.ADXRLOAD	This library contains an individual object module for every IRLM load module.
prefix.ADXRSAMP	This library contains the installation procedures for installing IRLM Version 2.

The following table describes all DB2 target libraries.

Table 49. DB2 target libraries

Target libraries	Description
prefix.SDSNBASE	This library contains all jobs that are required to complete SMP/E installation.
prefix.SDSNC.H	This library contains the header files. The command-line interface (CLI) requires header files. C language application programs can use header files.

Table 49. DB2 target libraries (continued)

Target libraries	Description
prefix.SDSNCLST	This TSO CLIST library contains code that simplifies the process of installing and migrating, aids program preparation and the use of DB2 utilities, and allows the use of DB2 Interactive (DB2I).
prefix.SDSNDBRM	This library contains the system DBRMs for DB2 Version 10.
prefix.SDSNEXIT	This program library is empty when first created. The installation jobs put the DSNZP*xxx* subsystem parameters load module, application defaults load module (*dsnhdecp*), and user exit modules into this library.
prefix.SDSNIVPD	This library contains the IVP input data and the expected output for sample applications.
prefix.SDSNLINK	This library contains early code of Version 10.
prefix.SDSNLOAD	This library contains Version 10 load modules. This library must be a PDSE data set.
prefix.SDSNLOD2	This library contains the PDSE data set, which contains JDBC and SQLJ DLLs.
prefix.SDSNMACS	This macro library contains macros that are needed for the CICS and IMS attachment facilities, the initialization parameter macros, and some data-mapping macros that are needed for some applications.
prefix.SDSNPFPE or *prefix*.SDSNPFPK	*prefix*.SDSNPFPE contains the English task and help panels, and *prefix*.SDSNPFPK contains the Kanji task and help panels.
prefix.SDSNSAMP	This initialization library contains the sample applications and data, the jobs for installing and migrating, the default installation and migration parameters, and catalog initialization data for DB2. The JCLIN for each FMID is stored in this library.
prefix.SDSNSPFM	This DB2 ISPF message library contains messages that are issued during install or migrate processing.
prefix.SDSNSPFP	This library is the DB2 ISPF library for installation task and help routing panels.
prefix.SDSNSPFS	This library is the DB2 ISPF skeleton library that is used to produce EDITJCL.
prefix.SDSNSPFT	This library is the DB2 ISPF command table library.
prefix.SDXRRESL	This library contains the IRLM load modules. This library might be empty if you chose to install IRLM elsewhere.
prefix.SDXRSAMP	The IRLM samples library might be empty if you chose to install IRLM elsewhere.

For a description of each SMP/E job that DB2 provides, see the following table. These jobs allocate the DB2 libraries and load them with the data from the installation tapes or cartridges.

Table 50. List of SMP/E jobs

Job name	Description
DSNTIJAA	This job creates the DB2 target and distribution zones, and defines the SMP/E control data sets in these zones and in the SMP/E global zone.

Table 50. List of SMP/E jobs (continued)

Job name	Description
DSNACEP1	This job invokes SMP/E to accept all the required and additional FMIDs into the DB2 distribution libraries (DLIBs).
DSNALLOC	This is the SMP/E allocation job. It creates the DB2 target and distribution libraries and defines the libraries in the SMP/E target and distribution zones for DB2 for the required and optional FMIDs.
DSNISMKD	This sample job invokes the supplied DSNMKDIR EXEC to allocate HFS paths for JDBC and SQLJ.
DSNLSMKD	This sample job invokes the supplied DSNLMKDR EXEC to allocate HFS paths for MQListener.
DSNWSMKD	This sample job invokes the supplied DSNWMKDR EXEC to allocate HFS paths for Web Services Object Runtime (WORF).
DSNAPPL1	This job invokes SMP/E to apply all the FMIDs to the DB2 target libraries for the required and additional FMIDs.
DSNRECV1	This job invokes SMP/E to receive all the required FMIDs (from both tapes or cartridges) from the base tape into the SMP/E control data sets.
DSNRECV2	This job invokes SMP/E to receive all the required FMIDs (from both tapes or cartridges) for IRLM into the SMP/E control data sets.
DSNRECV3	This job invokes SMP/E to receive all the additional FMIDs (from both tapes or cartridges) for ODBC, JDBC, and SQLJ into the SMP/E control data sets.
DSNRECV4	This job invokes SMP/E to receive FMIDs (from both tape or cartridge) for the Kanji DB2I panels into the SMP/E control data sets.
DSNTIJUD	This job invokes SMP/E to delete all Version 8 or Version 9.1 entries from the SMP/E libraries.
DSNDDEF1	This job creates DDDEF entries for the SMP/E target and distribution libraries.

Chapter 4. Tailoring DB2 jobs to your environment using the installation CLIST

The installation CLIST displays a series of ISPF panels that prompt you to supply parameter values or accept the supplied default values. In most cases, the fields on the ISPF panels correspond with DB2 subsystem parameters.

About this task

The values that you enter or accept on the ISPF panels describe the operating characteristics of your DB2 system. If you change the default values when you install DB2 or when you migrate to a newer release, the CLIST verifies that the values that you enter are within the allowable ranges. After a successful installation or migration, you can subsequently update these values to improve your operations.

Running the installation CLIST also prepares the jobs that are required for installation or migration. The CLIST tailors each installation or migration job according to the values that you specify on the ISPF panels.

Important: If you use the installation CLIST, do not change any members in your *prefix*.SDSNSAMP library. If you do, the installation CLIST can fail or produce unpredictable results when tailoring the installation or migration jobs.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Running the installation CLIST

To use the ISPF panels, you must first make the DB2 ISPF library available to TSO and then invoke the installation CLIST DSNTINST in ISPF mode. Save the output that the panel session produces to use later.

About this task

The installation CLIST allocates several data sets for input/output. From your TSO user ID, you should be able to allocate these data sets to the permanent or temporary unit names that are provided on installation panel DSNTIPA2. These devices can be defined by an esoteric device group.

Related concepts:

DASD Device Allocation (MVS Initialization and Tuning Guide)

Making the DB2 ISPF libraries available to TSO

Before you can invoke the CLIST, you need to make the DB2 ISPF libraries available to TSO.

© Copyright IBM Corp. 1982, 2012

Procedure

To make the DB2 ISPF libraries available to TSO, use one of the following approaches to concatenate the DB2 ISPF libraries to your normal allocations:

- To prepare for running only the DB2 installation CLIST, issue the following commands:

```
PROFILE WTP MSGID
ALLOCATE DDNAME(SYSEXEC) DSN('prefix.SDSNCLST') SHR REUSE
ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
    'ISP.SISPMENU') SHR REUSE
ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' +
    'ISP.SISPPENU') SHR REUSE
```

- To prepare for running the DB2 installation CLIST and the DB2 Interactive (DB2I) facility, issue the following commands. If you use Kanji DB2I panels, concatenate SDSNPFPK to ISPPLIB instead of SDSNPFPE.

```
PROFILE WTP MSGID
ALLOCATE DDNAME(SYSEXEC) DSN('prefix.SDSNCLST') SHR REUSE
ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
    'ISP.SISPMENU') SHR REUSE
ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' 'prefix.SDSNPFPE' +
    'ISP.SISPPENU') SHR REUSE
ALLOCATE DDNAME(ISPSLIB) DSN('prefix.SDSNSPFS' +
    'ISP.SISPSLIB' 'ISP.SISPSENU') SHR REUSE
```

  The PROFILE command provides complete error messages.

  The ALLOCATE command uses the default names of the libraries that contain the ISPF panels. These ISPF library names might be different at your site. To concatenate or merge existing libraries with them, put the library names in the list of names in parentheses after DSN with the largest block size first. (If two or more libraries have the same block size, you can list either one first.)

Updating the CLIST defaults input member: job DSNTIJXZ

Before you run the CLIST to migrate to Version 10, update an existing Version 10 subsystem, or install Version 10 using the settings of an existing Version 10 subsystem, update your defaults input member. This member contains the subsystem parameter and buffer pool settings from an existing subsystem, and it is used as input during your migration, update, or installation.

About this task

You can use job DSNTIJXZ to create an updated defaults input member, DSNTID*xx*, that contains the settings from your current subsystem. Running this job eliminates the need to manually compare settings in your current DSNTIJUZ job with the values that are defined in the defaults input member.

Before running this job, you might need to bind or refresh the package and plan for the DSNTXAZP program. The necessary bind commands are in installation job DSNTIJSG.

Procedure

To update the CLIST defaults input member:

1. Obtain the correct version of the DSNTIJXZ job.
   - If you are preparing to migrate to Version 10, use the Version 8 or Version 9.1 DSNTIJXZ job to create an updated migration input member for the Version 10 installation CLIST. For example, if you are migrating from Version 9.1, use the Version 9.1 DSNTIJXZ job.

- If you are preparing to update an existing Version 10 subsystem or install Version 10 using the settings of an existing Version 10 subsystem, use the Version 10 DSNTIJXZ job to create an updated input member.

2. Configure and run job DSNTIJXZ.
   - If you are preparing to migrate to Version 10, specify as input your existing Version 8 or Version 9.1 DSNTID*xx* member. This member is the one that was created when you installed or migrated to Version 8 or Version 9.1. If you do not have this member, specify as input the Version 8 or Version 9.1 DSNTIDXA member.

     For DB2 data sharing, verify that job DSNTIJXZ runs on the data sharing member that you are preparing to migrate and that the input to job DSNTIJXZ is the DSNTID*xx* member for that data sharing member.
   - If you are preparing to update an existing Version 10 subsystem or install Version 10 using the settings of an existing Version 10 subsystem, specify as input your existing Version 10 DSNTID*xx* member. This member is the one that was created when you installed or migrated to Version 10. If you do not have this member, specify as input the Version 10 DSNTIDXA member, or the DSNTIDXB member if you use SAP.

   Job DSNTIJXZ uses the DSNTXAZP tool to create a copy of the specified DSNTID*xx* member and update it with the current subsystem parameter settings, buffer pool settings, and other installation settings.

   Recommendation:
   - When you run the CLIST during migration to DB2 Version 10, specify the DSNTID*xx* member that is created by this job in the DATA SET(MEMBER) NAME field on panel DSNTIPA1.
   - When you run the DB2 Version 10 CLIST in UPDATE or INSTALL mode, specify the DSNTID*xx* member that is created by this job in the INPUT MEMBER NAME field on panel DSNTIPA1.

Related reference:

"DATA SET(MEMBER) NAME field" on page 269

"INPUT MEMBER NAME field" on page 272

DSNTXAZP tool

The DSNTXAZP tool updates a CLIST defaults input member (DSNTID*xx*) with the current subsystem parameter and buffer pool settings that are defined in a specified DB2 subsystem. This tool is invoked by job DSNTIJXZ.

The batch program requires the following arguments:

```
//STEP01   EXEC PGM=DSNTXAZP,PARM='ssid action'
```

ssid is the subsystem ID.

action is one of the following values:

UPDATE_ALL
: Creates a new CLIST defaults input member and updates it with the current subsystem parameter and buffer pool settings.

UPDATE_BPOOL
: Creates a new CLIST defaults input member and updates it with the current buffer pool settings only (no subsystem parameter changes).

UPDATE_ZPARM
Creates a new CLIST defaults input member and updates it with the current subsystem parameter settings only (no buffer pool changes).

The following DD statements are required for processing:

XAINPUT
Used as input to the DSNTXAZP program. It allocates the current CLIST defaults input member that will be used as the source for creating a new CLIST defaults input member. The version of the member should match the version of the selected DB2 subsystem.

XAOUTPUT
Used as output by the DSNTXAZP program. It allocates the generated CLIST defaults input member with updated subsystem parameter and buffer pool settings. This should be a member of a PDS that is defined with RECFM=FB, LRECL=80.

SYSPRINT
Used as output by the DSNTXAZP program for informational and error messages. It also contains a report on the parameters that were processed and modified. For additional details on the report, see the example below.

The following JCL is an example of running the tool:

```
//DSNTIXZ  EXEC PGM=DSNTXAZP,PARM='DSN UPDATE_ALL'
//XAINPUT  DD   DSN=DSN!!0.SDSNSAMP(DSNTIDXA),DISP=SHR
//XAOUTPUT DD   DSN=DSN!!0.SDSNSAMP(DSNTIDNU),DISP=SHR
//SYSPRINT DD   SYSOUT=*
```

The tool generates a CLIST input member with the current subsystem parameter settings, buffer pool settings, or both, and the following prolog:

```
* ********************************************************************
*  CREATED MEMBER = MY.DATASET.NAME(DSNTIDNU)                        *
*                                                                    *
*  CREATED ON = MAR 24, 2010 11:12:30                                *
*                                                                    *
*  SOURCE MEMBER = MY.SOURCE.NAME(DSNTIDXA)                          *
*                                                                    *
*  DESCRIPTIVE NAME = DB2 INSTALLATION DATA                          *
*                                                                    *
*  FUNCTION = PROVIDE UPDATED VALUES FOR MIGRATING TO A NEW RELEASE  *
*             OF DB2 OR INSTALLING A NEW RELEASE OF DB2              *
*                                                                    *
*  PARAMETERS:                                                       *
*    PARM NAME = NAME FOR ISPF PANEL VARIABLE                        *
*    PARM TYPE = TYPE FOR VALIDITY CHECKING                          *
*    DS SCP    = DATA SHARING SCOPE:  G = GROUP, M = MEMBER          *
*    MIN, MAX  = RANGE VALUES FOR VALIDITY CHECKING                  *
*    CURRENT VALUE = VALUE USED AS INPUT FOR NAMED ISPF VARIABLE     *
*                                                                    *
* ********************************************************************
* ************************************************* *
* PARM   PARM DS  MIN         MAX           CURRENT *
* NAME   TYPE SCP VALUE       VALUE         VALUE   *
* ************************************************* *
```

In addition to creating a new CLIST input member, the DSNTXAZP tool provides a report of all parameters. The report indicates whether each parameter was updated or if the original value from the input member was retained. It also shows the mapping between the subsystem parameter and the CLIST parameter. This is a sample report:

```
CLIST INPUT MEMBER GENERATION REPORT ** 2010-03-24 08:25:35
-----------------------------------------------------------

CLIST PARAMETER REPORT:

0001 PARAMETER NAME             = ABEXP
     ZPARM/BUFFERPOOL PARAMETER = ABEXP
     PARAMETER TYPE             = CHAR
     DATA SHARING SCOPE         = M
     MINIMUM VALUE              = NO
     MAXIMUM VALUE              = YES
     CURRENT CLIST VALUE        = YES
     CURRENT INSTALLED VALUE    = YES
     STATUS                     = RETAINED

0002 PARAMETER NAME             = ABIND
     ZPARM/BUFFERPOOL PARAMETER = ABIND
     PARAMETER TYPE             = CHAR
     DATA SHARING SCOPE         = M
     MINIMUM VALUE              = NONE
     MAXIMUM VALUE              = NONE
     CURRENT CLIST VALUE        = YES
     CURRENT INSTALLED VALUE    = NO
     STATUS                     = UPDATED
 :
nnnn PARAMETER NAME             = DB2REL
     ZPARM/BUFFERPOOL PARAMETER = N/A
     PARAMETER TYPE             = CHAR
     DATA SHARING SCOPE         = M
     MINIMUM VALUE              = NONE
     MAXIMUM VALUE              = NONE
     CURRENT CLIST VALUE        = 1010
     CURRENT INSTALLED VALUE    = N/A
     STATUS                     = RETAINED
  :

CHANGE SUMMARY REPORT:

  CLIST PARAMETER               ZPARM/BUFFERPOOL PARAMETER    VALUE
  ---------------------------------------------------------------------------------
  ABEXP                         -SAME-                        YES
* ABIND                         -SAME-                        NO (YES)
   :
* BP0_VPSEQT                    BP0 VPSEQT                    50 (35)
* BP0_DWQT                      BP0 DWQT                      50 (35)
* BP0_VDWQT                     BP0 VDWQT                     10,0 (20)
   :
  BUFMAX00                      BPO VPSIZE                    20000
   :
  DB2REL                        -NONE-                        1010
   :
* NUMCONCR                      CTHREAD                       220 (200)
  OVDSMAX                       DSMAX                         20000
   :
  ---------------------------------------------------------------------------------
  SOURCE MEMBER        : DSNA10.SDSNSAMP(DSNTIDXA)
  CREATED MEMBER       : DB2.CLIST.MEMBER(DSNTIDX0)
  PARAMETERS PROCESSED : n
  PARAMETERS MODIFIED  : m
  ---------------------------------------------------------------------------------
```

Multiple procedure libraries

If your z/OS system has multiple procedure libraries, ensure that the installation jobs access the appropriate set of procedures by specifying the order in which procedure libraries are searched.

You can use the following JCLLIB statement to specify the order in which procedure libraries are searched.

```
//ddname JCLLIB ORDER=(library[,library...])
```

The JCLLIB statement must follow the JOB statement and precede the first EXEC statement in the job. You can have DB2 insert this statement in your JCL for you by entering the statement on installation panel DSNTIPY.

Related reference:

"Job editing panel: DSNTIPY" on page 488

JCLLIB statement (MVS JCL Reference)

System affinity for installation jobs

You must ensure that the installation jobs run on the z/OS system where the appropriate DB2 subsystem is running.

The following methods are a couple of the z/OS installation-specific ways to establish system affinity:

- For JES2 multi-access spool (MAS) systems, use the following JCL statement:

  ```
  /*JOBPARM SYSAFF=cccc
  ```

  Where *cccc* is the JES2 name. You can specify an asterisk (SYSAFF=*) to indicate that the job should run on the system from which it was submitted.
- For JES3 systems, use the following JCL statement:

  ```
  //*MAIN SYSTEM=(main-name)
  ```

  The *main-name* is the JES3 name.

You can edit the jobs manually, or you can enter the preceding statements on installation panel DSNTIPY and have DB2 insert these statements for you.

Your installation might have other mechanisms for controlling where batch jobs run, such as by using job classes.

Related reference:

"Job editing panel: DSNTIPY" on page 488

JES2 Control Statements (MVS JCL Reference)

JES3 Control Statements (MVS JCL Reference)

Invoking the CLIST

After you make the DB2 ISPF libraries available to TSO, you can invoke the CLIST.

Procedure

To invoke the CLIST:

1. Check your TSO logon region size. Usually 2 MB is enough.
2. Invoke ISPF.
3. Select option 6 on the main ISPF panel.
4. Invoke the installation CLIST.
5. Enter one of the following specifications:

   ```
   EXEC 'prefix.SDSNCLST(DSNTINST)'
   EXEC 'prefix.SDSNCLST(DSNTINST)' 'CONTROL(LIST)'
   EXEC 'prefix.SDSNCLST(DSNTINST)' 'CHKOUTDS(NO)'
   ```

Use the second specification above if you want to receive the messages that trace the progress of the CLIST.

Use the third specification above if you want to use the CLIST but skip the verification of output data sets. By default, the CLIST verifies the following information when installation panels DSNTIPT and DSNTIPA1 are displayed:

- The CLIST verifies that the TEMP CLIST LIBRARY and SAMPLE LIBRARY that you specify on DSNTIPT can be allocated and opened for output.
- If you specify a value for OUTPUT MEMBER NAME on DSNTIPA1, the CLIST verifies that the data set *prefix*.SDSNSAMP can be allocated and opened for output.

General instructions for using the installation CLIST

The CLIST reads a set of default values and displays them on the panels. The values can be the original default values that are supplied by IBM or the values that you created in a previous run of the CLIST.

```
DSNT4781 BEGINNING EDITED DATA SET OUTPUT.
```

In general, you can change the default values to match the needs of your organization. However, some values cannot be changed, depending on whether you run the CLIST in INSTALL, MIGRATE, ENFM, or UPDATE mode. At any point while you run the CLIST, you can save your changes in the output member that is specified on panel DSNTIPA1. To save your changes, enter SAVE in the command line of any installation panel, and choose option 1. The following panel is displayed when you enter SAVE in the command line:

```
+--------------------------------+
|DSNTIPSV                        |
| Save changes in progress:      |
|                                |
| Select one.                    |
|    _ 1. Save and continue      |
|      2. Save and exit          |
|      3. Continue without saving |
|                                |
| PRESS: ENTER to continue       |
| RETURN to exit                 |
+--------------------------------+
```

Figure 8. DSNTIPSV

Related tasks:

"Make DB2 CLISTs available to TSO and batch users (DSNTIJVC)" on page 522

"Make DB2 load modules available to TSO and batch users" on page 584

Output from the panel session

As output, the CLIST produces two new data sets and a data set member.

Specifically, the panel session produces:

- A new data set member, if specified, that contains the resulting parameter values from the session. This member is stored in *prefix*.SDSNSAMP.
- A new data set, *prefix*.NEW.SDSNSAMP, that contains the edited JCL with the values that you entered on the panels.
- A new data set, *prefix*.NEW.SDSNTEMP, that contains tailored CLISTs for input to job DSNTIJVC, which is run during installation or migration.

Use job DSNTIJVC to combine the CLISTs into a common data set.

DB2 performs validity checking of the values that you enter during the panel sessions. If you receive an ISPF error message, press the HELP key for additional information.

The following figure illustrates by examples how the CLIST works during installation.

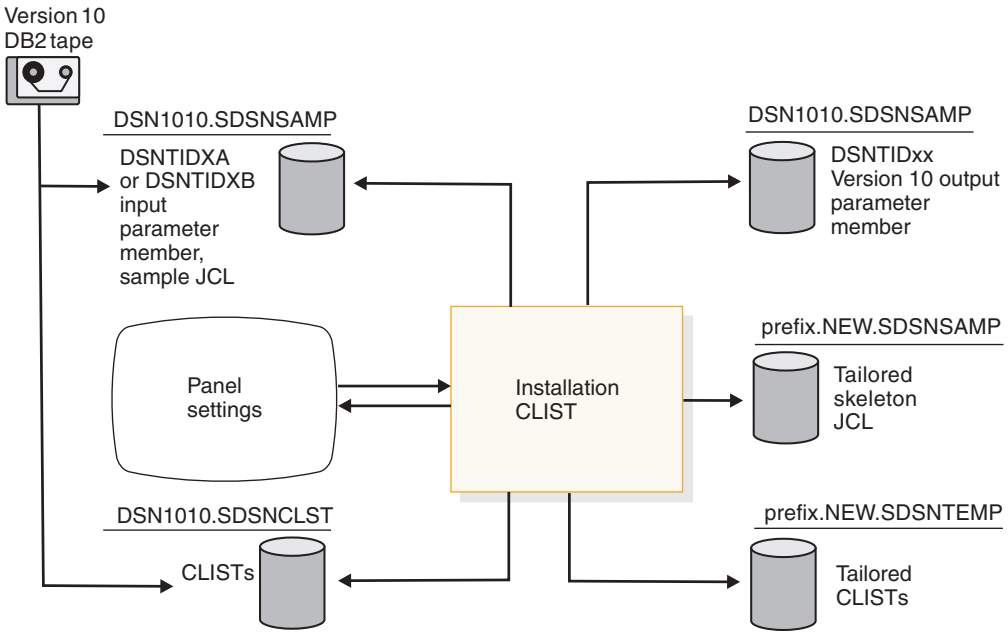


Figure 9. Examples of input to and output from the installation CLIST during installation

The following figure illustrates how the CLIST works during migration to conversion mode (from Version 8 or Version 9.1).

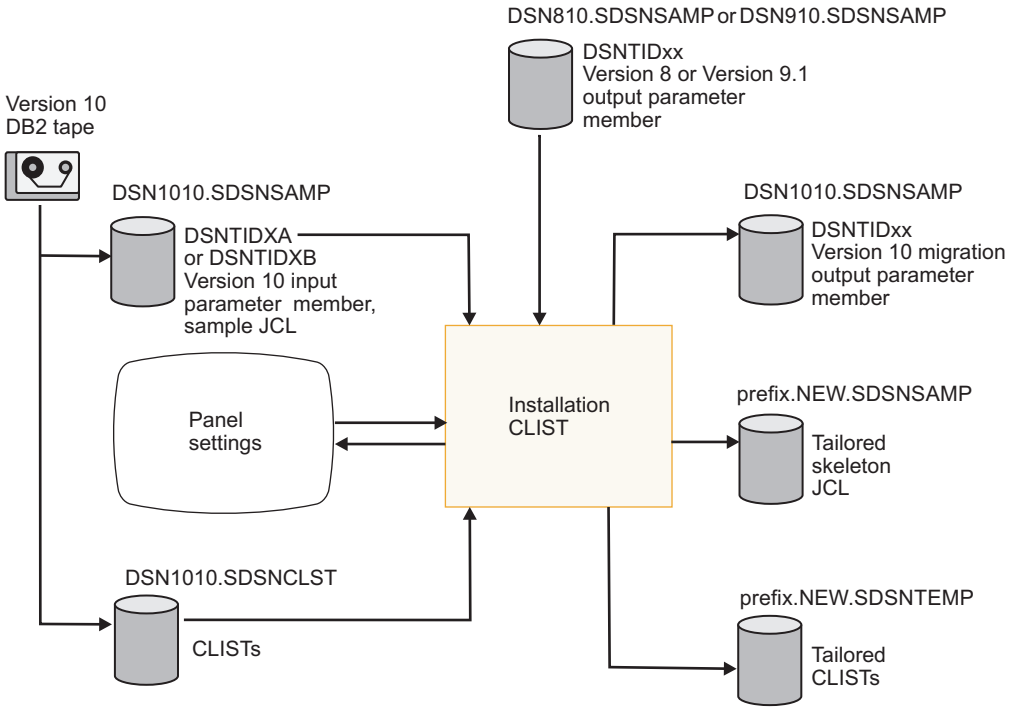


Figure 10. Examples of input to and output from the installation CLIST during migration to conversion mode (from Version 8 or Version 9.1)

The following figure illustrates how the CLIST works during migration to new-function mode.

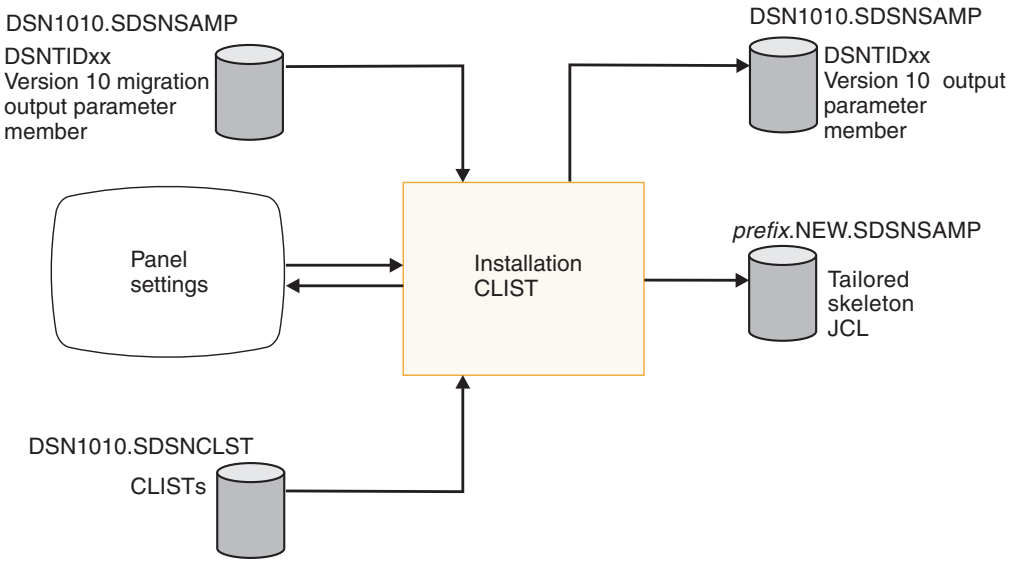


Figure 11. Examples of input to and output from the installation CLIST during migration to new-function mode

Actions that are allowed on panels

All panel sequences begin with the Main Panel (DSNTIPA1).

Preparation: After the description of each parameter, record your choice for a value before you actually use the panels.

Panels that have fields that are marked with asterisks contain values that are primed on the basis of values from a previous panel. The following message is found on these panels:

```
DSNT444I SCROLLING BACKWARD MAY CHANGE FIELDS MARKED WITH ASTERISKS
```

If you scroll back to the panel that has the original value, the values on the succeeding panels are refreshed **only** if the original value is changed. If the values are changed, the following message is displayed:

```
DSNT443I VALUES MARKED WITH AN ASTERISK HAVE BEEN UPDATED
```

For example, panel DSNTIPH has fields that are marked with asterisks indicating values that are primed on the basis of the CATALOG ALIAS value on installation panel DSNTIPA2.

Data entry: Enter your choice on a panel in the space that is marked by an arrow (===>). Begin your entry in the second position to the right of the arrow. (The first position is protected; you cannot write in it.)

Panel IDs: If you want the panel IDs to appear on each panel, enter the following command from any panel:

```
PANELID ON
```

Panel descriptions

Several terms are commonly used in the panel descriptions and panel field descriptions.

These terms include the following:

Scrolling installation panels: The installation panels enable you to scroll back to previous panels to review or change values. The **END** key (usually **PF3**) takes you back to the previous panel. Pressing Enter continues to validate entries in the current panel and displays the next panel. If you want to exit completely from the installation process, use the **RETURN** key (usually **PF4**).

Acceptable values: This part of a panel field description gives you the range of allowable values or the list of allowable values for an installation panel field. The acceptable values apply to the installation panel field, but not necessarily to the subsystem parameter that is associated with that field. Differences might occur if a panel field is used to determine the value of the associated subsystem parameter and the values do not correspond exactly. If the maximum allowable value of a panel field is greater than 1024, in most cases you can use the equivalent K value. (The CLIST automatically multiplies the K value by 1024.) If the maximum allowable value is greater than 1,048,576, in most cases you can use the equivalent M value. (The CLIST automatically multiplies the M value by 1,048,576.) If the maximum allowable value is greater than 1024 MB, in most cases you can use the equivalent G value. If the maximum allowable value is greater than 1024 GB, in most cases you can use the equivalent T value. The maximum acceptable values might be too large for smaller systems; therefore, ensure that the values that you enter are valid for the size of your system.

Default: The default values that are shown on the panel topics are the original values that are supplied by IBM. If you previously ran the CLIST and saved the updated panel values in a DSNTID*xx* data set that you are now using as input, the previously entered values now appear as default values. Panel values that are modified outside of the installation process are not saved in the DSNTID*xx* data set and are not reflected on the panels.

Update: This information identifies a corresponding field on the Update Selection Panel (DSNTIPB) or refers to information that provides instructions for updating the field value.

dsnhdecp: These are the names of the parameters in the application defaults load module *dsnhdecp*.

DSNZPxxx: These are the names of the parameters in the data-only load module DSNZP*xxx*.

Directory of panel field names

The installation panel fields enable you to specify subsystem parameters and other values that are used during installation or migration.

The panel fields are listed in alphabetical order in the following table.

Table 51. Panel fields

Panel field name	Panel	See
32K SORT PRIMARY SPACE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
32K SORT WORK SEG SIZE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
32K SORT WORK TS'S	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
32K TEMP PRIMARY SPACE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
32K TEMP WORK SEG SIZE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
32K TEMP WORK TS'S	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K SORT PRIMARY SPACE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K SORT WORK SEG SIZE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K SORT WORK TS'S	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K TEMP PRIMARY SPACE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K TEMP WORK SEG SIZE	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
4K TEMP WORK TS'S	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
ACCESS CONTROL	DSNTIPO3	"Default startup modules panel: DSNTIPO3" on page 358
ADMIN SCHEDULER	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
ADVANCED CONFIGURATION OPTIONS	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
AGGREGATION FIELDS	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
ALLOCATION UNITS	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
APPL DEFAULTS	DSNTIPO3	"Default startup modules panel: DSNTIPO3" on page 358

Table 51. Panel fields (continued)

Panel field name	Panel	See
APPL REGISTRATION TABLE	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
APPLICATION DBRM	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
APPLICATION ENCODING	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
APPLICATION LOAD	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
ARCHIVE LOG FREQ	DSNTIPL	"Active log data set parameters: DSNTIPL" on page 426
ARCHIVE LOG RACF	DSNTIPP	"Protection panel: DSNTIPP" on page 410
ART/ORT ESCAPE CHARACTER	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
ASCII CCSID	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
ASSISTANT	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
AUDIT TRACE	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
AUTH EXIT LIMIT	DSNTIPP	"Protection panel: DSNTIPP" on page 410
AUTH MEMBER	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
AUTH SEQUENCE	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
AUTO BIND	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
AUTO START	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
BACKOUT DURATION	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
BIF COMPATIBILITY	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
BIND NEW PACKAGE	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
BLOCK SIZE	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
BP0 - BP29	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
BP30 - BP49	DSNTIP2	"Buffer pool sizes panel 2: DSNTIP2" on page 342
BP8K0-BP8K9	DSNTIP2	"Buffer pool sizes panel 2: DSNTIP2" on page 342
BP16K0-BP16K9	DSNTIP2	"Buffer pool sizes panel 2: DSNTIP2" on page 342
BP32K-BP32K9	DSNTIP2	"Buffer pool sizes panel 2: DSNTIP2" on page 342
BUFFER POOL SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
C/CPP COMPILER LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
C/CPP COMPILER MODULE	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
C/CPP HEADER LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300

Table 51. Panel fields (continued)

Panel field name	Panel	See
C/370 COMPILER MESSAGES	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CACHE DYNAMIC SQL	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
CATALOG ALIAS	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATALOG DATA	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
CHECKPOINT TYPE	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
CICS COBOL LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
CICS EXCI LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
CICS LOAD LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
CICS MACRO LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
CICS PL/I LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
CLIST LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
CLIST ALLOCATION	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
COBOL COMPILER LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CODE STORAGE SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
COLUMNS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
COMMAND PREFIX	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
COMMAND SCOPE	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
COMPACT DATA	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
COMPRESS SMF RECS	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
COMPRESS SPT01	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CONTRACT THREAD STG	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
CONTROL ALL APPLICATIONS	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
COORDINATOR	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
COPY	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
COPY 1 NAME	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290
COPY 2 NAME	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290
COPY 1 PREFIX	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290
COPY 2 PREFIX	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290

Table 51. Panel fields (continued)

Panel field name	Panel	See
CPP AUTO CALL LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CPP CLASS LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CPP CLASS LIB HEADERS	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CPP PROCEDURE LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
CURRENT DEGREE	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
CURRENT MAINT TYPES	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
CURRENT REFRESH AGE	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
DATA SET(MEMBER) NAME	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
DATA SET NAME PREFIX	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
DATA SET NAME SUFFIX	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
DATA SET STORAGE SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
DATA SHARING	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
DATABASES	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
DATABASES	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
DATASET STATS TIME	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
DATE FORMAT	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
DBADM CREATE AUTH	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
DBRM LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
DB2 GENERIC LUNAME	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DB2 LOCATION NAME	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DB2 NETWORK LUNAME	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DB2 NETWORK PASSWORD	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DDF STARTUP OPTION	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DDF THREADS	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
DDF/RRSAF ACCUM	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
DEADLOCK CYCLE	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
DEADLOCK TIME	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
DEALLOC PERIOD	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
DECFLOAT ROUNDING MODE	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
DECIMAL ARITHMETIC	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372

Table 51. Panel fields (continued)

Panel field name	Panel	See
DECIMAL POINT IS	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
DECLARATION LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
DEF ENCODING SCHEME	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
DEFAULT 4-K BUFFER POOL FOR USER DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT 8-K BUFFER POOL FOR USER DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT 16-K BUFFER POOL FOR USER DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT 32-K BUFFER POOL FOR USER DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT BUFFER POOL FOR USER INDEXES	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT BUFFER POOL FOR USER LOB DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT BUFFER POOL FOR USER XML DATA	DSNTIP1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
DEFAULT PARTITION SEGSIZE	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
DEFAULT TEMPLATE	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
DEFINE CATALOG	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
DEFINE DATA SETS	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
DEL CF STRUCTS	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
DESCRIBE FOR STATIC	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
DEVICE TYPE 1	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
DEVICE TYPE 2	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
DIRECTORY AND CATALOG DATA	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
DIRECTORY AND CATALOG INDEXES	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
DIST SQL STR DELIMTR	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
DISCONNECT IRLM	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
DL/I BATCH TIMEOUT	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
DPROP SUPPORT	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
DRDA PORT	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
DSMAX	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
DUMP CLASS NAME	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
EBCDIC CCSID	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
EDM DBD CACHE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490

Table 51. Panel fields (continued)

Panel field name	Panel	See
EDM SKELETON POOL SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
EDM STATEMENT CACHE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
EDMPOOL LIMIT BELOW THE BAR	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
EVALUATE UNCOMMITTED	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
EXECUTED STMTS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
EXIT LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
EXPLAIN PROCESSING	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
EXTENDED SECURITY	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
EXTRA BLOCKS REQ	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
EXTRA BLOCKS SRV	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
FAST REPLICATION	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FAST RESTORE	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY PPRC	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FOR DB2 JAVA ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR DB2 MQSERIES ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR DB2 REXX ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR DB2 UTILITIES ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR DB2 WEB SERVICES ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR DB2 XML PROCESSING ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR MOST DB2 JAVA ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR MOST DB2 ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR ROUTINES IDENTIFIED TO PROGRAM CONTROL	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR SERIALIZED ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR THE DB2 CICS ROUTINE	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FOR UNIFIED DEBUGGER ROUTINES	DSNTIPR1	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
FORTRAN COMPILER LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
FORTRAN LINK EDIT LIB	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
FROM RELEASE	DSNTIPA1	"Main panel: DSNTIPA1" on page 265

Table 51. Panel fields (continued)

Panel field name	Panel	See
GDDM LOAD MODULES	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
GDDM MACLIB	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
GRANT EXECUTE	DSNTIPRB - DSNTIPRP	"DB2 routine selection panel: DSNTIPRA" on page 462
GROUP ATTACH	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
GROUP NAME	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
HIGH LEVEL ASSEMBLER LIB	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
IBM LE RUNTIME LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
IBM LE LINK EDIT LIB	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
IBM LE PRELINK MSG LIB	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
IDENTIFY/AUTH	DSNTIPO3	"Default startup modules panel: DSNTIPO3" on page 358
IDLE THREAD TIMEOUT	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
IGNORE SORTNUM STAT	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
IMMEDIATE WRITE	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
IMPLICIT TIME ZONE	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
IMS BMP TIMEOUT	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
IMS RESLIB	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
INCLUDE LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
INDEX SPACE ALLOCATION	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
INPUT MEMBER NAME	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
INSTALL DD CONTROL SUPT.	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
INSTALL IRLM	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
INSTALL TYPE	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
IRLM LOAD LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
IRLM LOCK MAXIMUM SPACE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
IRLM XCF GROUP NAME	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
ISPF ISPLINK MODULE	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
IVP DATA LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
LANGUAGE DEFAULT	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
LEVELID UPDATE FREQ	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
LIBRARY NAME PREFIX	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
LIBRARY NAME SUFFIX	DSNTIPA1	"Main panel: DSNTIPA1" on page 265

Table 51. Panel fields (continued)

Panel field name	Panel	See
LICENSE TERMS ACCEPTED	DSNTIPO2[1]	"Notice and acceptance of OTC license panel: DSNTIPO2" on page 275
LIMIT BACKOUT	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
LINK LIST ENTRY	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
LINK LIST LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
LINK LIST SEQUENCE	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
LOAD	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
LOAD DISTRIBUTION	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
LOAD LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
LOB INLINE LENGTH	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
LOCAL DATE LENGTH	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
LOCALE LC_CTYPE	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
LOCAL TIME LENGTH	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
LOCK ENTRY SIZE	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
LOCKS PER TABLE(SPACE)	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
LOCKS PER USER	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
LOG COPY 1, BSDS 2	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
LOG COPY 2, BSDS 1	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
LONG-RUNNING READER	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MACRO LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
MANAGE THREAD STORAGE	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX ABEND COUNT	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
MAX BATCH CONNECT	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX DATA CACHING	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
MAX DEGREE	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
MAX INACTIVE DBATS	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
MAX KEPT DYN STMTS	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX OPEN CURSORS	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
MAX OPEN FILE REFS	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX REMOTE ACTIVE	DSNTIPE	"Thread management panel: DSNTIPE" on page 333

Table 51. Panel fields (continued)

Panel field name	Panel	See
MAX REMOTE CONNECTED	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX STORAGE FOR LOCKS	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
MAX STORED PROCS	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
MAX TEMP RID	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
MAX TEMP STG/AGENT	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
MAX TSO CONNECT	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAX USERS	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
MAXIMUM LE TOKENS	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
MAXIMUM TAPE UNITS	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
MEMBER IDENTIFIER	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
MEMBER NAME	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
MINIMUM DIVIDE SCALE	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
MINUTES/CHECKPOINT	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
MIXED DATA	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
MONITOR SIZE	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
MONITOR TRACE	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
NON-VSAM DATA	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
NUMBER OF COPIES	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290
NUMBER OF LOCK ENTRIES	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
NUMBER OF LOGS	DSNTIPL	"Active log data set parameters: DSNTIPL" on page 426
NUMBER OF TCBS	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
OBJT REGISTRATION TABLE	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
OPTIMIZATION HINTS	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
OPTIMIZE EXTENT SIZING	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
OTC LICENSE USAGE	DSNTIPO1[1]	"OTC license usage panel: DSNTIPO1" on page 274
OUTPUT BUFFER	DSNTIPL	"Active log data set parameters: DSNTIPL" on page 426
OUTPUT MEMBER NAME	DSNTIPA1	"Main panel: DSNTIPA1" on page 265
PACKAGE AUTH CACHE	DSNTIPP	"Protection panel: DSNTIPP" on page 410
PACKAGE LISTS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
PACKAGE OWNER	DSNTIPRB - DSNTIPRP	"DB2 routine selection panel: DSNTIPRA" on page 462
PACKAGE STATEMENTS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313

Table 51. Panel fields (continued)

Panel field name	Panel	See
PACKAGES	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
PAD INDEXES BY DEFAULT	DSNTIPE	"Thread management panel: DSNTIPE" on page 333
PAD NUL-TERMINATED	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
PAGE PROTECT	DSNTIPJ	"IRLM panel 2: DSNTIPJ" on page 391
PARALLELISM EFFICIENCY	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
PARAMETER MODULE	DSNTIPO3	"Default startup modules panel: DSNTIPO3" on page 358
PERMANENT UNIT NAME	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
PLAN AUTH CACHE	DSNTIPP	"Protection panel: DSNTIPP" on page 410
PLAN MANAGEMENT	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
PLAN MANAGEMENT SCOPE	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
PLAN STATEMENTS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
PLANS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
PL/I COMPILER LIBRARY	DSNTIPU	"Data set names panel 2: DSNTIPU" on page 300
POOL THREAD TIMEOUT	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
PRIMARY QUANTITY	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
PROC NAME	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
QUIESCE PERIOD	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
RANDOM ATTACH	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
RANDOMIZE XML DOCID	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
READ COPY2 ARCHIVE	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
READ TAPE UNITS	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
REAL TIME STATS	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
REBUILD INDEX	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
RECALL DATABASE	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
RECALL DELAY	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
RECORDING MAX	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
RECORDS/CHECKPOINT	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
REGISTRATION DATABASE	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
REGISTRATION OWNER	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
REMOTE LOCATION	DSNTIPY	"Job editing panel: DSNTIPY" on page 488
REORDERED ROW FORMAT	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318

Table 51. Panel fields (continued)

Panel field name	Panel	See
REORG INDEX	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
REORG TABLESPACE	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
REQUIRE FULL NAMES	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485
RESOURCE AUTHID	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
RESOURCE TIMEOUT	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
RESTORE/RECOVER	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
RESYNC INTERVAL	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
RESYNC PORT	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
RETAINED LOCK TIMEOUT	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
RETENTION PERIOD	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
REVOKE DEP PRIV	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
RID POOL SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
RLF AUTO START	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
RLST ACCESS ERROR	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
RLST ACCESS ERROR	DSNTIPR	"Distributed data facility panel 1: DSNTIPR" on page 442
RLST NAME SUFFIX	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
RO SWITCH CHKPTS	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
RO SWITCH TIME	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
ROUTINE AUTH CACHE	DSNTIPP	"Protection panel: DSNTIPP" on page 410
ROUTINES CREATOR	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SAMPLE LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
SEC ADMIN 1 TYPE	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SEC ADMIN 2 TYPE	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SEC DEF CREATOR	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SECONDARY QTY	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
SECURE PORT	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
SECURITY ADMIN 1	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SECURITY ADMIN 2	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412

Table 51. Panel fields (continued)

Panel field name	Panel	See
SEPARATE SECURITY	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SEPARATE WORK FILES	DSNTIP9	"Work file database panel: DSNTIP9" on page 323
SET CHECK PENDING	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
SIGNON	DSNTIPO3	"Default startup modules panel: DSNTIPO3" on page 358
SINGLE VOLUME	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
SITE TYPE	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
SKIP UNCOMM INSERTS	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
SMF ACCOUNTING	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
SMF STATISTICS	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
SORT LIBRARY	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
SORT POOL SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
SPT01 INLINE LENGTH	DSNTIPA2	"DB2 catalog and directory panel: DSNTIPA2" on page 275
SQL STRING DELIMITER	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
STAR JOIN QUERIES	DSNTIP8	"Performance and optimization panel: DSNTIP8" on page 379
START IRLM CTRACE	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
STATISTICS HISTORY	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
STATISTICS ROLLUP	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
STATISTICS SYNC	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
STATISTICS TIME	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
STD SQL LANGUAGE	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
STRING DELIMITER	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
SUBGRP ATTACH	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
SUBSYSTEM MEMBER	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
SUBSYSTEM NAME	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
SUBSYSTEM NAME	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
SUBSYSTEM SEQUENCE	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
SUPPRESS SOFT ERRORS	DSNTIPM	"MVS PARMLIB updates panel: DSNTIPM" on page 422
SYSTEM ADMIN 1	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412

Table 51. Panel fields (continued)

Panel field name	Panel	See
SYSTEM ADMIN 2	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SYSTEM-LEVEL BACKUPS	DSNTIP6	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
SYSTEM LOB VAL STG	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
SYSTEM MACLIB	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
SYSTEM OPERATOR 1	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SYSTEM OPERATOR 2	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
SYSTEM PROCEDURES	DSNTIPW	"Data set names panel 3: DSNTIPW" on page 309
SYSTEM XML VAL STG	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
TABLES	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
TABLES IN STMT	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
TABLE SPACE ALLOCATION	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
TABLE SPACES	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
TCP/IP ALREADY VERIFIED	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
TCP/IP KEEPALIVE	DSNTIP5	"Distributed data facility panel 2: DSNTIP5" on page 448
TEMP CLIST LIBRARY	DSNTIPT	"Data set names panel 1: DSNTIPT" on page 295
TEMP DS UNIT NAME	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
TEMPORARY UNIT NAME	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
TIME FORMAT	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
TIME TO AUTOSTART	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
TIMEOUT VALUE	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
TIMESTAMP ARCHIVES	DSNTIPH	"System resource data set names panel: DSNTIPH" on page 290
TOTAL MAIN STORAGE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
TOTAL STORAGE BELOW 16MB	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
TRACE AUTO START	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
TRACE SIZE	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
TRACKER SITE	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
U LOCK FOR RR/RS	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386
UNICODE CCSID	DSNTIPF	"Application programming defaults panel 1: DSNTIPF" on page 362
UNICODE IFCIDS	DSNTIPN	"Tracing parameters panel: DSNTIPN" on page 343
UNKNOWN AUTHID	DSNTIPP1	"Protection panel 2: DSNTIPP1" on page 412
UNREGISTERED DDL DEFAULT	DSNTIPZ	"Data definition control support panel: DSNTIPZ" on page 485

Table 51. Panel fields (continued)

Panel field name	Panel	See
UPDATE RATE	DSNTIPL	"Active log data set parameters: DSNTIPL" on page 426
UR CHECK FREQ	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
UR LOG WRITE CHECK	DSNTIPL1	"Checkpoint parameters: DSNTIPL1" on page 428
USE DATA COMPRESSION	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
USE FOR DYNAMICRULES	DSNTIP4	"Application programming defaults panel 2: DSNTIP4" on page 372
USE PROTECTION	DSNTIPP	"Protection panel: DSNTIPP" on page 410
USER LOB VALUE STG	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
USER XML VALUE STG	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
UT DB2 SORT USE	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
UT SORT DATA SET ALLOCATION	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
UTIL TEMP STORCLAS	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
UTILITY TIMEOUT	DSNTIP61	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
VARY DS CONTROL INTERVAL	DSNTIP7	"Sizes panel 2: DSNTIP7" on page 318
VIEWS	DSNTIPD	"Sizes panel 1: DSNTIPD" on page 313
VSAM CATALOG, DEFAULT AND WORK FILE DATABASE	DSNTIPA3	"Data parameters panel: DSNTIPA3" on page 281
WLM ENV NAME	DSNTIPRB - DSNTIPRP	"DB2 routine selection panel: DSNTIPRA" on page 462
WLM ENVIRONMENT	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
WLM PROC NAME	DSNTIPX	"Routine parameters panel: DSNTIPX" on page 453
WORK FILE DB	DSNTIPK	"Define group or member panel: DSNTIPK" on page 286
WORKING STORAGE SIZE	DSNTIPC	"CLIST calculations panel 1: DSNTIPC" on page 490
WRITE TO OPER	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
WTO ROUTE CODES	DSNTIPO	"Operator functions panel: DSNTIPO" on page 351
WTOR ROUTE CODE	DSNTIPA	"Archive log data set parameters panel: DSNTIPA" on page 434
X LOCK FOR SEARCHED U/D	DSNTIPI	"IRLM panel 1: DSNTIPI" on page 386

Note:

1. The DSNTIPO1 and DSNTIPO2 panels display only if you bought the OTC license for DB2. Otherwise, these panels are not displayed.

Directory of subsystem parameters and application default values

Most subsystem parameters are associated with a macro and an installation panel.

The following table displays each macro parameter, and the macro where it is located. The table also shows the name of the installation panel that is associated with each parameter, whether each parameter can be updated online, and

corresponding topics. Online update capability does not apply to macro DSNHDECP; therefore, update values are not listed for those parameters.

Some parameters, when updated online, result in a change in system behavior. These parameters include:

- CACHEDYN
- CACHEDYN_FREELOCAL
- MAXKEEPD
- PARTKEYU
- SYSADM/SYSADM2
- XLKUPDLT

Table 52. Directory of subsystem parameters and application default values

Parameter	Macro	Panel	Update online	Security parameter	Topic
ABEXP	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
ABIND	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
ACCEL	DSN6SPRM	—	No	No	"Subsystem parameters that are not on installation panels" on page 253
ACCESS_CNTL_MODULE	DSN6SYSP	DSNTIPO3	No	No	"Default startup modules panel: DSNTIPO3" on page 358
ACCUMACC	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
ACCUMUID	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
ADMTPROC	DSN6SPRM	DSNTIPX	No	No	"Routine parameters panel: DSNTIPX" on page 453
AEXITLIM	DSN6SPRM	DSNTIPP	Yes	No	"Protection panel: DSNTIPP" on page 410
AGCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
ALCUNIT	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
ALL/dbname	DSN6SPRM	DSNTIPS	—	No	"Databases and spaces to start automatically panel: DSNTIPS" on page 441
AMCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
APPENSCH	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
ARCPFX1	DSN6ARVP	DSNTIPH	Yes	No	"System resource data set names panel: DSNTIPH" on page 290
ARCPFX2	DSN6ARVP	DSNTIPH	Yes	No	"System resource data set names panel: DSNTIPH" on page 290
ARCRETN	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
ARCWRTC	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
ARCWTOR	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
ARC2FRST	DSN6LOGP	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
ASCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
ASSIST	DSN6GRP	DSNTIPK	No	No	"Define group or member panel: DSNTIPK" on page 286

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
AUDITST	DSN6SYSP	DSNTIPN	No	Yes	"Tracing parameters panel: DSNTIPN" on page 343
AUTH	DSN6SPRM	DSNTIPP	No	Yes	"Protection panel: DSNTIPP" on page 410
AUTHCACH	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
BACKODUR	DSN6SYSP	DSNTIPL1	No	No	"Checkpoint parameters: DSNTIPL1" on page 428
BIF_COMPATIBILITY	DSN6SPRM	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453
BINDNV	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
BLKSIZE	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
BMPTOUT	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
CACHEDYN	DSN6SPRM	DSNTIP8	Yes	Yes	"Performance and optimization panel: DSNTIP8" on page 379
CACHEDYN_FREELOCAL	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
CACHEPAC	DSN6SPRM	DSNTIPP	No	Yes	"Protection panel: DSNTIPP" on page 410
CACHERAC	DSN6SPRM	DSNTIPP	No	Yes	"Protection panel: DSNTIPP" on page 410
CATALOG	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
CATALOG	DSN6SPRM	DSNTIPA2	—	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATDDACL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATDMGCL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATDSTCL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATXDACL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATXMGCL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CATXSTCL	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CDSSRDEF	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
CHECK_FASTREPLICATION	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
CHECK_SETCHKP	DSN6SPRM	DSNTIP61	Yes	No	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
CHGDC	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
CHKFREQ	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
CHKLOGR	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
CHKMINS	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
CHKTYPE	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
CMTSTAT	DSN6FAC	DSNTIPR	No	No	"Distributed data facility panel 1: DSNTIPR" on page 442

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
COMPACT	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
COMPRESS_SPT01	DSN6SPRM	DSNTIPA2	Yes	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
CONDBAT	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
CONTSTOR	DSN6SPRM	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
COORDNTR	DSN6GRP	DSNTIPK	No	No	"Define group or member panel: DSNTIPK" on page 286
CTHREAD	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
DBACRVW	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
DATE	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
DATELEN	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
DB2SORT	DSN6SPRM	DSNTIP61	Yes	No	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
DB2SUPLD	DSNHDECP	—	—	No	note 1
DDF	DSN6FAC	DSNTIPR	No	No	"Distributed data facility panel 1: DSNTIPR" on page 442
DDF_COMPATIBILITY	DSN6FAC	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
DDLTOX	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
DEALLCT	DSN6LOGP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
DECARTH	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
DECDIV3	DSN6SPRM	DSNTIP4	No	No	"Application programming defaults panel 2: DSNTIP4" on page 372
DECIMAL	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
DEF_DECFLOAT_ROUND_MODE	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
DEFLANG	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
DEFLTID	DSN6SPRM	DSNTIPP	No	No	"Protection panel: DSNTIPP" on page 410
DEL_CFSTRUCTS_ON_RESTART	DSN6SYSP	DSNTIPK	NO	No	"Define group or member panel: DSNTIPK" on page 286
DELIM	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
DESCSTAT	DSN6SPRM	DSNTIP4	Yes	No	"Application programming defaults panel 2: DSNTIP4" on page 372
DISABLE_EDMRTS	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
DISABSCL	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
DISALLOW_DEFAULT_COLLID	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
DLDFREQ	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
DLITOUT	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
DPSEGSZ	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
DSHARE	DSN6GRP	DSNTIPA1	No	No	"Main panel: DSNTIPA1" on page 265
DSMAX	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
DSQLDELI	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
DSSTIME	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
DSVCI	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
DYNRULS	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
EDM_SKELETON_POOL	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
EDMDBDC	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
EDMPOOL	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
EDMSTMTC	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
EDPROP	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
EN_PJSJ	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
ENSCHEME	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
EVALUNC	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
EXTRAREQ	DSN6SYSP	DSNTIP5	Yes	No	"Distributed data facility panel 2: DSNTIP5" on page 448
EXTRASRV	DSN6SYSP	DSNTIP5	Yes	No	"Distributed data facility panel 2: DSNTIP5" on page 448
EXTSEC	DSN6SYSP	DSNTIPR	Yes	Yes	"Distributed data facility panel 1: DSNTIPR" on page 442
FCCOPYDDN	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_COPY	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_LOAD	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_PPRC	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_REBUILD_INDEX	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_REORG_INDEX	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
FLASHCOPY_REORG_TS	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
GCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
GRPNAME	DSN6GRP	DSNTIPK	No	No	"Define group or member panel: DSNTIPK" on page 286
HONOR_KEEPDICTIONARY	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
IDAUTH_MODULE	DSN6SYSP	DSNTIPO3	No	No	"Default startup modules panel: DSNTIPO3" on page 358
IDBACK	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
IDFORE	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
IDTHTOIN	DSN6FAC	DSNTIPR	Yes	No	"Distributed data facility panel 1: DSNTIPR" on page 442
IDXBPOOL	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
IGNSORTN	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
IMMEDWRI	DSN6GRP	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
IMPDSDEF	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
IMPDSSIZE	DSN6SYSP	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
IMPLICIT_TIMEZONE	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
IMPTKMOD	DSN6SYSP	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
IMPTSCMP	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
INDEX_IO_PARALLELISM	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
INLISTP	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
IRLMAUT	DSN6SPRM	DSNTIPI	No	No	"IRLM panel 1: DSNTIPI" on page 386
IRLMPRC	DSN6SPRM	DSNTIPI	No	No	"IRLM panel 1: DSNTIPI" on page 386
IRLMRWT	DSN6SPRM	DSNTIPI	No	No	"IRLM panel 1: DSNTIPI" on page 386
IRLMSID	DSN6SPRM	DSNTIPI	No	No	"IRLM panel 1: DSNTIPI" on page 386
IRLMSWT	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
IX_TB_PART_CONV_EXCLUDE	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
IXQTY	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
LBACKOUT	DSN6SYSP	DSNTIPL1	No	No	"Checkpoint parameters: DSNTIPL1" on page 428
LC_CTYPE	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
LEMAX	DSN6SPRM	DSNTIPX	No	No	"Routine parameters panel: DSNTIPX" on page 453
LOB_INLINE_LENGTH	DSN6SYSP	DSNTIPD	Yes	No	"Sizes panel 1: DSNTIPD" on page 313
LOBVALA	DSN6SYSP	DSNTIPD	Yes	No	"Sizes panel 1: DSNTIPD" on page 313
LOBVALS	DSN6SYSP	DSNTIPD	Yes	No	"Sizes panel 1: DSNTIPD" on page 313
LRDRTHLD	DSN6SPRM	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
MAINTYPE	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
MAX_CONCURRENT_PKG_OPS	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
MAX_NUM_CUR	DSN6SPRM	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
MAX_ST_PROC	DSN6SPRM	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453
MAXARCH	DSN6LOGP	DSNTIPA	No	No	"Archive log data set parameters panel: DSNTIPA" on page 434
MAXCONQN	DSN6FAC	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
MAXCONQW	DSN6FAC	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
MAXDBAT	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
MAXKEEPD	DSN6SPRM	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
MAXOFILR	DSN6SYSP	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
MAXRBLK	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
MAXRTU	DSN6LOGP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
MAXTEMPS	DSN6SPRM	DSNTIP9	Yes	No	"Work file database panel: DSNTIP9" on page 323
MAXTEMPS_RID	DSN6SPRM	DSNTIP9	Yes	No	"Work file database panel: DSNTIP9" on page 323
MAXTYPE1	DSN6FAC	DSNTIPR	Yes	No	"Distributed data facility panel 1: DSNTIPR" on page 442
MCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
MEMBNAME	DSN6GRP	DSNTIPK	No	No	"Define group or member panel: DSNTIPK" on page 286
MGEXTSZ	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
MINSTOR	DSN6SPRM	DSNTIPE	Yes	No	"Thread management panel: DSNTIPE" on page 333
MIXED	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
MON	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
MONSIZE	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
MXDTCACH	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
NPGTHRSH	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
NUMLKTS	DSN6SPRM	DSNTIPJ	Yes	No	"IRLM panel 2: DSNTIPJ" on page 391
NUMLKUS	DSN6SPRM	DSNTIPJ	Yes	No	"IRLM panel 2: DSNTIPJ" on page 391
OJPERFEH	DSN6SPRM	—	No	No	"Subsystem parameters that are not on installation panels" on page 253
OPT1ROWBLOCKSORT	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
OPTHINTS	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
OPTIOWGT	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
OTC_LICENSE	DSN6SYSP	DSNTIPO2	No	No	"Notice and acceptance of OTC license panel: DSNTIPO2" on page 275
OUTBUFF	DSN6LOGP	DSNTIPL	No	No	"Active log data set parameters: DSNTIPL" on page 426

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
PADIX	DSN6SPRM	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
PADNTSTR	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
PARA_EFF	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
PARAMDEG	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
PCLOSEN	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
PCLOSET	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
PLANMGMT	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
PLANMGMTSCOPE	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
POOLINAC	DSN6FAC	DSNTIP5	Yes	No	"Distributed data facility panel 2: DSNTIP5" on page 448
PRIQTY	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
PRIVATE_PROTOCOL	DSN6FAC	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
PROTECT	DSN6ARVP	DSNTIPP	Yes	No	"Protection panel: DSNTIPP" on page 410
PTASKROL	DSN6SYSP	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
QUERY_ACCEL_OPTIONS	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
QUERY_ACCELERATION	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
QUIESCE	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
RANDOMATT	DSN6GRP	DSNTIPK	Yes	No	"Define group or member panel: DSNTIPK" on page 286
REALSTORAGE_MANAGEMENT	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
REALSTORAGE_MAX	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
REC_FASTREPLICATION	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
RECALL	DSN6SPRM	DSNTIPO	No	No	"Operator functions panel: DSNTIPO" on page 351
RECALLD	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
REFSHAGE	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
REORG_IGNORE_FREESPACE	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
REORG_LIST_PROCESSING	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
REORG_PART_SORT_NPSI	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
RESTART	DSN6SPRM	DSNTIPS	—	No	"Databases and spaces to start automatically panel: DSNTIPS" on page 441
RESTORE_RECOVER_FROMDUMP	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
RESTORE_TAPEUNITS	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
RESTRICT_ALT_COL_FOR_DCC	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
RESYNC	DSN6FAC	DSNTIPR	Yes	No	"Distributed data facility panel 1: DSNTIPR" on page 442
RETLWAIT	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
RETVLCFK	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
REVOKE_DEP_PRIVILEGES	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412
RGFCOLID	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFDBNAM	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFDEDPL	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFDEFLT	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFESCP	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFFULLQ	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFINSTL	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFNMORT	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RGFNMPRT	DSN6SPRM	DSNTIPZ	No	No	"Data definition control support panel: DSNTIPZ" on page 485
RLF	DSN6SYSP	DSNTIPO	No	No	"Operator functions panel: DSNTIPO" on page 351
RLFAUTH	DSN6SYSP	DSNTIPP	Yes	No	"Protection panel: DSNTIPP" on page 410
RLFERR	DSN6SYSP	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
RLFERRD	DSN6FAC	DSNTIPR	Yes	No	"Distributed data facility panel 1: DSNTIPR" on page 442
RLFTBL	DSN6SYSP	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
ROUTCDE	DSN6SYSP	DSNTIPO	No	No	"Operator functions panel: DSNTIPO" on page 351
RRF	DSN6SPRM	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
RRULOCK	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
SCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
SECADM1	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412
SECADM1_INPUT_STYLE	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SECADM1_TYPE	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412
SECADM2	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412
SECADM2_INPUT_STYLE	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SECADM2_TYPE	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
SECQTY	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
SEPARATE_SECURITY	DSN6SPRM	DSNTIPP1	Yes	Yes	"Protection panel 2: DSNTIPP1" on page 412
SEQCACH	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SEQPRES	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SIGNON_MODULE	DSN6SYSP	DSNTIPO3	No	No	"Default startup modules panel: DSNTIPO3" on page 358
SIMULATED_CPU_COUNT	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SIMULATED_CPU_SPEED	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SITETYP	DSN6SPRM	DSNTIPO	No	No	"Operator functions panel: DSNTIPO" on page 351
SJTABLES	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SKIPUNCI	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
SMF89	DSN6SYSP	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SMFACCT	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
SMFCOMP	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
SMFSTAT	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
SMSDCFL	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SMSDCIX	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
SPT01_INLINE_LENGTH	DSN6SPRM	DSNTIPA2	Yes[2]	No	"DB2 catalog and directory panel: DSNTIPA2" on page 275
SQLDELI	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
SRTPOOL	DSN6SPRM	DSNTIPC	Yes	No	"CLIST calculations panel 1: DSNTIPC" on page 490
SSID	DSNHDECP	DSNTIPM	—	No	"MVS PARMLIB updates panel: DSNTIPM" on page 422
STARJOIN	DSN6SPRM	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
STATCLUS	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253
STATHIST	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
STATIME	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
STATROLL	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
STATSINT	DSN6SPRM	DSNTIPO	Yes	No	"Operator functions panel: DSNTIPO" on page 351
STDSQL	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
STORMXAB	DSN6SYSP	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453
STORTIME	DSN6SYSP	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
SUPERRS	DSN6SPRM	DSNTIPM	Yes	No	"MVS PARMLIB updates panel: DSNTIPM" on page 422
SVOLARC	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
SYNCVAL	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
SYSADM	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
SYSADM2	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
SYSOPR1	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
SYSOPR2	DSN6SPRM	DSNTIPP	Yes	Yes	"Protection panel: DSNTIPP" on page 410
SYSTEM_LEVEL_BACKUPS	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
TBSBP8K	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TBSBP16K	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TBSBP32K	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TBSBPLOB	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TBSBPOOL	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TBSBPXML	DSN6SYSP	DSNTIP1	Yes	No	"Buffer pool sizes panel 1: DSNTIP1" on page 339
TCPALVER	DSN6FAC	DSNTIP5	Yes	Yes	"Distributed data facility panel 2: DSNTIP5" on page 448
TCPKPALV	DSN6FAC	DSNTIP5	Yes	No	"Distributed data facility panel 2: DSNTIP5" on page 448
TIME	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
TIMELEN	DSNHDECP	DSNTIP4	—	No	"Application programming defaults panel 2: DSNTIP4" on page 372
TRACSTR	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
TRACTBL	DSN6SYSP	DSNTIPN	No	No	"Tracing parameters panel: DSNTIPN" on page 343
TRKRSITE	DSN6SPRM	DSNTIPO	No	No	"Operator functions panel: DSNTIPO" on page 351
TSQTY	DSN6SYSP	DSNTIP7	Yes	No	"Sizes panel 2: DSNTIP7" on page 318
TSTAMP	DSN6ARVP	DSNTIPH	Yes	No	"System resource data set names panel: DSNTIPH" on page 290
TWOACTV	DSN6LOGP	DSNTIPH	No	No	"System resource data set names panel: DSNTIPH" on page 290
TWOARCH	DSN6LOGP	DSNTIPH	No	No	"System resource data set names panel: DSNTIPH" on page 290
TWOBSDS	DSN6LOGP	—	No	No	"Add a second BSDS" on page 592
UGCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
UIFCIDS	DSN6SYSP	DSNTIPN	Yes	No	"Tracing parameters panel: DSNTIPN" on page 343
UMCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
UNION_COLNAME_7	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels" on page 253

Table 52. Directory of subsystem parameters and application default values (continued)

Parameter	Macro	Panel	Update online	Security parameter	Topic
UNIT	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
UNIT2	DSN6ARVP	DSNTIPA	Yes	No	"Archive log data set parameters panel: DSNTIPA" on page 434
URCHKTH	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
URLGWTH	DSN6SYSP	DSNTIPL1	Yes	No	"Checkpoint parameters: DSNTIPL1" on page 428
USCCSID	DSNHDECP	DSNTIPF	—	No	"Application programming defaults panel 1: DSNTIPF" on page 362
UTIL_TEMP_STORCLAS	DSN6SPRM	DSNTIP61	Yes	No	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
UTILS_DUMP_CLASS_NAME	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
UTIMOUT	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
UTSORTAL	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
VOLTDEVT	DSN6SPRM	DSNTIP6	Yes	No	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
WFDBSEP	DSN6SPRM	DSNTIP9	No	No	"Work file database panel: DSNTIP9" on page 323
WLMENV	DSN6SYSP	DSNTIPX	Yes	No	"Routine parameters panel: DSNTIPX" on page 453
XLKUPDLT	DSN6SPRM	DSNTIPI	Yes	No	"IRLM panel 1: DSNTIPI" on page 386
XML_RANDOMIZE_DOCID	DSN6SYSP	DSNTIP8	Yes	No	"Performance and optimization panel: DSNTIP8" on page 379
XMLVALA	DSN6SYSP	DSNTIPD	Yes	No	"Sizes panel 1: DSNTIPD" on page 313
XMLVALS	DSN6SYSP	DSNTIPD	Yes	No	"Sizes panel 1: DSNTIPD" on page 313
ZOSMETRICS	DSN6SPRM	—	Yes	No	"Subsystem parameters that are not on installation panels"

Notes:

1. Serviceability parameter.
2. This subsystem parameter can only be changed online. Restarting DB2 has no effect on the setting.

Related concepts:

"The scope and uniqueness of DB2 subsystem parameters" on page 671

Subsystem parameters that are not on installation panels

Some subsystem parameters are defined by job DSNTIJUZ and are not associated with fields on any installation panels.

Subsystem parameters

The following subsystem parameters are not associated with installation panels.

ACCEL in macro DSN6SPRM

Specifies whether accelerator servers can be used with a DB2 subsystem, and how the accelerator servers are to be enabled and started. An accelerator server cannot be started unless it is enabled. This parameter cannot be changed online. You must stop and restart DB2 for a change to ACCEL to take effect.

NO specifies that accelerator servers cannot be used with the DB2 subsystem.

AUTO specifies that accelerator servers are automatically enabled and started when the DB2 subsystem is started.

COMMAND specifies that accelerator servers are automatically enabled when the DB2 subsystem is started. The accelerator servers can be started with the DB2 START ACCEL command.

CACHEDYN_FREELOCAL in macro DSN6SPRM

Indicates whether DB2 can free cached dynamic statements to relieve DBM1 below-the-bar storage. CACHEDYN_FREELOCAL applies only when the KEEPDYNAMIC(YES) bind option is active. If you specify 0, DB2 does not free cached dynamic statements to relieve high use of storage by dynamic SQL caching.

The default value is 1, which means that DB2 frees some cached dynamic statements to relieve high use of storage when the cached SQL statement pools have grown to a certain size.

DDF_COMPATIBILITY

Specifies how the data types of stored procedure output parameters are determined when a non-Java client calls a DB2 for z/OS stored procedure. Valid values are null and SP_PARMS_NJV.

The default value of null means that the data types of the returned output data match the data types of the parameters in the stored procedure definition. This is the default behavior starting in DB2 for z/OS Version 10.

A value of SP_PARMS_NJV means that the data types of the returned output data match the data types of the corresponding CALL statement arguments unless any of the following conditions are true:

- The non-Java client is later than Version 9.*x*.
- The stored procedure uses a parameter data type that was introduced in DB2 for z/OS Version 10 (XML, TIMESTAMP WITH TIMEZONE, or TIMESTAMP with precision greater than 6).

If either condition is true, the data types of the returned output data match the data types of the parameters in the stored procedure definition.

DDLTOX in macro DSN6SPRM

Specifies how long a request for SQL data definition is to wait for a resource before timing out. This parameter is used for the processing of DDL statements and GRANT, REVOKE, and LOCK statements. The actual timeout value is the product of the value of this subsystem parameter and the value of the IRLMRWT subsystem parameter. Valid values are integers between 1 and 254, inclusive.

The default value of 1 means that the value of the IRLMRWT subsystem parameter is used as the timeout value.

DISABLE_EDMRTS in macro DSN6SPRM

Specifies whether to disable the collection of real-time statistics by the environmental description manager (EDM). By default, EDM collects real-time statistics to track when packages were last used. Valid values are NO and YES.

The default value is NO, which means that EDM continues to collect real-time statistics about the last use of packages. This is the recommended value.

A value of YES means that EDM does not collect real-time statistics about the last use of packages. YES might be appropriate in the following situations:

- You have no need to collect information about the last use of packages.

- The SYSIBM.SYSPACKAGE table is persistently unavailable, which prevents the recording of EDM real-time statistics.
- Deferred externalization of these statistics to the SYSIBM.SYSPACKAGE table causes excessive DB2 shutdown time.
- Collection of these EDM statistics causes performance degradation.

In a data sharing environment, all members should use the same setting for this parameter.

DISABSCL in macro DSN6SPRM

Specifies whether SQLWARN1 and SQLWARN5 are set for non-scrollable cursors on OPEN and ALLOCATE CURSOR.

The default value is NO, which means that SQLWARN1 and SQLWARN5 are set.

This parameter is deprecated in DB2 Version 10.

DISALLOW_DEFAULT_COLLID in macro DSN6SPRM

Specifies whether the default collection ID, DSN_DEFAULT_COLLID_*plan-name*, is used for implicitly generated packages during the automatic DBRM to package conversion process. Valid values are YES and NO.

A value of YES has the following implications:

- The BIND PLAN command cannot be used with the MEMBER option. When this setting is used, any attempt to use the BIND PLAN command with the MEMBER option fails.
- The REBIND PLAN command cannot be used without the COLLID option or COLLID (*) specified if the plan has directly bound DBRMs. When this setting is in effect, use the COLLID option to specify a collection ID for the directly bound DBRMs.
- Automatic rebinds do not convert existing plans that were last bound from DBRMs; such attempts fail with a negative SQLCODE.

A value of NO has the following implications:

- The BIND PLAN command is not restricted from honoring the use of the MEMBER option. When this setting is used and a BIND PLAN statement contains the MEMBER option, DB2 first binds the specified DBRM as a package in a collection ID that is named DSN_DEFAULT_COLLID_*plan-name*. DB2 then binds the plan (*plan-name*) from that collection ID.
- The REBIND PLAN command can be used with or without the COLLID option or COLLID(*) specified. When this setting is used and the plan has directly bound DBRMs, DB2 first binds the DBRM as a package in a collection ID that is named DSN_DEFAULT_COLLID_*plan-name*. DB2 then rebinds the plan (*plan-name*) from that collection ID.
- Automatic rebinds can convert existing plans that were bound from DBRMs into packages from which those plans are then rebound. When an automatic rebind of a plan occurs and that plan was bound directly from a DBRM, DB2 first converts the plan into a package in a collection ID that is named DSN_DEFAULT_COLLID_*plan-name*. DB2 then rebinds the plan (*plan-name*) from that collection ID.

The default value of DISALLOW_DEFAULT_COLLID is NO.

EN_PJSJ in macro DSN6SPRM

Specifies whether to enable dynamic index ANDing, which is also called

pair-wise join processing, when star join processing is enabled on DB2. Valid values are ON and OFF. The default value is OFF, which means that dynamic index ANDing is disabled.

When dynamic index ANDing is enabled, it can improve the execution performance of qualified queries in the following cases:

- The fact table has indexes that provide matching for join predicates with some dimension tables having local filtering. The indexes on the fact table can be single-key indexes (but not limited).
- No multi-column index on the fact table is suitable for the existing star join method.

Therefore, setting EN_PJSJ to ON can provide more flexibility with index design on star schema fact tables when multi-column index design makes it difficult to effectively exploit the star join access method.

HONOR_KEEPDICTIONARY in macro DSN6SPRM
Specifies whether DB2 honors the LOAD and REORG parameter KEEPDICTIONARY when tables are converted from basic row format to reordered row format. If HONOR_KEEPDICTIONARY is set to a value of YES, DB2 honors the LOAD and REORG parameter KEEPDICTIONARY

If HONOR_KEEPDICTIONARY is set to the default value of NO, DB2 ignores the LOAD and REORG parameter KEEPDICTIONARY when tables are converted from basic row format to reordered row format.

IMPDSSIZE in macro DSN6SYSP
Specifies the maximum data set size (DSSIZE) in gigabytes that DB2 should use for creating an implicit base table space. This parameter is not used for the creation of implicit LOB or XML table spaces. Valid settings are 1, 2, 4, 8, 16, 32, 64, 128, and 256.

The default value of IMPDSSIZE is 4, unless you use SAP.

If you specify an SAP-specific input member (for example, DSNTIDXB) during installation or migration, the DSNTIJUZ job that is generated by the DB2 CLIST specifies IMPDSSIZE=64.

IMPTKMOD in macro DSN6SYSP
Specifies whether DB2 is to track modifications to the pages of implicitly created table spaces. This parameter only pertains to base table spaces. The setting is not used for implicitly created LOB or XML table spaces. Valid values are YES and NO.

The default value, YES, means that DB2 tracks changed pages of implicitly created table spaces in the space map pages. This tracking is intended to improve the performance of incremental image copies.

A value of NO means that DB2 does not track changed pages in the space map pages. DB2 uses the LRSN value in each page to determine whether a page has been changed.

If you specify an SAP-specific input member (for example, DSNTIDXB) during installation or migration, the DSNTIJUZ job that is generated by the DB2 CLIST specifies IMPTKMOD=NO.

INDEX_IO_PARALLELISM in macro DSN6SPRM
Specifies whether I/O parallelism is enabled for index insertion. I/O parallelism allows concurrent insert operations on multiple indexes and can reduce I/O wait time when many indexes are defined in a table. Valid values are YES and NO.

The default value is YES, which means that I/O parallelism is enabled.

INLISTP in macro DSN6SPRM
Allows you to specify the maximum number of elements in an IN-list for certain IN predicate optimizations to occur.

The default value for INLISTP is 50.

IX_TB_PART_CONV_EXCLUDE in macro DSN6SPRM
Specifies whether to exclude trailing columns from the table-controlled partitioning keys when table spaces are converted from index-controlled partitioning to table-controlled partitioning. Valid values are NO and YES.

The default, NO, means that when converting from index-controlled partitioning to table-controlled partitioning, the DB2 subsystem uses all columns of the original partitioning key to define the new partitioning key. This definition includes the trailing columns of the original partitioning key, which do not affect the partitioning.

A value of YES means that when converting from index-controlled partitioning to table-controlled partitioning, the DB2 subsystem excludes trailing columns from the original partitioning key in the definition of the new partitioning key. The trailing columns are the columns that do not affect the partitioning.

This parameter does not affect the automatic conversion to table-controlled partitioning when a CREATE INDEX statement is issued with a PART VALUES clause and without a CLUSTER clause. In this case, the DB2 subsystem uses the partitioning key that is specified by the CREATE INDEX statement.

MAX_CONCURRENT_PKG_OPS in macro DSN6SPRM
Specifies the maximum number of automatic bind requests, remote bind requests, remote rebind requests, CREATE TRIGGER statements, and CREATE PROCEDURE statements for packages that can be processed simultaneously. If the value of the AUTO BIND field of panel DSNTIPO is set to NO, the value of MAX_CONCURRENT_PKG_OPS has no effect.

The default value of MAX_CONCURRENT_PKG_OPS is 10.

MAXCONQN in macro DSNDSN6FAC
Specifies the maximum number of inactive or new connections that can be queued waiting for a DBAT to process the request.

OFF means that the depth of the connection queue is limited by the value of the CONDBAT subsystem parameter. ON means that the depth of the connection queue is limited by the value of the MAXDBAT subsystem parameter. A numeric value specifies the maximum number of connections that can be queued waiting for a DBAT to process a request.

When a request is added to the connection request queue and the thresholds specified by both the MAXDBAT and MAXCONQN subsystem parameters are both reached (unless MAXCONQN is set to OFF) then DDF closes the client connection longest waiting client connection in the queue. The closed connections give remote clients an opportunity to redirect the work to other members of the group that have more resources to process the work. The function is enabled only when DB2 subsystem is a member of a data sharing group.

The default value is OFF.

MAXCONQW in macro DSNDSN6FAC
Specifies the maximum length of time that a client connection waits for a DBAT to process the next unit-of-work or new connection request.

ON means that connections wait as long as the value specified by the IDHTOIN subsystem parameter. OFF means that connections wait indefinitely for a DBAT to process requests. A numeric value specifies a time duration in seconds that a connection waits for a DBAT to process the request.

Each queued connection request entry is examined to see if its time waiting in the queue has exceeded the specified value. If the time is exceeded, the client connection is closed. After all entries in the queue have been processed or the last entry whose time in the queue exceeded the threshold has been processed, a DSNL049I message is issued to indicating how many client connections were closed because of the MAXCONQW value. The function is enabled only when DB2 subsystem is a member of a data sharing group.

The default value is OFF.

NPGTHRSH in macro DSN6SPRM
Specifies whether DB2 is to use special access path selection for tables under a given size.

Unless you use SAP, the default value is 0, which means that no special access path selection is used.

If you specify an SAP-specific input member (for example, DSNTIDXB) during installation or migration, the DSNTIJUZ job that is generated by the DB2 CLIST specifies NPGTHRSH=10. This value means that DB2 favors index access only for tables with values smaller than 10 in the NPAGESF column of the SYSIBM.SYSTABSTATS catalog table.

For more information about using this subsystem parameter, see Favoring index access (DB2 Performance).

OJPERFEH in macro DSN6SPRM
Specifies whether to disable performance enhancements for outer join operations.

This parameter is deprecated in DB2 Version 10.

OPT1ROWBLOCKSORT in macro DSN6SPRM
Specifies whether DB2 explicitly blocks sort operations when the OPTIMIZE FOR 1 ROW clause is specified on a query. Valid values are ENABLE or DISABLE.

ENABLE means that when OPTIMIZE FOR 1 ROW is specified, DB2 chooses access paths that avoid sorts whenever such a path is available. For example, DB2 must choose between two index access paths. One index is very selective and filters two rows, but it requires a sort. The other index does not filter any rows, but it does not require a sort. If the OPT1ROWBLOCKSORT subsystem parameter is set to ENABLE, DB2 chooses the path that does not require a sort.

DISABLE means that when OPTIMIZE FOR 1 ROW is specified, DB2 avoids access paths that involve sorts. If an access path that avoids a sort exists, it is possible, although unlikely, that an access path that involves a sort is chosen instead. This behavior is used in Version 9 and earlier releases.

The default value of OPT1ROWBLOCKSORT is DISABLE.

OPTIOWGT in macro DSN6SPRM
Controls how DB2 balances the I/O cost and CPU estimates when selecting access paths. Valid values are ENABLE or DISABLE. ENABLE specifies that when selecting access paths, DB2 is to use a new formula that better balances the cost estimates of I/O response time and CPU usage. DISABLE specifies that DB2 is to use the original formula to balance the CPU and I/O estimates.

ENABLE is the default value.

Note: Changing the OPTIOWGT parameter could impact the access paths of queries that are otherwise stable. For more advice on how to set this parameter, contact IBM Software Support.

This parameter is deprecated in DB2 Version 10.

PRIVATE_PROTOCOL in macro DSN6FAC

Private protocol is no longer supported; however, this subsystem parameter controls package EXECUTE authorization behavior related to remote DB2 requester applications. When a remote DB2 requester application accesses the server DB2 subsystem through DRDA protocol, a plan owner is associated with the remote DB2 requester application plan. For packages that are executed as part of this plan by a remote DB2 requester application using DRDA protocol, the PRIVATE_PROTOCOL parameter has the following effects. This parameter is online updatable, and valid values are NO and AUTH.

NO indicates that when a remote DB2 requester application accesses the DB2 subsystem through DRDA protocol, package EXECUTE authority must be explicitly granted at this subsystem. The authority must be granted to the primary ID or an associated secondary ID for packages that the requester application executes as part of the plan

AUTH indicates that when a remote DB2 requester application accesses the DB2 subsystem through DRDA protocol, package EXECUTE authority must be explicitly granted to the plan owner at this subsystem. This plan owner must have EXECUTE authority at this subsystem for packages that the requester application executes as part of the plan. The plan owner needs no other privilege to execute the package.

The default value for PRIVATE_PROTOCOL is NO.

PTASKROL in macro DSN6SYSP

Indicates whether to roll up accounting trace records from a parallel query task into the originating task's accounting trace. A value of YES means the originating task is to generate an additional accounting trace record with all the roll-up values from parallel tasks.

Recommendation: Use the default value of YES. A value of NO means that each parallel task produces its own accounting trace record.

QUERY_ACCEL_OPTIONS in macro DSN6SPRM

Specifies additional types of SQL queries that are eligible for acceleration.

NONE specifies that no additional types of SQL queries are eligible. Therefore, the types of queries that are described in the other available values for this parameter are not eligible for acceleration. This is the default value.

1 specifies that queries that include data that is encoded with the EBCDIC mixed or graphic encoding schemes are eligible for acceleration.

2 specifies that an INSERT with SELECT statement is eligible for acceleration. However, only the select operation of the query is processed by the accelerator server.

QUERY_ACCELERATION in macro DSN6SPRM

Determines the default value that is to be used for the CURRENT QUERY ACCELERATION special register.

ENABLE specifies that queries are accelerated only if DB2 determines that it is advantageous to do so. If there is an accelerator failure while a query is running, or the accelerator returns an error, DB2 returns a negative SQLCODE to the application.

ENABLE_WITH_FAILBACK specifies that queries are accelerated only if DB2 determines that it is advantageous to do so. If the accelerator returns an error during the PREPARE or first OPEN for the query, DB2 executes the query without the accelerator. If the accelerator returns an error during a FETCH or a subsequent OPEN, DB2 returns the error to the user, and does not execute the query.

NONE specifies that no query acceleration is done.

The default value for QUERY_ACCELERATION is NONE.

REALSTORAGE_MANAGEMENT in macro DSN6SPRM

Specifies whether DB2 should manage real storage consumption. Valid values are ON, OFF, and AUTO.

A value of ON means that DB2 always discards unused real storage frames. Discarding the frames results in some CPU overhead, and this option is intended for systems in which the availability of real storage is limited. This value would most likely be appropriate for LPARs that have many DB2 subsystems, such as a development LPAR.

A value of OFF means that DB2 does not discard unused real storage frames until one of the following conditions is met:

- The LPAR had reached an auxiliary critical state.
- The total real and auxiliary storage has reached 80% of the value of the REALSTORAGE_MAX subsystem parameter.

A value of AUTO means that DB2 discards unused real storage frames when a significant amount of paging activity is detected. By discarding frames, DB2 tries to bring the system to a point where paging is limited or nonexistent. However, it might not be possible to bring the system to that point if other applications on the same LPAR cause the shortage of real storage frames.

The default setting is AUTO.

REALSTORAGE_MAX in macro DSN6SPRM

Specifies the maximum GB of real and auxiliary storage that DB2 can consume. Valid values are NOLIMIT and 1 to 65535.

The default value, NOLIMIT, means that DB2's real and auxiliary storage consumption is not bounded.

If a numerical value is specified for this parameter and the total real and auxiliary storage exceeds that limit, DB2 is terminated.

The recommendation is to set this parameter to twice the amount of real and auxiliary storage that the subsystem might reasonably consume.

REORG_IGNORE_FREESPACE in macro DSN6SPRM

Controls whether DB2 uses the PCTFREE and FREEPAGE values that are defined for a partition-by-growth table space in the following situations:

- When REORG TABLESPACE is run against a subset of the partitions in a partition-by-growth table space
- When REORG TABLESPACE is run against a partition-by-growth table space in which a table contains LOB columns

Valid values are YES and NO.

If REORG_IGNORE_FREESPACE is set to YES, DB2 uses 0 for the PCTFREE and FREEPAGE values when it reloads data into the table space.

The default value of REORG_IGNORE_FREESPACE is NO, which means that the REORG TABLESPACE utility uses the PCTFREE and FREEPAGE values.

REORG_LIST_PROCESSING in macro DSN6SPRM
Specifies the default setting for the PARALLEL option of the DB2 REORG TABLESPACE utility. Valid values for the REORG_LIST_PROCESSING subsystem parameter are PARALLEL and SERIAL.

The default value PARALLEL specifies that the REORG TABLESPACE utility is to use a default PARALLEL YES option when the PARALLEL keyword is not specified in the utility control statement. The PARALLEL YES option specifies that the REORG TABLESPACE utility is to process all partitions that are specified in the input LISTDEF statement in a single execution of the utility.

SERIAL specifies that the REORG TABLESPACE utility is to use a default PARALLEL NO option when the PARALLEL keyword is not specified in the utility control statement. The PARALLEL NO option specifies that each partition that is specified in the input LISTDEF statement is to be processed in a separate execution of the utility.

REORG_PART_SORT_NPSI in macro DSN6SPRM
Specifies whether the REORG TABLESPACE PART utility decides to sort all of the keys of a non-partitioned secondary index. The setting is ignored for a REORG that is not part-level or without non-partitioned secondary indexes.

The default value of REORG PART SORT NPSI is NO, which means that only keys of the non-partitioned secondary indexes that are in the scope of the REORG are sorted.

RESTRICT_ALT_COL_FOR_DCC in macro DSN6SPRM
Specifies whether restrictions apply to ALTER TABLE ALTER COLUMN statements for target tables that are defined with the DATA CAPTURE CHANGES attribute. Valid values are NO and YES.

The default value, NO, means that restrictions do not apply to altering column data types or default values of tables that are defined with the DATA CAPTURE CHANGES attribute. For such tables, ALTER TABLE ALTER COLUMN statements with the SET DATA TYPE, SET DEFAULT, or DROP DEFAULT options are allowed.

A value of YES means that restrictions apply to altering column data types or default values of tables that are defined with the DATA CAPTURE CHANGES attribute. ALTER TABLE ALTER COLUMN statements that are issued with the SET DATA TYPE, SET DEFAULT, or DROP DEFAULT options for such tables result in SQLCODE -148.

RETVLCFK in macro DSN6SPRM
Specifies whether the VARCHAR column is to be retrieved from a padded index. Valid values are YES or NO. The data sharing scope of this parameter is GROUP.

If you choose NO, DB2 might retrieve the non-padded value if it needs to return the non-padded value or needs the non-padded value for predicate evaluation.

If you choose YES, better performance might result because of index-only access. However, the data that is retrieved from the index is padded with blanks to the maximum length of the column. This might result in an incorrect output because predicates are applied to the padded value rather then the

non-padded value, which can cause different results in some cases. Also, use of the ENCRYPT function can result in an error because of the similar issues with functions being applied to the padded value rather than the non-padded value.

Important: Applications must be able to handle the padding blanks. If your application is sensitive to these blanks, keep the default value of NO or consider using non-padded indexes. You must rebind plans and packages to enable the change.

Recommendation: Accept the default value of NO. Do not set RETVLCFK to YES unless you have tested all of your applications to make sure that none of them have any predicates that are affected by trailing blanks, none of them use the ENCRYPT function or any other function, such as the LENGTH function, that are affected by trailing blanks, and that the application logic is not affected by trailing blanks.

Use NOT PADDED indexes if you want the index to use less space, you want to allow index-only retrieval with variable characters, and you do not want the incompatible retrieval of the full column width or incorrect results.

This parameter is deprecated in DB2 Version 10.

SECADM1_INPUT_STYLE in macro DSN6SPRM

Specifies whether the SECADM1 setting is passed as a hexadecimal string or as a standard character string. Valid values are HEX or CHAR. A value of HEX means that SECADM1 is a hexadecimal character string that represents a Unicode-encoded role name. A value of HEX is valid only when SECADM1_TYPE=ROLE. When SECADM1_INPUT_STYLE is set to HEX, SECADM1 must be an even number of bytes, 2 - 256, consisting entirely of 0 - 9 and A - F.

The default value for SECADM1_INPUT_STYLE is CHAR, which means that SECADM1 is passed as a standard character string. If SECADM1_TYPE=AUTHID, SECADM1 can be an authorization ID of 1 - 8 characters, or if SECADM1_TYPE=ROLE, SECADM1 can be an ordinary SQL identifier of 1 - 128 characters.

SECADM2_INPUT_STYLE in macro DSN6SPRM

Specifies whether the SECADM2 setting is passed as a hexadecimal string or as a standard character string. Valid values are HEX or CHAR. A value of HEX means that SECADM2 is a hexadecimal character string that represents a Unicode-encoded role name. A value of HEX is valid only when SECADM2_TYPE=ROLE. When SECADM2_INPUT_STYLE is set to HEX, SECADM2 must be an even number of bytes, 2 - 256, consisting entirely of 0 - 9 and A - F.

The default value for SECADM2_INPUT_STYLE is CHAR, which means that SECADM2 is passed as a standard character string. If SECADM2_TYPE=AUTHID, SECADM2 can be an authorization ID of 1 - 8 characters, or if SECADM2_TYPE=ROLE, SECADM2 can be an ordinary SQL identifier of 1 - 128 characters.

SEQCACH in macro DSN6SPRM

Specifies whether the sequential mode is to be used to read cached data from a 3990 controller. Valid values are BYPASS and SEQ.

If you specify BYPASS, DB2 prefetch bypasses the cache.

The default value, SEQ, means that DB2 prefetch uses sequential access for read activity. Many sites gain a performance benefit by specifying SEQ.

Recommendation: Specify SEQ if you have current disk devices with good cache sizes, especially if the units are Enterprise Storage System (ESS) or RAMAC Virtual Array (RVA).

This parameter is deprecated in DB2 Version 10.

SEQPRES in macro DSN6SPRM

Some DB2 utilities scan nonpartitioning indexes and then update a subset of the pages in the indexes. The SEQPRES subsystem parameter specifies whether, when reading data, these utilities are to allow data to remain in cache longer. Valid values are YES and NO.

The default value, YES, means that these DB2 utility prefetch reads remain in cache longer, possibly improving performance of subsequent writes in the following cases for a table with very large nonpartitioned indexes:

- LOAD PART integer RESUME
- REORG TABLESPACE PART

The utility cache option is useful only with 3390 geometry disk attached to the 3990 Model 6 or later controllers.

A value of NO means that DB2 utilities use the 3990 cache the same way as any other application, which is specified by the SEQCACH subsystem parameter.

This parameter is deprecated in DB2 Version 10.

SIMULATED_CPU_COUNT in macro DSN6SPRM

Specifies the number of CPUs that are being simulated. This parameter is to be used for DB2 optimization modeling only.

Valid values are OFF and numbers 1 to 255. The parameter is online changeable.

The default value, OFF, means that DB2 optimizes for the number of CPUs on which it is running. This setting is recommended for production environments.

SIMULATED_CPU_SPEED in macro DSN6SPRM

Specifies the microseconds of execution time for tasks or service request blocks (SRBs) per service unit for the CPU that is being simulated. This parameter is to be used for DB2 optimization modeling only.

Valid values are OFF and numbers 1 to 2,147,483,647. The parameter is online changeable.

The default value, OFF, means that DB2 optimizes for the CPU on which it is running. This setting is recommended for production environments.

SJTABLES in macro DSN6SPRM

Specifies when DB2 is to enable star join processing. For more information on setting this parameter, see Star schema access (DB2 Performance).

SMF89 in macro DSN6SYSP

Specifies whether DB2 is to do detailed tracking for measured usage pricing. If you select YES, DB2 does detailed measured usage tracking if SMF type 89 records are activated. When SMF89 is set to YES, DB2 invokes a z/OS service on every entry into or exit out of DB2 to ensure accurate tracking.

The default value is NO, which means that DB2 does not do detailed measured usage tracking. If the SMF type 89 record is activated, only high-level tracking is recorded in the SMF type 89 record. Selecting NO reduces CPU usage, but also increases the amount of time spent in DB2 as measured by SMF 89.

Recommendation: Select SMF89 YES only if you use measured usage pricing.

SMSDCFL in macro DSN6SPRM

Specifies a DFSMS data class for table spaces. Before you set this parameter, you need to define the data classes for your table space data sets. You also need to code SMS automatic class selection (ACS) routines to assign indexes to one SMS storage class and table spaces to a different SMS storage class.

SMSDCFL is one to eight characters long. When you use DFSMS and DB2 storage groups, you can use SMSDCFL to assign table spaces to different DFSMS data classes. If you assign a value to SMSDCFL, DB2 specifies that value when it uses Access Method Services to define a data set for a table space. If the value of SMSDCFL is one or more blanks, DB2 does not specify a data class when it creates data sets for table spaces.

The default value of SMSDCFL is a blank.

This parameter is deprecated in DB2 Version 10.

SMSDCIX in macro DSN6SPRM

Specifies a DFSMS data class for indexes. Before you set this parameter, you need to define the data classes for your index data sets. You also need to code SMS automatic class selection (ACS) routines to assign indexes to one SMS storage class and table spaces to a different SMS storage class.

SMSDCIX is one to eight characters long. When you use DFSMS and DB2 storage groups, you can use the subsystem parameters SMSDCFL and SMSDCIX to assign table spaces and indexes to different DFSMS data classes. If you assign a value to SMSDCIX, DB2 specifies that value when it uses Access Method Services to define a data set for an index. If the value for SMSDCIX is one or more blanks, DB2 does not specify a data class when it creates data sets for indexes.

The default value of SMSDCIX is a blank.

This parameter is deprecated in DB2 Version 10.

STATCLUS in macro DSN6SPRM

Specifies the type of clustering statistics that are to be collected by the RUNSTATS utility. Valid values are ENHANCED and STANDARD.

The default value is ENHANCED. The ENHANCED clustering statistics result in an improved CLUSTERRATIO formula and the statistic DATAREPEATFACTOR. These statistics allow improved SQL access paths based on improved recognition of the number of getpages that are required to retrieve the qualified rows and whether those getpages benefit from prefetch.

Recommendation: Use the default value of ENHANCED unless directed otherwise by IBM.

This parameter is deprecated in DB2 Version 10.

UNION_COLNAME_7 in macro DSN6SPRM

Specifies what behavior DB2 uses for result column names in UNION queries. The default value of NO results in Version 8 behavior. If the column name is the same across all sub-queries in the UNION query, the result column name will be that column name. Otherwise, the result column is unnamed.

When the value of UNION_COLNAME_7 is YES, DB2 uses Version 7 and earlier behavior. The column name that is returned in the SQLNAME field of

an SQLDA following a DESCRIBE statement, where the result table that is being described is the result of a union, is the column name from the first subquery of the union operation.

ZOSMETRICS in macro DSN6SPRM

Enables DB2 to gather z/OS metrics, including CPU usage and storage metrics, by using the z/OS Resource Measurement Facility™ (RMF™) interface. Valid values are YES and NO.

The default value of ZOSMETRICS is NO.

The following actions are required to make the RMF interface operational:

- Verify that the DSNWVRMF module is placed in the LINKLIST library. After installation, refresh the LINKLIST by issuing the following MVS command:

  ```
  f lla,refresh
  ```

- Start the RMF and RMF Monitor III Sysplex Data Retrieval Service.

In some cases, the z/OS metrics might not be provided, or they might be only partially provided in the DB2 trace. This lack of metrics is likely caused by setup problems that are related to the RMF. If support is required, provide the incomplete trace data.

Related concepts:

Star schema access (DB2 Performance)

Related tasks:

"Installation step 5: Define DB2 initialization parameters: DSNTIJUZ" on page 516

Related reference:

Defining data classes (DFSMS Implementing System-Managed Storage)

Subsystem parameter module and application defaults module values

The subsystem parameter module is generated by job DSNTIJUZ each time that you install, migrate, or update DB2. By default, this load module has the name that you specify in the PARAMETER MODULE field of installation panel DSNTIPO3.

Six macros expand to form this data-only subsystem parameter load module. The load module contains the DB2 execution-time parameters that you selected using the ISPF panels. These six macros are DSN6ARVP, DSN6FAC, DSN6LOGP, DSN6SPRM, DSN6SYSP, and DSN6GRP.

The application defaults load module, traditionally referred to as DSNHDECP, is also generated by job DSNTIJUZ. This module contains the application programming defaults. By default, this load module has the name that you specify in the APPL DEFAULTS field of installation panel DSNTIPO3.

Data sharing

If you plan to use data sharing, see "Subsystem parameters and application defaults in a data sharing environment" on page 670.

Main panel: DSNTIPA1

The entries on the Main panel control input to and output from the installation CLIST.

When processing is complete, this panel is displayed again. The values that you enter are saved in the ISPF profile for your authorization ID and are displayed each time that you run the CLIST.

To save your panel input, you must specify an output member name in the OUTPUT MEMBER NAME field.

The DSNTINST CLIST saves the panel input into your DSNTID*xx* output member just before the CLIST issues this message:

```
DSNT4781 BEGINNING EDITED DATA SET OUTPUT
```

```
DSNTIPA1 DB2 VERSION 10 INSTALL, UPDATE, MIGRATE, AND ENFM - MAIN PANEL
===> _

Check parameters and reenter to change:
 1  INSTALL TYPE          ===> INSTALL   Install, Migrate, ENFM, or Update
 2  DATA SHARING          ===> NO        Yes or No (blank for ENFM or Update)

For migration only: Enter the release you are migrating from and the data set
and member name with the settings from the previous Installation/Migration:
 3  FROM RELEASE          ===>           V8 or V9
 4  DATA SET(MEMBER) NAME ===>

For DB2 SMP/E libraries (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST, etc.), enter:
 5  LIBRARY NAME PREFIX   ===> DSN1010
 6  LIBRARY NAME SUFFIX   ===>

For install data sets (NEW.SDSNSAMP, NEW.SDSNCLST, RUNLIB.LOAD, etc.), enter:
 7  DATA SET NAME PREFIX  ===> DSN1010
 8  DATA SET NAME SUFFIX  ===>

Enter to set or save panel values (by reading or writing the named members):
 9  INPUT MEMBER NAME     ===> DSNTIDXA  Default parameter values
 10 OUTPUT MEMBER NAME    ===>           Save new values entered on panels
PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 12. Main panel: DSNTIPA1

Recommended approach for a new installer

If you are installing for the first time, try the following suggestions.

- In the INSTALL TYPE field, enter INSTALL.
- Set the PREFIX and SUFFIX fields to the values that you used when you allocated the DB2 libraries by using job DSNALLOC.
- In the INPUT MEMBER NAME field, use DSNTIDXA (the default) for the first run. If you are using SAP, specify DSNTIDXB, the SAP-specific CLIST input member, instead. For any later runs, the CLIST sets the default input name to the prior output name.
- Specify a value in the OUTPUT MEMBER NAME field to save your options. Specify values in the TEMP CLIST LIBRARY, CLIST LIBRARY, and SAMPLE LIBRARY fields on installation panel DSNTIPT when you want output data sets tailored.

Do not run the installation jobs before tailoring them with the CLIST. If you want to update the parameters later, the CLIST sets the default input name to the prior output name.

INSTALL TYPE field

The value of the INSTALL TYPE field specifies whether you are installing DB2, updating DB2, migrating to Version 10 conversion mode (from Version 8 or Version 9.1), or migrating to new-function mode.

Acceptable values:	INSTALL, UPDATE, MIGRATE, ENFM
Default:	INSTALL
DSNHDECP:	none

INSTALL
Specifies that you are installing DB2 for the first time. This is the default value for the first run of the CLIST. After you complete the installation, DB2 is in new-function mode.

UPDATE
Specifies that you are updating parameters for an existing DB2 subsystem.

MIGRATE
Specifies that you are migrating from Version 8 to Version 10 or from Version 9.1 to Version 10.

When you are migrating, the DATA SET(MEMBER) NAME field is required.

ENFM Specifies that you are migrating the DB2 catalog to new-function mode. You must run the CLIST in MIGRATE mode before you can choose this option. You should be stabilized in Version 10 conversion mode (from Version 8 or Version 9.1) before using this option to move to new-function mode. In a data sharing environment, the process to migrate to new-function mode is done once for the data sharing group.

If you are updating or migrating, you use the same set of panels that you use for installation. Each panel displays all fields; however, the fields that cannot be changed in update or migrate mode are protected. This way, you can see the values that are related to ones that you want to change.

You can also choose either INSTALL or UPDATE to recheck values you that chose before.

Certain fields cannot be changed during a migration. See panels DSNTIPH, DSNTIPD, DSNTIP7, and DSNTIPP for more information. Ensure that those fields are correct in the data set member that you provide.

Related concepts:

"The update process" on page 501

Related tasks:

"Installing DB2" on page 507

"Migrating your DB2 subsystem to conversion mode (from Version 8 or Version 9.1)" on page 570

"Migrating your DB2 subsystem to new-function mode" on page 649

Related reference:

"System resource data set names panel: DSNTIPH" on page 290

"Sizes panel 1: DSNTIPD" on page 313

"Sizes panel 2: DSNTIP7" on page 318

"Protection panel: DSNTIPP" on page 410

DATA SHARING field (DSHARE subsystem parameter)

The DSHARE subsystem parameter controls whether the data sharing function is to be used.

Acceptable values:	YES, NO, or blank for update and ENFM
Default:	NO
DSNZP*xxx*:	DSN6GRP DSHARE

Choose NO if you are not using data sharing. If you choose YES, you will continue to panel DSNTIPK after completing panel DSNTIPA2.

If you specify YES during **installation,** this window is displayed:

```
.-------------------------------.
| DSNTIP0A                      |
|                               |
|  DATA SHARING FUNCTION:       |
|                               |
|  Select one.                  |
|          _ 1. Group           |
|            2. Member          |
|            3. Enable          |
|                               |
|  PRESS:  ENTER to continue    |
|          RETURN to exit       |
|                               |
|                               |
|                               |
'-------------------------------'
```

Figure 13. DSNTIP0A

DATA SHARING FUNCTION:

Specify a data sharing function.

Acceptable values:	Group, Member, Enable
Default:	none
DSNZP*xxx*:	none

A value is required. After entering a value, you proceed to panels DSNTIPA2 and DSNTIPK.

If you specify YES in the DATA SHARING field during **migration**, a window is displayed asking if the current member is the first to migrate.

```
.-------------------------------------.
| DSNTIP0B                            |
|                                     |
|  FIRST MEMBER OF GROUP TO MIGRATE?  |
|                                     |
|  Select one.                        |
|          _ 1. Yes                   |
|            2. No                    |
|                                     |
|  PRESS:  ENTER to continue          |
|          RETURN to exit             |
|                                     |
|                                     |
'-------------------------------------'
```

Figure 14. DSNTIP0B

FIRST MEMBER OF GROUP TO MIGRATE?

Specify Yes if this is the first member of a data sharing group to migrate.

Acceptable values:	Yes, No
Default:	none
DSNZP*xxx*:	none

A value is required. After entering a value, you proceed to panels DSNTIPA2 and DSNTIPK.

Related tasks:

"Adding a new member" on page 691

"Enabling DB2 data sharing" on page 679

"Installing a new data sharing group" on page 678

FROM RELEASE field

If you are migrating DB2, the FROM RELEASE field specifies whether you are migrating from Version 8 or Version 9.1.

Acceptable values:	V8, V9
Default:	none
DSNZP*xxx*:	none

V8 Indicates that you are migrating from Version 8 to Version 10.

V9 Indicates that you are migrating from Version 9.1 to Version 10.

This field must be blank if the INSTALL TYPE field is not set to MIGRATE.

DATA SET(MEMBER) NAME field

The value of the DATA SET(MEMBER) NAME field indicates the name of the input data set to use for migrating from your DB2 Version 8 or Version 9.1 release. If you are installing, migrating to new-function mode, or updating, the field must remain blank.

Acceptable values:	1 to 44 alphanumeric characters
Default:	NULL

DSNZP*xxx*: none

The named member contains the output parameters that are produced when you last installed, updated, or migrated DB2. Give the fully qualified data set name in the following form:

```
any.data.set.name(member)
```

This is an example of an actual data set name:

```
DSN910.SDSNSAMP(DSNTID81)
```

If you no longer have this data set member, or if the one you have is incorrect, use the installation or update process from your previous release to re-create or correct the member. Enter the correct values on the panels, and save them under a new output member name. Discard the JCL that is created by this process; use the newly created member for migration.

Alternatively, you can run job DSNTIJXZ on your subsystem to create an updated data set member that contains the subsystem parameter and buffer pool settings from your previous release.

Valid data set name

Valid data set names can be unqualified or qualified:

Unqualified name
One to eight alphanumeric or national characters, a hyphen, or the character X'C0'. The first character must be alphabetic or national. Do not use hyphens in data set names for RACF-protected data sets. For example, ALPHA is an unqualified data set name.

Qualified name
Multiple names joined by periods. Each name is coded like an unqualified name. Therefore, the name must contain a period within every eight characters. For example, ALPHA.PGM is a qualified data set name. The maximum length of a qualified data set name is:
- 44 characters if you use the TSO PROFILE setting NOPREFIX.
- 42 characters if you use the TSO PROFILE setting PREFIX.
- For an output tape data set, 17 characters, including periods. If the name is longer than 17 characters, only the right-most 17 characters are written to the tape header label.

Related tasks:

"Updating the CLIST defaults input member: job DSNTIJXZ" on page 220

"Migrating an existing data sharing group to a new release" on page 711

LIBRARY NAME PREFIX field

The LIBRARY NAME PREFIX field specifies the input prefix for the SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNDBRM, and SDSNCLST libraries.

Acceptable values: 1 to 18 characters
Default: DSN1010
DSNZP*xxx*: none

The prefix must be the same as the name that you specified for the symbolic parameter TARGPRE in SMP/E job DSNALLOC.

This is the prefix of the DB2 SMP/E target libraries. If you specify DSNTIDXA or DSNTIDXB in the INPUT MEMBER NAME field, the prefix for the INCLUDE LIBRARY field, and the prefix and suffix for the LINK LIST LIBRARY field through the IVP DATA LIBRARY field on "Data set names panel 1: DSNTIPT" on page 295 are set. If all these data sets do not have the same prefix and suffix, you can change them on installation panel DSNTIPT.

LIBRARY NAME SUFFIX field

The LIBRARY NAME SUFFIX field specifies a suffix for the names of the libraries that are to be used by installation and sample jobs.

Acceptable values:	1 to 17 characters
Default:	NULL
DSNZP*xxx*:	none

The fully qualified data set name cannot exceed 44 characters. Names that exceed eight characters must be in groups of no more than eight characters, separated by periods.

Use a suffix only if you have added a common suffix to the following libraries when you created them in job DSNTIJAE:

prefix.ADSNLOAD	*prefix*.SDSNLINK	*prefix*.SDSNMACS
prefix.SDSNLOAD	*prefix*.SDSNEXIT	*prefix*.ADSNMACS
prefix.SDSNCLST	*prefix*.SDSNSAMP	*prefix*.SDSNDBRM
prefix.SDXRRESL	*prefix*.SDSNIVPD	

If you did not add a common suffix to these libraries, enter their correct data set names on panel DSNTIPT.

To use the default DB2 data set names, specify DSN1010 in the LIBRARY NAME PREFIX field, and leave the LIBRARY NAME SUFFIX field blank.

DATA SET NAME PREFIX field

The DATA SET NAME PREFIX field specifies the input prefix for data sets that are to be created by the CLIST and installation, migration, and sample jobs.

Acceptable values:	1 to 18 characters
Default:	DSN1010
DSNZP*xxx*:	none

Several of these data sets are partitioned data sets, which are deleted, if they exist, and are created or re-created during the tailoring session. If you specify DSNTIDXA or DSNTIDXB in the INPUT MEMBER NAME field, the prefix for fields 1, 2, and 3, and the prefix and suffix for fields 4, 5, and 6 on installation panel "Data set names panel 1: DSNTIPT" on page 295 are set. If all these data sets do not have the same prefix and suffix, you can change them on installation panel DSNTIPT.

DATA SET NAME SUFFIX field

The DATA SET NAME SUFFIX field specifies the suffix for data sets that are to be created by the CLIST and installation, migration, and sample jobs.

Acceptable values:	1 to 17 characters
Default:	NULL
DSNZP*xxx*:	none

The suffix applies to the data sets listed below. The fully qualified data set name cannot exceed 44 characters. Names that exceed eight characters must be in groups of no more than eight characters, separated by periods.

prefix.NEW.SDSNTEMP	*prefix*.NEW.SDSNSAMP	*prefix*.NEW.SDSNCLST
prefix.DBRMLIB.DATA	*prefix*.RUNLIB.LOAD	*prefix*.SRCLIB.DATA

If you do not use a common suffix to these libraries, enter their correct data set names on panel DSNTIPT.

To use the default DB2 data set names, specify DSN1010 in the DATA SET NAME PREFIX field, and leave the DATA SET NAME SUFFIX field blank.

INPUT MEMBER NAME field

The INPUT MEMBER NAME field specifies the member name of the input data set. This data set contains the default DB2 Version 10 parameter values that are to be used for installation and migration, as in *prefix*.SDSNSAMP.*suffix*.

Acceptable values:	1 to 8 characters
Default:	DSNTIDXA
DSNZP*xxx*:	none

Installing DB2

If you are installing DB2 for the first time and not using SAP, use the supplied defaults input member, DSNTIDXA. If you are using SAP, you should specify DSNTIDXB, the SAP-specific input member. If you process the panels several times within a single run of the CLIST, all the previous values that are entered, except edited output data sets, remain the same.

To install DB2 using default parameter values from a previous installation, you must supply the member that contains the output from the previous run of the CLIST. That member was specified in the OUTPUT MEMBER NAME field during the previous run of the CLIST.

Recommendation: Before rerunning the CLIST, use job DSNTIJXZ to refresh the output member from the previous run of the CLIST.

Migrating to conversion mode (from Version 8 or Version 9.1)

If you are migrating to conversion mode (from Version 8 or Version 9.1), you must specify two member names for input values: one in the INPUT MEMBER NAME field, and one in the DATA SET(MEMBER) NAME field. The INPUT MEMBER NAME field must specify a member that contains the default parameter values for the new release (usually DSNTIDXA, or DSNTIDXB if you use SAP). This member is applied first to establish the CLIST parameters.

The member that is specified in the DATA SET(MEMBER) NAME field must be different from the one that is specified in the INPUT MEMBER NAME field. The DATA SET(MEMBER) NAME field must specify a member that contains the values from your previous Version 8 or Version 9.1 release. You can create this member by running job DSNTIJXZ on your previous release. This member is applied last and it overrides the CLIST values that are established by the member that is specified in the INPUT MEMBER NAME field.

Migrating a data sharing group to conversion mode (from Version 8 or Version 9.1)

If you are migrating a data sharing group, enter DSNTIDXA in this field only for the first member that you migrate. For each subsequent member, enter the name that you specified in the OUTPUT MEMBER NAME field when you ran the CLIST to migrate the first member.

Migrating to new-function mode

If you are migrating to new-function mode, enter a member name for input values. The INPUT MEMBER NAME that you specify for migration to new-function mode should be the same as the OUTPUT MEMBER NAME that you specified during migration to conversion mode (from Version 8 or Version 9.1). In a data sharing environment, the INPUT MEMBER NAME should be the same as the OUTPUT MEMBER NAME that was specified during migration of the first member of the data sharing group to Version 10 conversion mode (from Version 8 or Version 9.1).

Updating an existing DB2 subsystem

Before running the CLIST in UPDATE mode, use job DSNTIJXZ to refresh the output member from the previous run of the CLIST. Specify the refreshed member in the INPUT MEMBER NAME field.

The following table lists the data set names that are generated with the *prefix* and *suffix* values from PREFIX and SUFFIX only when the INPUT MEMBER NAME is DSNTIDXA or DSNTIDXB.

Table 53. Resulting data set names when using prefix and suffix parameters

Default library name	CLIST edited library name
prefix.DBRMLIB.DATA	*prefix*.DBRMLIB.DATA*suffix*
prefix.RUNLIB.LOAD	*prefix*.RUNLIB.LOAD.*suffix*
prefix.SRCLIB.DATA	*prefix*.SRCLIB.DATA.*suffix*
prefix.SDSNDBRM	*prefix*.SDSNDBRM.*suffix*
prefix.SDSNLINK	*prefix*.SDSNLINK.*suffix*
prefix.SDSNLOAD	*prefix*.SDSNLOAD.*suffix*
prefix.SDSNMACS	*prefix*.SDSNMACS.*suffix*
prefix.ADSNLOAD	*prefix*.ADSNLOAD.*suffix*
prefix.ADSNMACS	*prefix*.SDSNMACS.*suffix*
prefix.SDSNSAMP	*prefix*.SDSNSAMP.*suffix*
prefix.SDSNCLST	*prefix*.SDSNCLST.*suffix*
prefix.SDSNIVPD	*prefix*.SDSNIVPD.*suffix*
prefix.SDSNC.H	*prefix*.SDSNC.H
prefix.SDXRRESL.	*prefix*.SDXRRESL.*suffix*

Related tasks:

"Adding a new member" on page 691

"Migrating an existing data sharing group to a new release" on page 711

Related reference:

"DATA SET(MEMBER) NAME field" on page 269

OUTPUT MEMBER NAME field

The OUTPUT MEMBER NAME field specifies the member name of the output data set in which to save the values that are entered on the installation panels.

Acceptable values:	1 to 8 characters
Default:	NULL
DSNZP*xxx*:	none

If you do not specify a name, the values are lost when you leave the installation CLIST, and you no longer have the values available for future updates. This member is stored in *prefix*.SDSNSAMP (not the one created by the DSNTINST CLIST). To avoid replacing any members of *prefix*.SDSNSAMP that were shipped with the product, specify DSNTID*xx* as the value of OUTPUT MEMBER NAME, where *xx* is any alphanumeric value except SA, VB, XA, or XB.

Always give a new value in the OUTPUT MEMBER NAME field for a new panel session. You supply the name from your current session in the INPUT MEMBER NAME field for your next session. You should not use the same member name for output as for input.

Recommendation: Write down the output member name that you entered for reference during future sessions.

OTC license usage panel: DSNTIPO1

Use the DSNTIPO1 panel to indicate whether DB2 is to operate under a one-time license charge. This panel is displayed only if you have purchased DB2 10 for z/OS Value Unit Edition (FMID JDBAA1Z) for use with z/OS on zNALC.

```
+----- OTC LICENSE USAGE ------+
| DSNTIPO1                      |
|  Will this subsystem or data  |
|  sharing member be operated   |
|  under an OTC license?        |
|                               |
|     _ 1. No                   |
|       2. Yes                  |
|                               |
|  PRESS:  ENTER to continue    |
|          RETURN to exit       |
|                               |
+-------------------------------+
```

Figure 15. OTC license usage panel: DSNTIPO1

On this panel, specify whether this DB2 subsystem or data sharing member is to be configured to operate using the OTC license. Enter a 1 (NO) or 2 (YES) .

Notice and acceptance of OTC license panel: DSNTIPO2

Use the DSNTIPO1 panel to indicate whether DB2 is to operate under a one-time license charge. This panel is displayed only if you have purchased DB2 10 for z/OS Value Unit Edition (FMID JDBAA1Z) for use with z/OS on zNALC.

Panel DSNTIPO2 is displayed only when you have indicated on panel DSNTIPO1 (OTC license usage) that an OTC license is or will be in effect on this DB2 subsystem or data sharing member. This panel displays the terms of that license and checks for your consent to abide by them. Review the terms by using the ISPF Up and Down keys to scroll through the text.

```
DSNTIPO2             NOTICE AND ACCEPTANCE OF OTC LICENSE
 ===>

  This DB2 subsystem or data sharing member has been selected for operation
  using an OTC license.  Use of this license is controlled by terms defined
  of the following agreement:

  ---------------------------------- TERMS -----------------------------------
                                                                   More:     +
  A "Qualified Sysplex" means a Parallel Sysplex in which z/OS is eligible for
  aggregated zNALC charges as described in the Charges section of the Attach-
  ment for zNALC License Charges on IBM System z.

  Your license authorizes use of the Program only in a validly licensed System
  z New Application License Charge ("zNALC") LPAR for Eligible Workloads which
  are defined as workloads that meet the following criteria:
   * A workload (excluding applications running under Lotus Domino) that in-
  ------------------------------------------------------------------------------

 You must accept these terms of this license in order to proceed:
   LICENSE TERMS ACCEPTED ===>        Enter YES to proceed or RETURN to exit

 PRESS:  UP/DOWN to scroll terms   RETURN to EXIT   HELP for more information
```

Figure 16. Notice and acceptance of OTC license panel: DSNTIPO2

LICENSE TERMS ACCEPTED field (OTC_LICENSE subsystem parameter)

The OTC_LICENSE subsystem parameter indicates whether you accept the terms of the one-time charge (OTC) license.

Acceptable values:	YES
Default:	none
DSNZP*xxx*:	DSN6SYSP.OTC_LICENSE

This field accepts only one value, YES. You cannot pass this panel with any other value in that field. If you do not accept the terms of this license, press the Return key to exit the DB2 installation CLIST.

DB2 catalog and directory panel: DSNTIPA2

The entries on the DSNTIPA2 panel define the name of the ICF catalog for DB2 VSAM data sets and indicate whether to create the specified catalog. The entries also define z/OS Storage Management Subsystem (SMS) classes that are to be used when defining data sets for the DB2 catalog and directory.

The values that you enter on this and each of the subsequent panels are saved in the data set member that you named in the OUTPUT MEMBER NAME field on the Main Panel, DSNTIPA1.

DB2 Version 10 requires SMS, so ensure that the following conditions are met before you proceed:

- SMS is active on the system where this DB2 subsystem resides or is being installed or migrated. SMS must be configured to allocate all data sets for the DB2 catalog and directory in extended format and to use extended addressability.
- Your system storage administrator has provided the required SMS automatic class selection (ACS) routines for data sets and DB2 storage groups created during DB2 installation or migration.
- Your system storage administrator has provided any SMS data classes, management classes, and storage classes that you plan to specify on this panel and the DSNTIPA3 panel.

To help performance and recovery, reduce dependency on a single storage device by using more than one SMS scheme.

```
DSNTIPA2        INSTALL DB2 - DB2 Catalog and Directory
===> _

Check parameters and reenter to change:

 1  CATALOG ALIAS         ===> DSNCAT     Alias of VSAM catalog for DB2 subsystem data sets
 2  DEFINE CATALOG        ===> YES        YES or NO
                             ---------------- SMS -----------------
                             DATA CLASS    MGMT CLASS    STOR CLASS
                             ----------    ----------    ----------
 3  DIRECTORY AND CATALOG ===>         ===>           ===>
    DATA
 4  DIRECTORY AND CATALOG ===>         ===>           ===>
    INDEXES
 5  COMPRESS SPT01        ===> NO     Compress the SPT01 directory space
                                      (NO or YES)
 6  SPT01 INLINE LENGTH   ===> 32138  Max length of SPT01 inline LOB data
                                      (NOINLINE or 1 - 32138)


PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 17. DB2 catalog and directory panel: DSNTIPA2

CATALOG ALIAS field (CATALOG subsystem parameter)

The value of the CATALOG ALIAS field specifies the high-level qualifier for VSAM data sets that are to be used by the DB2 catalog and directory.

The installation CLIST also uses the value in the CATALOG ALIAS field to customize the following statements in DB2 installation and migration jobs:

- The NAME parameter of the DEFINE ALIAS statement for the optional VSAM ICF user catalog for DB2 data sets (job DSNTIJCA)
- The CATALOG parameter of AMS DELETE and DEFINE statements for DB2 catalog (DSNDB06), DB2 directory (DSNDB01), active logs, and bootstrap data sets (jobs DSNTIJDE, DSNTIJEN, and DSNTIJIN)
- The VCAT parameter of various CREATE STOGROUP statements (jobs DSNTIJTM, DSNTIJSG, DSNTEJ1, DSNTEJ2A, and DSNTEJ3M)

Acceptable values: 1 to 8 characters

Default:	DSNCAT
Update:	see Changing the high-level qualifier for DB2 data sets
DSNZP*xxx*:	DSN6SPRM CATALOG

VSAM data set cataloging options: The installation jobs classify the DB2 VSAM data sets, which include recovery log, subsystem, and user data sets. You must create the catalog that defines these data sets through the VSAM ICF.

Recommendation: Use an ICF user catalog to classify all DB2 objects because you can use aliases for user catalogs. Multi-level ICF catalog aliases are acceptable. However, when you use the CREATE STOGROUP statement, you might need to use an alias for the VCAT option, which must be a single-level one- to eight-character name. You can use a master catalog, but only if the name of the master catalog is a single-level name that is one to eight characters.

Ensure that your alias conforms to your local naming conventions. If you are migrating, you should use the same ICF catalog aliases for DB2 Version 10 that you used for your previous version because Version 10 uses many of the data sets from your previous version that are already classified.

Whether you are installing or migrating, DB2 does not require you to catalog all DB2 VSAM data sets in the same ICF catalog nor use the same prefix or catalog alias. However, data sets for the DB2 catalog (DSNDB06) and DB2 directory (DSNDB01) must have the high-level qualifier that you specify in the CATALOG ALIAS field. See the following table for guidelines for the high-level qualifier of data sets. The BSDS is VSAM KSDS. The archive logs are sequential. All other data sets are VSAM linear data sets.

Table 54. DB2 data sets ICF catalog guidelines

DB2 data sets	Guidelines
DB2 directory (DSNDB01) DB2 catalog (DSNDB06)	The first-level qualifier of these data sets must be the value that you specify in the CATALOG ALIAS field. These data sets must be SMS-managed with allocation for extended format and extended addressability.
Active logs Bootstrap data set Default database (DSNDB04) Work file database	The first-level qualifier of these data sets does not need to match the value that you specify in the CATALOG ALIAS field. You can also catalog the data sets in different ICF catalogs using different prefixes.
Archive logs	The first-level qualifier of these data sets does not need to match the value that you specify in the CATALOG ALIAS field. If the archive log data set is allocated on disk, the data set must be cataloged according to the values of the ARCPFX1 or ARCPFX2 subsystem parameters, which are set through the COPY 1 PREFIX and COPY 2 PREFIX fields on panel DSNTIPH. If the archive log data set is allocated on a tape device, cataloging the data set is optional.

Table 54. DB2 data sets ICF catalog guidelines (continued)

DB2 data sets	Guidelines
User table spaces User index spaces	The first-level qualifier of these data sets does not need to match the value that you specify in the CATALOG ALIAS field. You can catalog different user spaces in different ICF catalogs, using different prefixes.

You must provide any catalog connections for log and bootstrap data sets that you catalog apart from the DB2 directory and catalog data sets.

Recommendation: Add an alias for the proper catalog.

Although you can catalog two DB2 subsystems in the same ICF catalog, they must not share the same ICF catalog alias because the alias is the only parameter that makes the data set names unique.

Data set naming conventions: The value that you specify as the ICF catalog alias is also used as the high-level qualifier for DB2 directory and catalog VSAM data sets. The data sets for the DB2 directory and catalog databases and the default database are all VSAM linear data sets (LDSs). Their data set names have the following format:

```
dddddddd.DSNDBn.bbbbbbbb.xxxxxxxx.y0001.Accc
```

With the following definitions:

dddddddd
: The high-level qualifier, which is the value that you supply for the CATALOG ALIAS field.

DSNDB*n*
: A constant identifying this as a DB2 data set. For *n*, use C for a cluster name or D for a data component name.

bbbbbbbb
: The database name. The system database names are:

  DSNDB01
  : The DB2 directory database

  DSNDB04
  : The default database

  DSNDB06
  : The DB2 catalog database

  DSNDB07
  : The default work file database

xxxxxxxx
: The individual table space or index space.

*y***0001.A***ccc*
: The data set. For table spaces and index spaces that can be reorganized with SHRLEVEL CHANGE or SHRLEVEL REFERENCE, *y* can be I or J, depending on whether REORG has been run. For other table spaces and index spaces, *y* is I. *ccc* is the partition number of a partitioned table space or index space, or the relative data set number of a simple or segmented table space or index space.

For example, if the catalog alias is DSNCAT, one of the DB2 directory data sets is named:

```
DSNCAT.DSNDBD.DSNDB01.DBD01.I0001.A001
```

Similarly, one of the DB2 catalog data sets is named:

```
DSNCAT.DSNDBD.DSNDB06.SYSTSVEW.I0001.A001
```

DEFINE CATALOG field

The value of the DEFINE CATALOG field determines whether DB2 is to create a new ICF catalog.

Acceptable values:	YES, NO
Default:	YES
Update:	see Changing the high-level qualifier for DB2 data sets
DSNZP*xxx*:	none

YES Builds a new ICF catalog by using the alias that is specified in the CATALOG ALIAS field.

NO Signals that the catalog that is named in the CATALOG ALIAS field already exists; the CLIST does not create a new one.

If you specify YES, DB2 edits job DSNTIJCA which, when run, creates a user catalog and an alias for that catalog. DB2 creates the high-level qualifier of the catalog name by adding a number to the end of the alias that is defined in the CATALOG ALIAS field. If the alias has fewer than eight characters, DB2 appends a "1" to the end of the alias. For example, if you accept the default of DSNCAT for the CALTALOG ALIAS field, the catalog that DB2 creates is named DSNCAT1.USER.CATALOG. If the alias has eight characters, DB2 changes the last character into a 1. If the last character is already a 1, DB2 changes the 1 to a 2.

DIRECTORY AND CATALOG DATA field (CATDDACL, CATDMGCL, and CATDSTCL subsystem parameters)

The DIRECTORY AND CATALOG DATA field specifies the explicit Storage Management Subsystem (SMS) classes that are to be used for defining VSAM data sets for the DB2 catalog and directory. Your SMS storage administrator defines these SMS classes. To use ACS routines for defining these data sets, leave this field blank.

For migration, values that you enter in this field initially apply only for defining new DB2 catalog and directory data sets. Existing DB2 data sets are not changed until they are redefined as part of a REORG invoked from the DSNTIJEN job.

Table 55. Acceptable values and defaults for columns of DIRECTORY AND CATALOG DATA field

Column	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank
Update:	None	None	None
DSNZP*xxx*:	DSN6SPRM CATDDACL	DSN6SPRM CATDMGCL	DSN6SPRM CATDSTCL

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL

statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

DIRECTORY AND CATALOG INDEXES field (CATXDACL, CATXMGCL, and CATXSTCL subsystem parameters)

The DIRECTORY AND CATALOG INDEXES field specifies the explicit Storage Management Subsystem (SMS) classes that are to be used for defining VSAM data sets for DB2 catalog and directory indexes. Your SMS storage administrator defines these SMS classes. To use ACS routines for defining these data sets, leave this field blank.

For migration, values that you enter in this field initially apply only for defining new data sets for DB2 catalog and directory indexes. Existing DB2 catalog and directory index data sets are not changed until they are redefined as part of a REORG invoked from the DSNTIJEN job.

Table 56. Acceptable values and defaults for columns of DIRECTORY AND CATALOG INDEXES field

Column	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank
Update:	None	None	None
DSNZP*xxx*:	DSN6SPRM CATXDACL	DSN6SPRM CATXMGCL	DSN6SPRM CATXSTCL

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

COMPRESS SPT01 field (COMPRESS_SPT01 subsystem parameter)

The value of the COMPRESS_SPT01 subsystem parameter specifies whether the SPT01 directory table space is to be compressed.

Acceptable values:	NO, YES
Default:	NO
Update:	option 1 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM COMPRESS_SPT01

NO The SPT01 table space is not compressed. This is the default value.

YES The SPT01 table space is compressed during the next reorganization of the table space.

In a data sharing environment, all members should use the same setting for the COMPRESS_SPT01 parameter. The result of a table space reorganization will be determined by the value of the COMPRESS_SPT01 parameter on the member that executes the REORG TABLESPACE utility.

SPT01 INLINE LENGTH field (SPT01_INLINE_LENGTH subsystem parameter)

The SPT01_INLINE_LENGTH subsystem parameter specifies the maximum length of LOB column data in the SPT01 directory table space that is to be maintained in the base table. The length is specified in single-byte characters.

This parameter applies to DB2 Version 10 new-function mode and after step EFM0027 of job DSNTIJEN is completed. In a data sharing environment, all members must use the same setting for this parameter.

Acceptable values:	NOINLINE, 1 to 32138
Default:	32138
Update:	option 1 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SPT01_INLINE_LENGTH

To optimize performance, use the largest possible setting for this parameter, which will decrease the likely number of fetches from the SPT01 LOB auxiliary tables. A large value also increases the efficiency of SPT01 compression when you specify YES for the COMPRESS_SPT01 subsystem parameter. Use a smaller value for the SPT01_INLINE_LENGTH parameter to increase the capacity for storing additional rows in SPT01 if the size of SPT01 exceeds the maximum of 64 GB even with compression activated.

A change to this parameter does not take effect until you issue the SET SYSPARM command. In other words, even if you start or restart DB2 after changing the value, the change is not honored until you issue the SET SYSPARM command. If you increase the setting, DB2 places SPT01 in REORG-pending advisory status (AREO*) after you issue the SET SYSPARM command. If you decrease the setting, DB2 places SPT01 in REORG-pending restrictive status (REORP) after you issue the SET SYSPARM command. If you change the setting of SPT01_INLINE_LENGTH for a data sharing member, change the setting on all members before running the -SET SYSPARM command on any member.

Data parameters panel: DSNTIPA3

The entries on the DSNTIPA3 panel define the storage management preferences for the storage groups and data sets that are created during installation and migration.

Recommendation: If you use dual active logging, place the two active logs on different disk volumes and, ideally, on different channels and control units. To do that, if you specify volume serial numbers explicitly in the LOG COPY 1, BSDS 2 and LOG COPY 2, BSDS 1 fields, do not use the same volume serial number for both fields.

For information about updating parameters with changes that cannot be made by using the panels, see “The update process” on page 501.

```
DSNTIPA3        INSTALL DB2 - DATA PARAMETERS PANEL
===> _

Check parameters and reenter to change:
 1  PERMANENT UNIT NAME    ==> 3390                Device type for MVS catalog
                                                    and partitioned data sets
 2  TEMPORARY UNIT NAME    ==> SYSDA               Device type for
                                                    temporary data sets

                                         ---------------- SMS -----------------
                                VOL/SER    DATA CLASS    MGMT CLASS    STOR CLASS
                                -------    ----------    ----------    ----------
 3  CLIST ALLOCATION       ==>         ==>           ==>           ==>
 4  NON-VSAM DATA          ==>         ==>           ==>           ==>
 5  VSAM CATALOG, DEFAULT, ==>         ==>           ==>           ==>
    AND WORK FILE DATABASE
 6  LOG COPY 1, BSDS 2     ==>         ==>           ==>           ==>
 7  LOG COPY 2, BSDS 1     ==>         ==>           ==>           ==>

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 18. Data parameters panel: DSNTIPA3

PERMANENT UNIT NAME field

The PERMANENT UNIT NAME field specifies the device type or unit name that is to be used for allocating data sets that are created during installation or migration.

Acceptable values:	blank, or valid device type or unit name
Default:	3390
Update:	none
DSNZP*xxx*:	none

The device type or unit name applies to the following data sets:
- ICF catalog
- *prefix*.DBRMLIB.DATA.*suffix*
- *prefix*.RUNLIB.LOAD.*suffix*
- *prefix*.SRCLIB.DATA.*suffix*
- The two data sets that the DSNTINST CLIST generates:
  - *prefix*.NEW.SDSNTEMP
  - *prefix*.NEW.SDSNSAMP

The value of PERMANENT UNIT NAME identifies a direct access unit name for partitioned data sets and the ICF catalog. If you want to use different device types for different data sets, edit the installation or migration jobs after you complete the tailoring session. A common device type is 3390.

The value of PERMANENT UNIT NAME is sometimes used during IVP processing to place output (from COPY TABLESPACE, for example) on the device type that is specified for the PERMANENT UNIT NAME field.

A change to PERMANENT UNIT NAME during migration does not affect the ICF catalog, DB2 catalog, directory, or logs. The new value is used for data sets that are created during migration.

TEMPORARY UNIT NAME field

The value of the TEMPORARY UNIT NAME field specifies the device type or unit name that is to be used for allocating temporary data sets.

Acceptable values:	blank, or valid device type or unit name
Default:	SYSDA
Update:	none
DSNZP*xxx*:	none

The value of TEMPORARY UNIT NAME is the direct access or disk unit name that is used for the precompiler, compiler, assembler, sort, linkage editor, and utility work files in the tailored jobs and CLISTs.

Be aware that this field was previously used to specify the VOLTDEVT system parameter, which provides the device type that is used by DB2 utilities that dynamically allocate temporary data sets. You now specify the VOLTDEVT parameter in the TEMPORARY DS UNIT NAME field on "DB2 utilities parameters panel 2: DSNTIP61" on page 404.

CLIST ALLOCATION field

The CLIST ALLOCATION field controls the storage management properties for the *prefix*.NEW.SDSNTEMP and *prefix*.NEW.SDSNSAMP data sets, which the CLIST creates during the tailoring session.

Table 57. Acceptable values and defaults for field columns

Column	VOL/SER	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid z/OS volume serial number	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank	Blank
Update:	None	None	None	None
DSNZP*xxx*:	None	None	None	None

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

NON-VSAM DATA field

The values of the NON-VSAM DATA field specify the storage management properties for non-VSAM data. Specifically, the NON-VSAM DATA field applies to the *prefix*.DBRMLIB.DATA, *prefix*.RUNLIB.LOAD, and *prefix*.SRCLIB.DATA data sets.

Table 58. Acceptable values and defaults for field columns

Column	VOL/SER	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid z/OS volume serial number	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank	Blank
Update:	None	None	None	None
DSNZP*xxx*:	None	None	None	None

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

VSAM CATALOG, DEFAULT AND WORK FILE DATABASE field

The values in the VSAM CATALOG, DEFAULT AND WORK FILE DATABASE field specify storage management properties. These properties apply to temporary data sets, the default storage group and sample storage group, and the VSAM catalog (if a new one is created).

Table 59. Acceptable values and defaults for field columns

Column	VOL/SER	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid z/OS volume serial number	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank	Blank
Update:	None	None	None	None
DSNZP*xxx*:	None	None	None	None

The CREATE STOGROUP statements for the default storage group and sample storage groups are tailored to include or exclude the VOLUMES, DATACLAS, MGMTCLAS, and STORCLAS parameters according to whether you specify arguments for them in this field.

- If a column is not blank, the corresponding parameter is included, with the value as its argument.
- If a column is blank, the corresponding parameter is excluded.
- If all columns are blank, the VOLUMES parameter is included, with the asterisk as its argument.

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the

corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

For migration, this field has no effect on the VSAM catalog or the default storage group because these objects are provided by the previous release of DB2.

LOG COPY 1, BSDS 2 field

When installing DB2, you can change the values of the LOG COPY 1, BSDS 2 field. Specify the storage management properties for the first copy of the active log data set and the second copy of the bootstrap data set (BSDS).

When migrating DB2, you cannot change any values in this field.

This field has four columns where you can indicate the volume serial number and the SMS classes for data, management, and storage that are to be used for defining particular data sets or storage groups. The acceptable values and default values for each column are shown in the following table. All fields are optional.

Table 60. Acceptable values and defaults for field columns

Column	VOL/SER	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid z/OS volume serial number	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank	Blank
Update:	None	None	None	None
DSNZP*xxx*:	None	None	None	None

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

LOG COPY 2, BSDS 1 field

When installing DB2, you can change the values of the LOG COPY 2, BSDS 1 field. Use this field to specify the storage management properties for the second copy of the active log and the first copy of the BSDS.

When migrating DB2, you cannot change any values in this field.

This field has four columns where you can indicate the volume serial number and the SMS classes for data, management, and storage that are to be used for defining

particular data sets or storage groups. The acceptable values and default values for each column are shown in the following table. All fields are optional.

Table 61. Acceptable values and defaults for field columns

Column	VOL/SER	DATA CLASS	MGMT CLASS	STOR CLASS
Acceptable values:	Blank or a valid z/OS volume serial number	Blank or a valid SMS data class name	Blank or a valid SMS management class name	Blank or a valid SMS storage class name
Default:	Blank	Blank	Blank	Blank
Update:	None	None	None	None
DSNZP*xxx*:	None	None	None	None

A blank value in a column directs the CLIST to remove the corresponding storage-management-related parameters from all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field. A non-blank value is used by the CLIST to customize the corresponding storage-management-related parameters in all JCL, AMS, ALLOC, and DDL statements that are used to create data sets and storage groups that are associated with the field.

Recommendation: To help performance and recovery, avoid dependency on a single storage device by using an SMS scheme that uses more than one volume serial number.

Define group or member panel: DSNTIPK

The DSNTIPK panel follows panel DSNTIPA2 when you select a data sharing function (GROUP, MEMBER, or ENABLE).

You must start DB2 and IRLM group names with an alphabetic character. You should carefully consider the naming convention for a data sharing system. See "Data sharing naming conventions" on page 153 for guidance on planning a naming convention before you choose names for the fields on panel DSNTIPK.

```
DSNTIPK         INSTALL DB2 - DEFINE GROUP OR MEMBER
===> _

Check parameters and reenter to change:

 1  GROUP NAME    ===> DSNCAT    Name of the DB2 group
 2  MEMBER NAME   ===> DSN1      Name of DB2 member in group
 3  WORK FILE DB  ===> DSN1      Work file database name for this member
 4  GROUP ATTACH  ===>           Group attach name for TSO, batch, utilities
 5  SUBGRP ATTACH ===>           Subgroup attach name for this member
 6  COORDINATOR   ===> NO        NO or YES. Allow this member to coordinate
                                  parallel processing on other members.
 7  ASSISTANT     ===> NO        NO or YES. Allow this member to assist
                                  with parallel processing.
 8  RANDOM ATTACH ===> YES       NO or YES. Allow this member to be used
                                  at random for group attachment requests
 9  DEL CF STRUCTS===> NO        NO or YES. Attempt to delete coupling
                                  facility structures during restart.





PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 19. Define group or member panel: DSNTIPK

GROUP NAME field (GRPNAME subsystem parameter)

The GRPNAME subsystem parameter identifies the name of a new or existing DB2 data sharing group that is to be used for this subsystem.

Acceptable values:	1 to 8 characters consisting of letters A - Z, numbers 0 - 9, $, #, @
Default:	DSNCAT
DSNZP*xxx*:	DSN6GRP GRPNAME

The group name encompasses the entire data sharing group and forms the basis for the coupling facility structure names.

To avoid names that IBM uses for its z/OS cross-system coupling facility (XCF) groups, the first character must be an uppercase letter J - Z unless the name begins with DSN. Do not use SYS as the first three characters, and do not use UNDESIG as the group name.

MEMBER NAME field (MEMBNAME subsystem parameter)

The MEMBNAME subsystem parameter specifies the name of a new or existing DB2 data sharing member that is to be used for this subsystem.

Acceptable values:	1 to 8 characters
Default:	DSN1
DSNZP*xxx*:	DSN6GRP MEMBNAME

Recommendation: Use the z/OS subsystem name. DB2 uses this name as its XCF member name. An example of a member name is DB1G. The member name can consist of letters A - Z, numbers 0 - 9, $, #, and @.

WORK FILE DB field

The value of the WORK FILE DB field specifies the name of the work file database that is to be used for the DB2 member.

Acceptable values:	1 to 8 characters
Default:	DSN1
DSNZP*xxx*:	none

Each DB2 member has its own work file database (called DSNDB07 in a non-data-sharing environment). One member of the data sharing group can have the name DSNDB07, but the recommendation is that you use a more meaningful name, such as WRKDSN1. You cannot specify a name that begins with DSNDB unless the name is DSNDB07.

GROUP ATTACH field (SSID DECP value)

The value of the GROUP ATTACH field specifies a generic group attachment name. This name is to be used for batch programs, the call attachment facility (CAF), the RRS attachment facility (RRSAF), IMS, CICS Transaction Server for z/OS, and utilities.

Acceptable values:	1 to 4 characters
Default:	none
dsnhdecp:	SSID

An example of a group attachment name is DB0G. The value that you specify here is also used in the IEFSSN*xx* member of SYS1.PARMLIB.

If you leave this field blank, the SSID DECP value is set to the value that is specified in the SUBSYSTEM NAME field on panel DSNTIPM.

Related concepts:

"Group attachment names and subgroup attachment names" on page 155

Group attachments and subgroup attachments (DB2 Data Sharing Planning and Administration)

Related tasks:

"Registering the command prefixes, member group attachment name, and subgroup attachment name" on page 195

Related reference:

"SUBSYSTEM NAME field (SSID DECP value)" on page 422

SUBGRP ATTACH field

The SUBGRP ATTACH field identifies the name of a new or existing subgroup attachment that is to be associated with this subsystem. A subgroup attachment name can be specified with a group attachment name to organize how TSO, CAF, RRSAF, JDBC, ODBC, and DB2 utilities connections find an active DB2 subsystem.

To specify subgroup attachment names, you must use Version 10 early code.

Acceptable values:	blank, 1 to 4 characters. Characters must be letters A – Z, numbers 0 – 9, $, #, or @
Default:	blank
Update:	modify IEFSSN*xx* and IPL

DSNZP*xxx*: none

The value that you specify for the SUBGRP ATTACH field is used in the IEFSSN*xx* member of SYS1.PARMLIB.

The default value, blank, means that the member that is specified in the MEMBER NAME field is not to be associated with a subgroup attachment. If a subgroup attachment name is entered in the SUBGRP ATTACH field, a group attachment name must also be specified in the GROUP ATTACH field. A subgroup attachment can belong to only one group attachment and cannot have the same name as the group attachment. A member can belong to one subgroup attachment at most.

Related concepts:

"Group attachment names and subgroup attachment names" on page 155

Group attachments and subgroup attachments (DB2 Data Sharing Planning and Administration)

Related tasks:

"Registering the command prefixes, member group attachment name, and subgroup attachment name" on page 195

COORDINATOR field (COORDNTR subsystem parameter)

The COORDNTR subsystem parameter controls whether this DB2 member is to be able to coordinate parallel processing on other members of the group.

Acceptable values: YES, NO
Default: NO
DSNZP*xxx*: DSN6GRP COORDNTR

YES Specifies that a read-only query on this DB2 member can be processed in part on other members of the group.

NO Specifies that only this DB2 member can process a query.

ASSISTANT field (ASSIST subsystem parameter)

The ASSIST subsystem parameter controls whether this DB2 member is to be able to assist a parallelism coordinator with parallel processing.

Acceptable values: YES, NO
Default: NO
DSNZP*xxx*: DSN6GRP ASSIST

NO Causes this member to not be considered as an assistant at either bind time or run time.

YES Causes member to be considered as an assistant at both bind time and run time. To qualify as an assistant at run time, the VPPSEQT and VPXPSEQT buffer pool thresholds of this member must each be greater than zero.

RANDOM ATTACH field (RANDOMATT subsystem parameter)

The RANDOMATT subsystem parameter specifies whether this DB2 member can be used for randomized group attachment or subgroup attachment processing. This setting is ignored when the GROUP ATTACH field is blank.

Acceptable values: YES, NO

Default:	YES
DSNZP*xxx*:	DSN6GRP RANDOMATT

YES
Allows this member to be used, at random, for group attachments or subgroup attachments.

NO Prevents this member from being considered for group attachments or subgroup attachments unless no other members of the defined group are available.

Related concepts:

"Group attachment names and subgroup attachment names" on page 155

Group attachments and subgroup attachments (DB2 Data Sharing Planning and Administration)

DEL CF STRUCTS field (DEL_CFSTRUCTS_ON_RESTART subsystem parameter)

The DEL_CFSTRUCTS_ON_RESTART parameter specifies whether, during restart, DB2 is to attempt to delete the SCA, IRLM lock structure, and any allocated group buffer pools from the coupling facility.

Acceptable values:	NO, YES
Default:	NO
Update:	option 3 on DSNTIPB
DSNZP*xxx*:	DSN6SYSP.DEL_CFSTRUCTS_ON_RESTART

NO During restart, DB2 does not attempt to delete the data sharing group's coupling facility structures.

YES
During restart, DB2 attempts to delete the data sharing group's coupling facility structures. The deletion is successful only if no other DB2 subsystems are connected to the structures when the deletion is attempted. If the deletion is successful, the deleted structures are recovered as necessary during a group restart or group buffer pool RECOVER-pending (GRECP) recovery process. This process might increase the amount of time that DB2 takes to complete a restart.

Use the same setting for the DEL CF STRUCTS field on all members of a data sharing group.

System resource data set names panel: DSNTIPH

The entries on the DSNTIPH panel specify the names of bootstrap data sets, active logs, and archive logs. These entries also specify the number of copies (1 for single logging or 2 for dual logging) for the active and archive logs.

Fields 1, 2, 4, 5, 7, and 8 on the DSNTIPH panel contain the prefix that was entered in the CATALOG ALIAS field on installation panel DSNTIPA2. If you scroll back to panel DSNTIPA2 and change the CATALOG ALIAS value, the values for fields 1, 2, 4, 5, 7, and 8 on DSNTIPH change. When you scroll from panel DSNTIPA2 to panel DSNTIPH, check these values and enter them again if necessary. In MIGRATE or UPDATE modes, the CATALOG ALIAS value cannot be changed, so the fields on DSNTIPH are not affected.

Dual logging improves reliability of recovering and, for active log reads, eases device contention.

Recommendation: Specify dual logging for both active and archive logs. If you specify dual logging, and an error occurs during offload to the archive logs, DB2 restarts the archive process using the second copy of the active log.

```
DSNTIPH         INSTALL DB2 - SYSTEM RESOURCE DATA SET NAMES
===>
DSNT443I Values marked with an asterisk have been updated
Enter data below:

Bootstrap Data Sets (BSDS):

* 1 COPY 1 NAME         ===> DSNCAT.BSDS01
* 2 COPY 2 NAME         ===> DSNCAT.BSDS02

Active Logs:
  3 NUMBER OF COPIES    ===> 2        2 or 1. Number of active log copies
* 4 COPY 1 PREFIX       ===> DSNCAT.LOGCOPY1
* 5 COPY 2 PREFIX       ===> DSNCAT.LOGCOPY2

Archive Logs:
  6 NUMBER OF COPIES    ===> 2        2 or 1. Number of archive log copies
* 7 COPY 1 PREFIX       ===> DSNCAT.ARCHLOG1
* 8 COPY 2 PREFIX       ===> DSNCAT.ARCHLOG2
  9 TIMESTAMP ARCHIVES  ===> NO       NO, YES or EXT (Extended date format)



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 20. System resource data set names: DSNTIPH

COPY 1 NAME field

The value of the COPY 1 NAME field specifies the fully qualified name that is to be used for the first copy of the bootstrap data set.

Acceptable values:	valid data set name; 1 to 33 characters
Default:	DSNCAT.BSDS01 or DSNCAT.DSN1.BSDS01
Update:	option 4 on panel DSNTIPB; not during migration
DSNZP*xxx*:	none

For non-data-sharing environments, the default prefix is DSNCAT.BSDS*xx*. For data sharing environments, the default prefix is DSNCAT.DSN1.BSDS*xx*. The resulting data set name is DSNCAT.BSDS*xx*.A*nnnnnnn* or DSNCAT.DSN1.BSDS*xx*.A*nnnnnnn* where:

- DSNCAT is the value of the CATALOG ALIAS field on panel DSNTIPA2. You can change this portion of the data set prefix on this panel. If you change it, you need to supply another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the value of the MEMBER NAME field on panel DSNTIPK.
- *xx* is **01** for the first copy of the logs and **02** for the second copy.
- A*nnnnnnn* is generated by DB2.

For the definition of a valid data set name, see Valid data set name.

COPY 2 NAME field

The value of the COPY 2 NAME field specifies the fully qualified name that is to be used for the second copy of the bootstrap data set.

Acceptable values:	valid data set name; 1 to 33 characters
Default:	DSNCAT.BSDS02 or DSNCAT.DSN1.BSDS02
Update:	option 4 on panel DSNTIPB; not during migration
DSNZP*xxx*:	none

For non-data-sharing environments, the default prefix is DSNCAT.BSDS*xx*. For data sharing environments, the default prefix is DSNCAT.DSN1.BSDS*xx*. The resulting data set name is DSNCAT.BSDS*xx*.A*nnnnnnn* or DSNCAT.DSN1.BSDS*xx*.A*nnnnnnn* where:

- DSNCAT is the value of the CATALOG ALIAS field on panel DSNTIPA2. You can change this portion of the data set prefix on this panel. If you change it, you need to supply another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the value of the MEMBER NAME field on panel DSNTIPK.
- *xx* is **01** for the first copy of the logs and **02** for the second copy.
- A*nnnnnnn* is generated by DB2.

For the definition of a valid data set name, see Valid data set name.

NUMBER OF COPIES field (TWOACTV subsystem parameter)

The TWOACTV subsystem parameter specifies the number of copies of the active log that DB2 is to maintain: 1 (single logging) or 2 (dual logging).

Acceptable values:	1, 2
Default:	2
Update:	option 4 on panel DSNTIPB; not during migration
DSNZP*xxx*:	DSN6LOGP TWOACTV

Dual logging increases reliability of recovery. If your DB2 subsystem creates copies of the archive log on tape, two tape drives must be available during the offload process.

COPY 1 PREFIX field

The value of the COPY 1 PREFIX field specifies the prefix that is to be used for the first copy of the active log data sets.

Acceptable values:	valid data set name prefix; 1 to 30 characters
Default:	DSNCAT.LOGCOPY1 or DSNCAT.DSN1.LOGCOPY1
Update:	option 4 on panel DSNTIPB; not during migration
DSNZP*xxx*:	none

For non-data-sharing environments, the default prefix is DSNCAT.LOGCOPY*x*. For data sharing environments, the default prefix is DSNCAT.DSN1.LOGCOPY*x*. The resulting data set name is DSNCAT.LOGCOPY*x*.A*nnnnnnn* or DSNCAT.DSN1.LOGCOPY*x*.A*nnnnnnn*, where:

- DSNCAT is the value of the CATALOG ALIAS field on panel DSNTIPA2. You can change this portion of the data set prefix on this panel. If you change it, you

need to specify another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the of the MEMBER NAME field on panel DSNTIPK.
- LOGCOPY is part of the data set prefix that you can change on this panel.
- *x* is **1** for the first copy of the logs and **2** for the second copy.
- *nnnnnnn* is the data set number.

For the definition of a valid data set name, see Valid data set name.

COPY 2 PREFIX field

The value of the COPY 2 PREFIX field specifies the prefix that is to be used for the second copy of the active log data sets.

Acceptable values:	valid data set name prefix; 1 to 30 characters
Default:	DSNCAT.LOGCOPY2 or DSNCAT.DSN1.LOGCOPY2
Update:	option 4 on panel DSNTIPB; not during migration
DSNZP*xxx*:	none

If you are using single logging, accept the default value. Do not leave the entry blank.

For non-data-sharing environments, the default prefix is DSNCAT.LOGCOPY*x*. For data sharing environments, the default prefix is DSNCAT.DSN1.LOGCOPY*x*. The resulting data set name is DSNCAT.LOGCOPY*x*.A*nnnnnnn* or DSNCAT.DSN1.LOGCOPY*x*.A*nnnnnnn*, where:
- DSNCAT is the value of the CATALOG ALIAS field on panel DSNTIPA2. You can change this portion of the data set prefix on this panel. If you change it, you need to specify another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the value of the MEMBER NAME field on panel DSNTIPK.
- LOGCOPY is part of the data set prefix that you can change on this panel.
- *x* is **1** for the first copy of the logs and **2** for the second copy.
- *nnnnnnn* is the data set number.

For the definition of a valid data set name, see Valid data set name.

NUMBER OF COPIES field (TWOARCH subsystem parameter)

The TWOARCH subsystem parameter specifies the number of copies of the archive log that DB2 is to produce during offloading: 1 (single logging) or 2 (dual logging).

Acceptable values:	1, 2
Default:	2
Update:	option 4 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP TWOARCH

Dual logging increases reliability of recovery.

COPY 1 PREFIX field (ARCPFX1 subsystem parameter)

The ARCPFX1 subsystem parameter specifies the prefix that is to be used for the first copy of the archive log data set.

Acceptable values:	valid data set name prefix; 1 to 35 characters
Default:	DSNCAT.ARCHLOG1 or DSNCAT.DSN1.ARCLG1
Update:	option 4 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCPFX1

For definitions of valid data set names, see Valid data set name.

COPY 2 PREFIX field (ARCPFX2 subsystem parameter)

The ARCPFX2 subsystem parameter specifies the prefix that is to be used for the second copy of the archive log data set. If you use single logging, accept the default value.

Acceptable values:	valid data set name prefix; 1 to 35 characters
Default:	DSNCAT.ARCHLOG2 or DSNCAT.DSN1.ARCLG2
Update:	option 4 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCPFX2

Do not leave the COPY 2 PREFIX field blank.

TIMESTAMP ARCHIVES field (TSTAMP subsystem parameter)

The TSTAMP subsystem parameter specifies whether the DB2 archive log data set name is to contain the date and time that the archive log data set was created.

Acceptable values:	NO, YES, EXT
Default:	NO
Update:	option 4 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP TSTAMP

NO The archive data set name does not contain a timestamp.

YES
The maximum allowable length of the user-controlled portion of the archive log prefix is reduced from 35 characters to 19 characters. This reduction in size permits the 16-character date and time qualifiers (timestamp) to be added to the archive log data set prefix. The timestamp format is as follows:

```
.Dyyddd.Thhmmsst,
```

where:

D	is the letter D.
yy	is the last two digits of the year.
ddd	is the day of the year.
T	is the letter T.
hh	is the hour.
mm	is the minutes.
ss	is the seconds.
t	is tenths of a second.

EXT
The archive data set name contains a timestamp with an **extended** date component in the format:

```
.Dyyyyddd.
```

A value of EXT in this field causes the lengths of the values that are entered for field COPY 1 PREFIX and field COPY 2 PREFIX to be audited to ensure that neither exceeds 17 bytes (19 bytes for other settings of TIMESTAMP ARCHIVES).

Data set names panel 1: DSNTIPT

The entries on the DSNTIPT panel establish data set names for the DB2 libraries that are used in the DB2 CLIST and JCL that DB2 provides. The values that you enter on this panel are edited into all pertinent sample and installation jobs.

You can fill in these values in one of three ways: same data set name prefix, no data set name prefix, or a new data set name prefix. The following table summarizes these selections.

Table 62. Summary of values

If you use ...	Then
Same data set name prefix or data set names	Current data sets are deleted and reallocated for installation and migration.
No data set names	No new output is created. Previous output remains intact.
New prefix	Output is saved in new data set. Previous output remains intact.

When you run the CLIST in INSTALL or MIGRATE mode, the following warning message is displayed for any output data set that already exists:

```
DSNT434I WARNING, DATA SETS MARKED WITH ASTERISKS EXIST AND WILL BE OVERWRITTEN
```

To avoid deleting these data sets, take one of the following actions:
- Press Enter to leave the installation process.
- Change the data set names.

Press Enter again if you want to continue; this overwrites your data sets.

When you are in update or ENFM mode, this panel is displayed immediately after panel DSNTIPA1. This allows you to check the SDSNSAMP data set name to see if it is the one you want to use for the DSNTIJUZ job. Data sets are marked with an asterisk but not deleted or reallocated if you use the same name. Instead, the data set is compressed, and only the DSNTIJUZ member is replaced within the data set. Other members in the data set are left unchanged. When the CLIST completes processing and returns to panel DSNTIPA1, the following message is displayed:

```
WARNING MESSAGE(S) WAS PRODUCED DURING CLIST TAILORING SESSION
```

```
DSNTIPT         INSTALL DB2 - DATA SET NAMES PANEL 1
===> _

Data sets allocated by the installation CLIST for edited output:
 1  TEMP CLIST LIBRARY ===> prefix.NEW.SDSNTEMP
 2  SAMPLE LIBRARY     ===> prefix.NEW.SDSNSAMP
Data sets allocated by the installation jobs:
 3  CLIST LIBRARY      ===> prefix.NEW.SDSNCLST
 4  APPLICATION DBRM   ===> prefix.DBRMLIB.DATA.suffix
 5  APPLICATION LOAD   ===> prefix.RUNLIB.LOAD.suffix
 6  DECLARATION LIBRARY===> prefix.SRCLIB.DATA.suffix
Data sets allocated by SMP/E and other methods:
 7  LINK LIST LIBRARY  ===> prefix.SDSNLINK.suffix
 8  LOAD LIBRARY       ===> prefix.SDSNLOAD.suffix
 9  MACRO LIBRARY      ===> prefix.SDSNMACS.suffix
10  LOAD DISTRIBUTION  ===> prefix.ADSNLOAD.suffix
11  EXIT LIBRARY       ===> prefix.SDSNEXIT.suffix
12  DBRM LIBRARY       ===> prefix.SDSNDBRM.suffix
13  IRLM LOAD LIBRARY  ===> prefix.SDXRRESL.suffix
14  IVP DATA LIBRARY   ===> prefix.SDSNIVPD.suffix
15  INCLUDE LIBRARY    ===> prefix.SDSNC.H
 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 21. Data set names panel 1: DSNTIPT

TEMP CLIST LIBRARY and SAMPLE LIBRARY are data sets that are allocated by the installation CLIST for edited output. CLIST LIBRARY is allocated by DSNTIJVC. If the INPUT MEMBER NAME field on installation panel DSNTIPA1 is set to DSNTIDXA or DSNTIDXB, the three data sets default to *prefix*.NEW.SDSNTEMP, *prefix*.NEW.SDSNCLST, and *prefix*.NEW.SDSNSAMP respectively. In these names, *prefix* is the value that is entered for the DATA SET NAME PREFIX field on installation panel DSNTIPA1. The following table shows the job-tailoring fields.

Table 63. Job-tailoring fields

Mode	Tailored output	No tailored output
Installing	All three fields entered	All three fields blank
Migrating	All three fields entered	All three fields blank
Updating	SAMPLE LIBRARY entered	SAMPLE LIBRARY blank

DB2 adds blanks to these fields after a successful tailoring session to avoid writing over the tailored output.

TEMP CLIST LIBRARY field

The value of the TEMP CLIST LIBRARY field specifies the name of the data set where edited CLISTs are to be placed.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.NEW.SDSNTEMP
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

This field must not be blank if you are tailoring output.

SAMPLE LIBRARY field

The value of the SAMPLE LIBRARY field specifies the name of the edited JCL library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.NEW.SDSNSAMP
Update:	option 5 on panel DSNTIPB
DSNZP*xxx*:	none

In update mode, the new sample library data set is not reallocated. It is compressed and member DSNTIJUZ is overwritten. This field must not be blank if you are tailoring output.

CLIST LIBRARY field

The CLIST LIBRARY field identifies the name of the data set into which job DSNTIJVC is to load all CLISTs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.NEW.SDSNCLST
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

This field must not be blank if you are tailoring output.

APPLICATION DBRM field

The value of the APPLICATION DBRM field specifies the name of the library that is to be used for DB2 sample application DBRMs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.DBRMLIB.DATA.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The APPLICATION DBRM field, APPLICATION LOAD field, and DECLARATION LIBRARY field are for DB2-provided sample applications. The names of your own development libraries are most likely different from the default values that are shown here. Job DSNTIJMV references another set of DBRMLIB, RUNLIB, and SRCLIB data sets for SYS1.PROCLIB. See “Installation step 2: Define DB2 to z/OS: DSNTIJMV” on page 508 for more information. These fields must not be blank.

APPLICATION LOAD field

The value of the APPLICATION LOAD field specifies the name of the DB2 sample application load module library.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.RUNLIB.LOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The APPLICATION DBRM field, APPLICATION LOAD field, and DECLARATION LIBRARY field are for DB2-provided sample applications. The names of your own development libraries most likely are different from the default values that are shown here. Job DSNTIJMV references another set of DBRMLIB, RUNLIB, and SRCLIB data sets for SYS1.PROCLIB. See “Installation step 2: Define DB2 to z/OS: DSNTIJMV” on page 508 for more information. These fields must not be blank.

DECLARATION LIBRARY field

The value of the DECLARATION LIBRARY field specifies the name of the DB2 declaration library that is to be used for sample application include files.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SRCLIB.DATA.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The DECLARATION LIBRARY field is for DB2-provided sample applications. The name of your own development library is most likely different from the default value for this field. Job DSNTIJMV references another set of DBRMLIB, RUNLIB, and SRCLIB data sets for SYS1.PROCLIB. See "Installation step 2: Define DB2 to z/OS: DSNTIJMV" on page 508 for more information. This field must not be blank.

LINK LIST LIBRARY field

The LINK LIST LIBRARY field specifies the name of the APF-authorized DB2 early code library.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNLINK.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The LINK LIST LIBRARY field specifies the name of a data set that is allocated during SMP processing. This field must not be blank.

LOAD LIBRARY field

The value of the LOAD LIBRARY field specifies the name of the main APF-authorized DB2 load module library that is to be used by installation and sample jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNLOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The LOAD LIBRARY field specifies the name of a data set that is allocated during SMP processing. This field must not be blank.

MACRO LIBRARY field

The value of the MACRO LIBRARY field specifies the name of the macro library that is to be used by installation and sample jobs. This library contains the CICS and IMS attachment facility macros, the initialization parameter macros, and some data-mapping macros that are needed for some applications.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNMACS.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The MACRO LIBRARY field specifies the name of a data set that is allocated during SMP processing. This field must not be blank.

LOAD DISTRIBUTION field

The value of the LOAD DISTRIBUTION field specifies the name of the distribution load module library that is to be used by installation and sample jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.ADSNLOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The LOAD DISTRIBUTION field specifies the name of a data set that is allocated during SMP processing. This field must not be blank.

EXIT LIBRARY field

The value of the EXIT LIBRARY field specifies the name of the library where your DSNZP*xxx* module, application defaults load module (*dsnhdecp*), and exit routines are to be placed.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNEXIT.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

When you use *prefix*.SDSNLOAD and *prefix*.SDSNEXIT together, list *prefix*.SDSNEXIT first to override the IBM defaults in *prefix*.SDSNLOAD.

The EXIT LIBRARY field specifies the name of a data set that is allocated during SMP processing. This field must not be blank.

DBRM LIBRARY field

The value of the DBRM LIBRARY field identifies the name of the library where the DBRMs that are shipped with DB2 are to be placed.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNDBRM.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

The DBRM LIBRARY field specifies the names of data sets that are allocated during SMP processing. This field must not be blank.

IRLM LOAD LIBRARY field

The value of the IRLM LOAD LIBRARY field specifies the name of the IRLM load library data set that is to be used in the IRLM procedure.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDXRRESL.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

This field must not be blank. It specifies the name of a data set that is allocated during SMP processing.

IVP DATA LIBRARY field

The value of the IVP DATA LIBRARY field specifies the data set name of SDSNIVPD. SDSNIVPD is the SMP/E target library for the DB2 installation verification procedure (IVP) input data and for the expected output from the sample applications.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNIVPD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

INCLUDE LIBRARY field

The value of the INCLUDE LIBRARY field identifies the name of the include library data set that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	*prefix*.SDSNC.H
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

This library is used in the DB2 language PROCS for C and C++.

Data set names panel 2: DSNTIPU

The entries on panels DSNTIPU and DSNTIPW establish data set names for other product libraries. The values that you enter on these panels are edited into sample and installation jobs. If you do not have the product, accept the default values. Jobs for those particular products should not be run.

DB2 makes assumptions about which one of the possible C, C++, and PL/I compilers that you are using, depending on the values you supply or leave as default in the C, C++, and PL/I fields.

Many data set names for other products appear in the jobs. You can enter most of these data sets on this panel and on installation panel DSNTIPW. These names are shown in the following table as they appear in the jobs that are shipped with DB2. Change the names of the data sets if they are different at your site.

Table 64. Data set names that are used in jobs for related products

Job	Data set name	Function
DSNTEJ1	SYS1.MACLIB	Assembler macro library
	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ1L	CEE.SCEELKED	Language Environment linkage editor library
	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ1P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ1U	CEE.SCEERUN	Language Environment dynamic runtime library
	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)

Table 64. Data set names that are used in jobs for related products (continued)

Job	Data set name	Function
DSNTEJ2A	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ2C	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2D	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2E	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2F	SYS1.MACLIB	Assembler macro library
	SYS1.VSF2FORT	VS Fortran runtime library
DSNTEJ2P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ3C	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ3P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ4C	IMSVS.RESLIB	IMS linkage editor library
	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ4P	IMSVS.RESLIB	IMS linkage editor library
	CEE.SCEELKED	PL/I linkage editor base library
	CEE.SCEERUN	PL/I dynamic runtime base library
DSNTEJ5A	CICS410.SDFHLOAD	CICS command translator and linkage editor
	CICS410.SDFHMAC	CICS macro library
	SYS1.MACLIB	Assembler macro library
DSNTEJ5C	CICS410.SDFHLOAD	CICS command translator and linkage editor library
	IGY.SIGYCOMP	Enterprise COBOL for z/OS
		See also the list of libraries that are used by DSNH CLIST in DSNH (TSO CLIST) (DB2 Commands)
DSNTEJ5P	CICS410.SDFHLOAD	CICS command translator and linkage editor library
	CICS410.SDFHPLI	CICS PL/I linkage editor library
	CEE.SCEELKED	Language Environment link editor library
DSNTEJ6D	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6R	CEE.SCEERUN	Language Environment dynamic runtime library
	CEE.SCEEH.H	C library headers
	CEE.SCEELKED	Language Environment linkage editor library
DSNTEJ6S	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6T	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6U	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6V	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6W	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6Z	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ61	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ62	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ63	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ64	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ65	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ66	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ7	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ71	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ73	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ75	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ76	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ77	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ78	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJXP	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTIJMV	SYS1.MACLIB	Assembler macro library
	CEE.SCEERUN	Language Environment dynamic runtime library

Table 64. Data set names that are used in jobs for related products (continued)

Job	Data set name	Function
	CEE.SCEELKED	Language Environment linkage editor library
	EDCPRLK	Language Environment pre-link editor library
	CEE.SCEEMSGP	Language Environment pre-link message file
	CBC.SCCNCMP	C/C++ for z/OS compiler library
	CCNDRVR	C/C++ compiler load module
	CEE.SCEEH.H	C library headers
	CEE.SCLBH.H	C++ library headers
	CEE.SCEECPP	C++ autolink library
	CBC.SCLBCPP	C++ class library
	CICS.SCLBCPP	CICS library for COBOL
	CICS.SDFHLOAD	CICS command translator and linkage editor
	CICS.SDFLPLI	CICS library for PL/I
	prefix.SDSNLOAD(DSNHPC)	DB2 precompiler
	prefix.SDSNLOAD	DB2 linkage editor library
	DSNHPC	DB2 precompiler module
	GDDM®.SADMSAM	GDDM macro library
	GDDM.SADMMOD	GDDM load module library
	IGY.SIGYCOMP	Enterprise COBOL for z/OS compiler library
	IGYCRCTL	Enterprise COBOL for z/OS compiler load module
	IMSVS.RESLIB	IMS linkage editor library
	ISP.SISPLOAD	ISPF ISPLINK module
	IBM.SIBMZCMP	Enterprise PL/I for z/OS compiler library
	SYS1.VSF2FORT	VS Fortran runtime library

When the compiler fields are left blank, the DSNH CLIST and the provided JCL procedures operate differently. The DSNH CLIST issues a specific call statement, using the default load module data set name as the argument of the call. The JCL procedures use the z/OS link list to find the data set in which the load module resides.

Use this panel to define the data set names of your IBM Language Environment, C/370™ or C/C++, COBOL, FORTRAN, and PL/I program product libraries. For more information about these libraries, consult the appropriate program product documentation.

Data sets that are specified on this panel are used by the DB2 installation process to tailor the DB2 language procedures that are generated by installation job DSNTIJMV:

- DSNHASM can be used to prepare DB2 programs using assembly language
- DSNHC can be used to prepare DB2 programs that use C.
- DSNHCPP can be used to prepare DB2 programs that use C++.
- DSNHCPP2 can be used to prepare a class and a client for a DB2 object-oriented program that use C++.
- DSNHCPPS contains the header file search path that is to be used by DSNHCPP and DSNHCPP2.
- DSNHFOR can be used to prepare DB2 programs using FORTRAN
- DSNHICOB can be used to prepare DB2 programs using COBOL
- DSNHPLI can be used to prepare DB2 programs using PL/I
- DSNHSQL can be used to prepare DB2 external SQL procedures

Data sets that you specify on this panel are also used by the DB2 installation process to tailor the DB2 Interactive (DB2I) program preparation CLIST, DSNH.

Use fields 1 through 3 to specify the IBM Language Environment runtime environment, link editor, and pre-link editor message libraries. The CLIST assumes that these libraries are used by all language products except FORTRAN.

Use fields 5 through 12 to define C/370 Version 2 Release 1 (C only), AD/Cycle®, C/370 Version 1 Release 2 (C only), C/C++ for MVS/ESA Version 3 Release 2, C/C++ for OS/390, or C/C++ for z/OS. If you need to define C++, you must define C/C++ for C as well.

If C is not installed on your system:

- Accept the default values for fields 5 through 12.
- Do not run IVP jobs DSNTEJ2D, DSNTEJ2U, DSNTEJ6D, DSNTEJ6R, DSNTEJ6T, DSNTEJ63, DSNTEJ6W, DSNTEJ6Z, DSNTEJ64, DSNTEJ65, DSNTEJ66, DSNTEJ71, DSNTEJ73, and DSNTEJ75.

If C++ is not installed on your system:

- Accept the default values for fields 5 through 12.
- Do not run jobs DSNTEJ2E or DSNTEJ6V. Skip steps PH02US08 and PH02US09 of IVP job DSNTEJ2U.
- Remove all statements that refer to DAYNAME and MONTHNAME from part DSNTESU in the *prefix*.SDSNSAMP library if C++ is not available.

If COBOL is not installed on your system:

- Accept the default value for field 13.
- Do not run IVP jobs DSNTEJ2C, DSNTEJ3C, DSNTEJ4C, DSNTEJ5C, DSNTEJ61, DSNTEJ62, DSNTEJ76, DSNTEJ77, and DSNTEJ78.

If Fortran is not installed on your system:

- Accept the default values for fields 14 and 15.
- Do not run IVP job DSNTEJ2F.

If PL/I is not installed on your system:

- Accept the default values for field 16.
- Do not run IVP jobs DSNTEJ1P, DSNTEJ2P, DSNTEJ3P, DSNTEJ4P, DSNTEJ5P, DSNTEJ6P, DSNTEJ6S, and DSNTEJ6U.

```
DSNTIPU         INSTALL DB2 - DATA SET NAMES PANEL 2
===>

Enter data set names below:
 1  IBM LE RUNTIME LIBRARY   ===> CEE.SCEERUN
 2  IBM LE LINK EDIT LIB     ===> CEE.SCEELKED
 3  IBM LE PRELINK MSG LIB   ===> CEE.SCEEMSGP

 4  HIGH LEVEL ASSEMBLER LIB ===>
 5  C/CPP COMPILER MODULE    ===> CCNDRVR
 6  C/CPP COMPILER LIBRARY   ===> CBC.SCCNCMP
 7  C/CPP HEADER LIBRARY     ===> CEE.SCEEH.H
 8  C/370 COMPILER MESSAGES  ===>
 9  CPP CLASS LIB HEADERS    ===> CEE.SCLBH.H
10  CPP AUTO CALL LIBRARY    ===> CEE.SCEECPP
11  CPP CLASS LIBRARY        ===> CBC.SCLBCPP
12  CPP PROCEDURE LIBRARY    ===> CBC.SCCNUTL
13  COBOL COMPILER LIBRARY   ===>
14  FORTRAN COMPILER LIBRARY ===>
15  FORTRAN LINK EDIT LIB    ===> SYS1.VSF2FORT
16  PL/I COMPILER LIBRARY    ===>

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 22. Data set names panel 2: DSNTIPU

IBM LE RUNTIME LIBRARY field

The IBM LE RUNTIME LIBRARY field identifies the name of the IBM Language Environment dynamic runtime library that is to be used by sample and installation jobs.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CEE.SCEERUN
Update:	cannot change during update
DSNZP*xxx*:	none

The data set name typically includes the qualifier SCEERUN. Leave this field blank if SCEERUN is in the link list. If you enter a value in this field, it is used in the STEPLIB concatenation of the JCL procedures generated by installation job DSNTIJMV, including the following procedures:

- The DSNDBM1 and DSNDIST address spaces for DB2
- The DB2–supplied WLM procedures DSNWLM and DSNCICS
- The DB2 language procedures
- The JOBLIB concatenation of many IVP jobs

If you plan to use DB2 to run XPLINK or AMODE 64 applications, provide the SCEERUN and SCEERUN2 libraries for IBM Language Environment in the z/OS program search order. See Description of Language Environment target libraries (z/OS Language Environment Customization) for more information.

IBM LE LINK EDIT LIB field

The IBM LE LINK EDIT LIB field identifies the name of the IBM Language Environment linkage editor library that sample and installation jobs are to use.

Acceptable values:	valid data set name: see Valid data set name
Default:	CEE.SCEELKED
Update:	cannot change during update

DSNZP*xxx*: none

The data set name typically includes the qualifier SCEELKED. SCEELKED is required to link-edit load modules for C, C++, COBOL, and PL/I application programs. The value that you enter here is used to customize the DB2 language procedures and the DSNH CLIST.

IBM LE PRELINK MSG LIB field

The IBM LE PRELINK MSG LIB field identifies the data set name for messages that are to be issued by the IBM Language Environment pre-linkage editor (EDCPRLK).

Acceptable values:	valid data set name: see Valid data set name
Default:	CEE.SCEEMSGP
Update:	cannot change during update
DSNZP*xxx*:	none

The SMP/E target data set name typically includes the qualifier SCEEMSGP. SCEEMSGP is required to pre-link edit load modules for C, C++, COBOL, and PL/I application programs. The value that you enter here is used to customize DB2 language procedures and the DSNH CLIST.

HIGH LEVEL ASSEMBLER LIB field

The value of the HIGH LEVEL ASSEMBLER LIB field specifies the data set name of the assembler load module library. This value is to be used by sample and installation jobs.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

The data set name typically includes the qualifier SASMMOD1. The value that you specify for this field is used to customize the DSNHASM language procedure and the DSNH CLIST. It is also added to the STEPLIB concatenation of each DB2-provided job that uses the assembler. You can leave this field blank if the library is in the link list.

C/CPP COMPILER MODULE field

The C/CPP COMPILER MODULE field specifies the load module name of the C/370 or C/C++ compiler that is to be used on your system.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CCNDRVR
Update:	cannot change during update
DSNZP*xxx*:	none

The default value is CCNDRVR for C/C++ for z/OS. The value that you enter here is used to determine the configuration for the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, the DSNH CLIST, and the IVP jobs that use C and C++. The value is used as follows:

- If the entry begins with the string “EDC”, the CLIST configures your system to use C/370.

  Important: You cannot use C/370 and C++.
- If the entry begins with the string “CBC”, the CLIST configures your system to use C/C++ for OS/390 and C/C++ for MVS/ESA Version 3 Release 2.
- If the entry begins with any other value, including blanks, the CLIST configures your system to use C/C++ for z/OS.

C/CPP COMPILER LIBRARY field

The C/CPP COMPILER LIBRARY FIELD specifies the name of the compiler library that is to be used for C/C++ or C/370.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CEE.SCCNCMP
Update:	cannot change during update
DSNZP*xxx*:	none

This value is used to customize the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, and the DSNH CLIST. This field can be left blank if the compiler library is in the link list.

The following guidelines apply to the target data set name:
- For C/C++ for z/OS, the SMP/E target data set name typically includes the qualifier SCCNCMP.
- For C/C++ for OS/390, the SMP/E target data set name typically includes the qualifier SCBCCMP.
- For C/C++ for MVS/ESA Version 3 Release 2, the SMP/E target data set name typically includes the qualifier SCBC3CMP.
- For C/370, the SMP/E target data set name typically includes the qualifier SEDCDCMP or SEDCCOMP.

C/CPP HEADER LIBRARY field

The C/CPP HEADER LIBRARY field specifies the name of the header include library that is to be used for C/C++ or C/370.

Acceptable values:	blank, or valid load module name: see Valid data set name
Default:	CEE.SCEEH.H
Update:	cannot change during update
DSNZP*xxx*:	none

- For C/C++, the SMP/E target data set name typically includes the qualifier SCEEH.H.
- For C/370, the SMP/E target data set name typically includes the qualifier SEDCDCMP or SEDCCOMP.

This value is used to customize the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, and the DSNH CLIST. This field can be left blank if the compiler library is in the link list.

C/370 COMPILER MESSAGES field

The value of the C/370 COMPILER MESSAGES field is the name of the message library that is to be used for the C/370 compiler.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

The data set name typically includes the qualifier SEDCDMSG or SEDCMSGS. If you are using C or C++, leave this field blank. This value is used to customize the DSNHC and DSNHSQL language procedures and the DSNH CLIST.

You can specify C/C++ for MVS/ESA V3R1, or C/C++ for MVS/ESA V3R2. These are shipped as single products, but you need to define them separately on this line and on line 1, depending on your need for C or C++. If you specify a name in this field, a STEPLIB is added to the compile step of the DSNHCPP and DSNHCPP2 procedures in job DSNTIJMV, and to the C++ portion of the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

CPP CLASS LIB HEADERS field

The value of the CPP CLASS LIB HEADERS field specifies the data set name for the C++ class header files.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CBC.SCLBH.H
Update:	cannot change during update
DSNZP*xxx*:	none

Accept the default value if C++ is not available on your system.

- For C/C++ for z/OS and C/C++ for OS/390, the data set name typically includes the qualifier CBC.SCLBH.H.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCLB3H.HPP.

If you specify a name in this field, it is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

CPP AUTO CALL LIBRARY field

The value of the CPP AUTO CALL LIBRARY field specifies the data set name of the C++ auto call library.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CEE.SCEECPP
Update:	cannot change during update
DSNZP*xxx*:	none

This data set name typically includes the qualifier SCEECPP. Accept the default value if C++ is not available on your system. The value of the CPP AUTO CALL LIBRARY field is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

CPP CLASS LIBRARY field

The value of the CPP CLASS LIBRARY field specifies the data set name of the C++ class library.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	CEE.SCLBCPP
Update:	cannot change during update
DSNZP*xxx*:	none

Accept the default value if C++ is not available on your system.

- For C/C++ for z/OS and C/C++ for OS/390, the data set name typically includes the qualifier SCLBCPP.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCLB3CPP.

The value of the CPP CLASS LIBRARY field is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

CPP PROCEDURE LIBRARY field

The value of the CPP PROCEDURE LIBRARY field specifies the data set name for the C++ procedure library.

Acceptable values:	valid data set name: see Valid data set name
Default:	CEE.SCCNUTL
Update:	cannot change during update
DSNZP*xxx*:	none

Accept the default value if C++ is not available on your system.

- For C/C++ for z/OS, the data set name typically includes the qualifier SCCNUTL.
- For C/C++ for OS/390, the data set name typically includes the qualifier SCBCUTL.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCBC3UTL.

COBOL COMPILER LIBRARY field

The value of the COBOL COMPILER LIBRARY field identifies the name of the compiler library that is to be used for COBOL.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

The data set name typically includes the qualifier SIGYCOMP. The value that you enter here is used to customize the DSNHICOB language procedure and the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

FORTRAN COMPILER LIBRARY field

The FORTRAN COMPILER LIBRARY field identifies the data set name of the Fortran compiler library that is to be used by the DB2 installation process.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

If you specify a value here, it is used to customize the DSNHFOR language procedure and the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

FORTRAN LINK EDIT LIB field

The FORTRAN LINK EDIT LIB field identifies the data set name of the Fortran linkage editor library that is to be used by the DB2 installation process.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	SYS1.VSF2FORT
Update:	cannot change during update
DSNZP*xxx*:	none

The value that you enter here is used to customize the DSNHFOR language procedure and the DSNH CLIST. Accept the default value if FORTRAN is not available on your system, and do not run IVP job DSNTIJ2F.

PL/I COMPILER LIBRARY field

The value of the PL/I COMPILER LIBRARY field specifies the name of the PL/I compiler library that is to be used by installation and sample jobs.

Acceptable values:	blank, or valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

For Enterprise PL/I, the data set name typically includes the qualifier SIBMZCMP. The value that you enter here is used to customize the DSNHPLI language procedure and the DSNH CLIST. You can leave the field blank if the PL/I compiler library is in the system link list.

Data set names panel 3: DSNTIPW

The entries on the DSNTIPW panel establish data set names for the libraries of other products in your system. The values that are entered on this panel are edited into all pertinent sample and installation jobs.

If you do not have the product, accept the default. The default cannot be blank.

Many data set names for other products appear in the jobs. You can enter most of these data set names on this panel and on installation panel DSNTIPU. These names are shown in the table on Data set names panel 2: DSNTIPU as they appear in the jobs that are shipped with DB2. Change the names of the data sets if they are different at your site.

```
DSNTIPW         INSTALL DB2 - DATA SET NAMES PANEL 3
===>

Enter data set names below:
 1  SYSTEM MACLIB        ===> SYS1.MACLIB
 2  SYSTEM PROCEDURES    ===> SYS1.PROCLIB
 3  SORT LIBRARY         ===> SYS1.SORTLIB
 4  IMS RESLIB           ===>
 5  ISPF ISPLINK MODULE  ===> ISP.SISPLOAD
 6  GDDM MACLIB          ===> GDDM.SADMSAM
 7  GDDM LOAD MODULES    ===> GDDM.SADMMOD
 8  CICS LOAD LIBRARY    ===> CICSTS.SDFHLOAD
 9  CICS MACRO LIBRARY   ===> CICSTS.SDFHMAC
10  CICS COBOL LIBRARY   ===> CICSTS.SDFHCOB
11  CICS PL/I LIBRARY    ===> CICSTS.SDFHPL1
12  CICS EXCI LIBRARY    ===> CICSTS.SDFHEXCI




PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 23. Data set names panel 3: DSNTIPW

SYSTEM MACLIB field

The value of the SYSTEM MACLIB field specifies the data set name of the assembler macro library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	SYS1.MACLIB
Update:	cannot change during update
DSNZP*xxx*:	none

SYSTEM PROCEDURES field

The value of the SYSTEM PROCEDURES field specifies the data set name of the system procedures library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	SYS1.PROCLIB
Update:	cannot change during update
DSNZP*xxx*:	none

SORT LIBRARY field

The value of the SORT LIBRARY field specifies the name of the data set where the DFSORT load module resides.

Acceptable values:	valid data set name: see Valid data set name
Default:	SYS1.SORTLIB
Update:	cannot change during update
DSNZP*xxx*:	none

If you use the DB2 Sort product, DB2 does not use the value of this field. However, you should set this value to a valid data set name in case you need to use DFSORT as a fallback. The following guidance applies if you set the DB2SORT subsystem parameter to DISABLE or DB2 Sort is not installed.

If you use DFSORT, and your load library is not in the link list, you can change the DSNUPROC JCL procedure in job DSNTIJMV.

If DFSORT is installed as your primary z/OS sort product, you do not need to take any action to make DFSORT available to DB2. If multiple releases of DFSORT are installed, ensure that DFSORT R14 is found first in the system search order.

If DFSORT is not installed as your primary z/OS sort product, use one of the following methods to enable DB2 to use DFSORT:

- Add the DFSORT SORTLPA library to the link pack area list; then add the DFSORT SICELINK library to the link list.
- Add the DFSORT SICELINK library to the link list; then add the DFSORT SORTLPA library to the link list.

  Important: If any non-IBM primary sort product is installed in the link list, install DFSORT in the link list, in system search order, after the non-IBM primary sort product.

  Recommendation: If any non-IBM primary sort product is installed in the link pack area, add the DFSORT libraries to the link list.
- Add the DFSORT SICELINK library to the JOBLIB statement; then add the DFSORT SORTLPA library to the JOBLIB statement.
- Add the DFSORT SICELINK library to the STEPLIB DD statement; then add the DFSORT SORTLPA library to the STEPLIB DD statement.
- Add the DFSORT modules to a private library that is equivalent to one of the above configurations.

  Important: If your non-IBM primary sort product is run from a private library, you must use DFSORT in the same way.

If you install DFSORT in the link pack area or link library, you must install it after you install the non-IBM primary sort product.

DB2 uses only the SORT and MERGE functions in DFSORT. If you want to use DFSORT for any other uses outside of this limited DB2 support, you must separately order and license DFSORT.

Related information:

DFSORT Installation and Customization

IMS RESLIB field

The IMS RESLIB field identifies the data set name of the IMS linkage editor library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

If you do not have IMS, you do not need to connect DB2 to IMS, and you can skip the phase 4 sample application jobs DSNTEJ4C and DSNTEJ4P.

ISPF ISPLINK MODULE field

The ISPF ISPLINK MODULE field identifies the data set name of the ISPF load module library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	ISP.SISPLOAD
Update:	cannot change during update
DSNZP*xxx*:	none

GDDM MACLIB field

The GDDM MACLIB field identifies the data set name of the GDDM macro library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	GDDM.SADMSAM
Update:	cannot change during update
DSNZP*xxx*:	none

This field and the GDDM LOAD MODULES field must both have a valid data set name or must both be blank. The data set name that you specify in this field is included in the compile step SYSLIB DD concatenations of DSNHASM, DSNHC, DSNHCOB, DSNHCOB2, DSNHICOB, DSNHICB2, and DSNHPLI. The installation CLIST only generates sample jobs DSNTEJ75 and DSNTEJ78 if you specify a GDDM MACLIB name.

GDDM LOAD MODULES field

The GDDM LOAD MODULES field identifies the data set name of the GDDM load module library that is to be used by sample and installation jobs.

Acceptable values:	valid data set name: see Valid data set name
Default:	GDDM.SADMMOD
Update:	cannot change during update
DSNZP*xxx*:	none

This field and the GDDM MACLIB field must both have a valid data set name or must both be blank. The data set name that you specify in this field is included in the link-edit SYSLIB concatenations of DSNHASM, DSNHC, DSNHCOB, DSNHCOB2, DSNHICOB, DSNHICB2, and DSNHPLI.

CICS LOAD LIBRARY field

The CICS LOAD LIBRARY field identifies the data set name that is to be used for the CICS load module library.

Acceptable values:	valid data set name: see Valid data set name
Default:	CICSTS.SDFHLOAD
Update:	cannot change during update
DSNZP*xxx*:	none

If you do not use CICS, use a blank in the CICS LOAD LIBRARY field.

CICS MACRO LIBRARY field

The CICS MACRO LIBRARY field identifies the data set name that is to be used for the CICS macro library.

Acceptable values:	valid data set name: see Valid data set name
Default:	CICSTS.SDFHMAC
Update:	cannot change during update
DSNZP*xxx*:	none

CICS COBOL LIBRARY field

The CICS COBOL LIBRARY FIELD specifies the data set name for the CICS library that the COBOL programs are to use.

Acceptable values:	valid data set name: see Valid data set name
Default:	CICSTS.SDFHCOB
Update:	cannot change during update
DSNZP*xxx*:	none

CICS PL/I LIBRARY field

The CICS PL/I LIBRARY field identifies the data set name for the CICS library that PL/I programs are to use.

Acceptable values:	valid data set name: see Valid data set name
Default:	CICSTS.SDFHPLI
Update:	cannot change during update
DSNZP*xxx*:	none

CICS EXCI LIBRARY field

The CICS EXCI LIBRARY field identifies the data set name for the CICS library that contains the CICS EXCI load modules.

Acceptable values:	valid data set name: see Valid data set name
Default:	CICS.SDFHEXCI
Update:	cannot change during update
DSNZP*xxx*:	none

Sizes panel 1: DSNTIPD

The entries on the DSNTIPD panel establish the size of the DB2 catalog, directory, and log data sets. Entries on the DSNTIPD panel also define the amount of storage that can be used for storing large object (LOB) values.

The values that you supply on this panel are estimates that are used in calculating sizes for main storage and data sets. The values do not reduce any system limits and do not preclude an application or user from exceeding these estimates, within reasonable limits. For example, if you specify 500 databases, you could create 600. However, if you exceed the values by a large margin, you might encounter a shortage of main storage or use many secondary extents for some data sets. You can usually change the main storage values by using the next panel. If you cannot change these values, see “The update process” on page 501 for the appropriate method.

The installation CLIST contains formulas that calculate the space for each catalog data set and the indexes that DB2 requires for each data set. Data that you enter on this panel is used in these formulas. Use integers; do not enter fractions. You can use K (as in 32 K) for multiples of 1024 bytes and M (as in 16 M) for multiples of 1,048,576 bytes in most fields, but do not exceed the maximum value that is accepted by the field. For example, for the PACKAGE LISTS field, which has a maximum of 32,000, you can enter 31 K, meaning 31,744 bytes. Values of 32 K and above exceed the maximum acceptable value for this field.

Many of the fields on this panel affect the values of the EDMPOOL, EDMSTATC, and EDMDBDC parameters in macro DSN6SPRM.

The defaults for most of the parameters on this panel correspond to the medium storage sizes for the site models that are shown in Table 18 on page 30.

If you are migrating, DB2 Version 10 adopts the DB2 catalog, directory, BSDS, and active logs from your previous version. Therefore, during migration, you cannot change any of the fields on this panel that affect those data sets.

Updating the parameters:

You can alter the characteristics of the DB2 catalog, directory, BSDS, and active and archive logs by using the methods described on "Updating other parameters" on page 505. You cannot actually change the values of these parameters.

```
DSNTIPD         INSTALL DB2 - SIZES PANEL 1
 ===> _

 Check numbers and reenter to change:
  1  DATABASES          ===> 200      In this subsystem
  2  TABLES             ===> 20       Per database (average)
  3  COLUMNS            ===> 10       Per table (average)
  4  VIEWS              ===> 3        Per table (average)
  5  TABLE SPACES       ===> 20       Per database (average)
  6  PLANS              ===> 200      In this subsystem
  7  PLAN STATEMENTS    ===> 30       SQL statements per plan (average)
  8  PACKAGES           ===> 300      In this subsystem
  9  PACKAGE STATEMENTS ===> 10       SQL statements per package (average)
 10  PACKAGE LISTS      ===> 2        Package lists per plan (average)
 11  EXECUTED STMTS     ===> 15       SQL statements executed (average)
 12  TABLES IN STMT     ===> 2        Tables per SQL statement (average)
 13  LOB INLINE LENGTH  ===> 0        Default length of inline LOBs (0-32680)
 14  USER LOB VALUE STG ===> 10240    Max KB storage per user for LOB values
 15  SYSTEM LOB VAL STG ===> 2048     Max MB storage per system for LOB values
 16  USER XML VALUE STG ===> 204800   Max KB storage per user for XML values
 17  SYSTEM XML VAL STG ===> 10240    Max MB storage per system for XML values

 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 24. Sizes panel 1: DSNTIPD

DATABASES field for panel DSNTIPD

The value of the DATABASES field of panel DSNTIPD indicates the estimated number of user databases that are to exist in your subsystem.

Acceptable values:	1 to 64000
Default:	200
Update:	see "The update process" on page 501; cannot update during migration
DSNZP*xxx*:	none

TABLES field

The value of the TABLES field indicates the estimated average number of tables that is to exist for each database in your subsystem.

Acceptable values:	1 to 400
Default:	20
Update:	see “The update process” on page 501; cannot update during migration
DSNZP*xxx*:	none

COLUMNS field

The value of the COLUMNS field identifies the average number of columns per table in your subsystem. This value is to be used in calculating sizes for main storage and data sets.

Acceptable values:	1 to 750
Default:	10
Update:	see “The update process” on page 501; cannot update during migration
DSNZP*xxx*:	none

VIEWS field

The value of the VIEWS field specifies the estimated average number of views that are to exist for each table in your subsystem.

Acceptable values:	1 to 200
Default:	3
Update:	see “The update process” on page 501
DSNZP*xxx*:	none

TABLE SPACES field

The value of the TABLE SPACES field indicates the estimated average number of table spaces that is to exist per database in your subsystem.

Acceptable values:	1 to 400
Default:	20
Update:	see “The update process” on page 501; cannot update during migration
DSNZP*xxx*:	none

PLANS field

The value of the PLANS field is the estimated number of application plans that are to exist in your subsystem. Each program requires a separate application plan.

Acceptable values:	1 to 32000
Default:	200
Update:	see “The update process” on page 501; cannot update during migration
DSNZP*xxx*:	none

PLAN STATEMENTS field

The value of the PLAN STATEMENTS field specifies the estimated average number of SQL statements that are to exist for each application plan.

Acceptable values:	1 to 32000
Default:	30
Update:	see "The update process" on page 501; cannot update during migration
DSNZP*xxx*:	none

PACKAGES field

The value of the PACKAGES field specifies the estimated number of packages that are to exist in the system.

Acceptable values:	1 to 256000
Default:	300
Update:	see "The update process" on page 501; cannot update during migration
DSNZP*xxx*:	none

PACKAGE STATEMENTS field

The value of the PACKAGE STATEMENTS field specifies the estimated number of individual SQL statements that are to exist for each package.

Acceptable values:	1 to 32000
Default:	10
Update:	see "The update process" on page 501; cannot update during migration
DSNZP*xxx*:	none

PACKAGE LISTS field

The value of the PACKAGE LISTS field is the average number of packages that are estimated to be in a package list for each plan.

Acceptable values:	1 to 32000
Default:	2
Update:	see "The update process" on page 501; cannot update during migration
DSNZP*xxx*:	none

EXECUTED STMTS field

The value of the EXECUTED STMTS field is an estimate of the average number of SQL statements that are to be executed per plan.

Acceptable values:	1 to 32000
Default:	15
Update:	see "The update process" on page 501; not during migration
DSNZP*xxx*:	none

The number of SQL statements that are executed can be less than the number of statements that are written.

TABLES IN STMT field

The value of the TABLES IN STMT field indicates the estimated average number of tables that is to be used per SQL statement.

Acceptable values:	1 to 16
Default:	2
Update:	see “The update process” on page 501; cannot update during migration
DSNZP*xxx*:	none

Some SQL statements use more than one table (for example, those using joins, unions, or subselect clauses). Consider how often you expect to use such statements when choosing a value for this parameter.

LOB INLINE LENGTH field (LOB_INLINE_LENGTH subsystem parameter)

The LOB_INLINE_LENGTH subsystem parameter specifies the default length (in bytes) that is to be used for inline LOB columns.

An *inline LOB* is the portion of a LOB column that resides in the containing table's base table space rather than in the column's auxiliary table space. The value in this field is used when the INLINE LENGTH option is not specified in the definition of a LOB column in a CREATE TABLE or ALTER TABLE ADD statement.

Acceptable values:	0 to 32680
Default:	0
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LOB_INLINE_LENGTH

When you enter a value in this field, keep the following points in mind:

- A value of 0 causes LOB columns to be created with no inline attribute by default.
- Odd values are rounded to the next even integer for any DBCLOB column created. For example, if you enter 1023, then 1024 bytes of data for new DBCLOB columns reside in the base table.
- The value of this parameter is in bytes, regardless of the LOB column's data type or subtype.

Related tasks:

Improving performance for LOB data (DB2 Performance)

Related reference:

CREATE TABLE (DB2 SQL)

ALTER TABLE (DB2 SQL)

USER LOB VALUE STG field (LOBVALA subsystem parameter)

The LOBVALA subsystem parameter controls the maximum amount of storage, in KB, that each user is to have for storing LOB values.

Acceptable values:	1 to 2097152
Default:	10240
Update:	option 8 on panel DSNTIPB

DSNZP*xxx*:	DSN6SYSP LOBVALA

SYSTEM LOB VAL STG field (LOBVALS subsystem parameter)

The LOBVALS subsystem parameter sets an upper limit, in MB, for the amount of memory per system that is to be used for storing LOB values.

Acceptable values:	1 to 51200
Default:	2048
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LOBVALS

USER XML VALUE STG field (XMLVALA subsystem parameter)

The XMLVALA subsystem parameter specifies, in KB, an upper limit for the amount of storage that each user is to have for storing XML values.

Acceptable values:	1 to 2,097,152 KB
Default:	204,800 KB
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP XMLVALA

SYSTEM XML VALUE STG field (XMLVALS subsystem parameter)

The XMLVALS subsystem parameter specifies, in MB, an upper limit for the amount of storage that each system can use for storing XML values.

Acceptable values:	1 to 51200 MB
Default:	10240 MB
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP XMLVALS

Sizes panel 2: DSNTIP7

The first two entries on the DSNTIP7 panel define default values for the system-required objects that DB2 creates automatically. DB2 uses these options for any indexes or table spaces that it implicitly creates. The remaining fields provide general default settings for creating table spaces.

```
DSNTIP7           INSTALL DB2 - SIZES PANEL 2
 ===> _

 Enter options for implicitly-created table spaces and indexes:
  1  DEFINE DATA SETS         ===> YES        When creating implicit table space
  2  USE DATA COMPRESSION     ===> NO         When creating implicit table space

 Enter general options for table spaces below:
  3  TABLE SPACE ALLOCATION   ===> 0          Default space allocation in KB for
                                               table spaces
                                               (0 for DB2 default or 1-4194304)
  4  INDEX SPACE ALLOCATION   ===> 0          Default space allocation in KB for
                                               index spaces
                                               (0 for DB2 default or 1-4194304)
  5  VARY DS CONTROL INTERVAL ===> YES        Optimize VSAM CONTROL INTERVAL to
                                               page size for data set allocation
  6  OPTIMIZE EXTENT SIZING   ===> YES        Use sliding secondary quantity
                                               for DB2-managed data sets
  7  REORDERED ROW FORMAT     ===> ENABLE     Use RRF by default when creating
                                               table spaces
  8  DEFAULT PARTITION SEGSIZE===> 32         For explicitly-created table spaces
                                               (0, 4, 8, 12, ..., 56, 60, or 64)

 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 25. Sizes panel 2: DSNTIP7

DEFINE DATA SETS field (IMPDSDEF subsystem parameter)

The IMPDSDEF subsystem parameter specifies whether DB2 is to define the underlying data set for certain implicitly created table spaces and indexes.

This parameter applies to:

- Base table spaces
- Index spaces of indexes that are implicitly created on base tables
- Implicitly created LOB or XML table spaces
- Auxiliary indexes
- Node ID indexes
- Document ID indexes

Acceptable values:	YES, NO
Default:	YES
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IMPDSDEF

YES Underlying data sets are defined when the table space or index space is created. This is the default setting.

NO Underlying data sets are defined when data is first inserted into the table or index.

However, if all of the following conditions are met, underlying data sets are defined when the table space or index space is created:

- The table contains an LOB column.
- The table is in a clone relationship.
- The containing table space is a partition-by-growth table space.

USE DATA COMPRESSION field (IMPTSCMP subsystem parameter)

The IMPTSCMP subsystem parameter determines whether a table space that is created in an implicitly created database is to use data compression.

Acceptable values:	NO, YES
Default:	NO
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IMPTSCMP

NO Disables data compression. This is the default value.

YES Enables data compression for table spaces that are created in implicitly created databases.

The value of the USE DATA COMPRESSION field applies only to implicitly created base table spaces. It is not used for implicitly created LOB or XML table spaces.

TABLE SPACE ALLOCATION field (TSQTY subsystem parameter)

The TSQTY subsystem parameter specifies the amount of space in KB that is to be allocated for primary and secondary spaces. This parameter applies to DB2-defined data sets for table spaces that are created without the PRIQTY and SECQTY clauses.

Acceptable values:	0 to 4194304
Default:	0
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TSQTY

A value of 0 indicates that DB2 is to use a default value of 1 cylinder for a non-LOB table space or 10 cylinders for a LOB table space.

In a data sharing environment, this parameter has group scope.

Related concepts:

How SQL statements are invoked (DB2 SQL)

INDEX SPACE ALLOCATION field (IXQTY subsystem parameter)

The IXQTY subsystem parameter controls the amount of space in KB that is to be allocated for primary and secondary spaces. This parameter applies to DB2-defined data sets for index spaces that are created without the PRIQTY and SECQTY clauses.

Acceptable values:	0 to 4194304
Default:	0
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IXQTY

A value of 0 indicates that DB2 is to use a default allocation of one cylinder.

In a data sharing environment, this parameter has group scope.

VARY DS CONTROL INTERVAL field (DSVCI subsystem parameter)

The DSVCI subsystem parameter controls whether DB2-managed data sets that are created by CREATE TABLESPACE or CREATE INDEX statements are to have variable VSAM control intervals.

Acceptable values:	YES, NO
Default:	YES
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DSVCI

YES DB2 creates DB2-managed data sets for table spaces and index spaces. The DB2-managed data sets have a VSAM control interval that corresponds to the buffer pool that is used for the table space or index space.

NO DB2-managed data sets are created with a fixed control interval of 4 KB, regardless of the buffer pool size.

This parameter is online-updatable. If you change this value from NO to YES, any pre-existing or migrated data sets remain in 4 KB control intervals until they are redefined. If you change this value from YES to NO, any pre-existing or migrated data sets in 8 KB, 16 KB, or 32 KB control intervals remain in those control intervals until they are redefined. You can explicitly redefine a data set. In addition, data sets are implicitly redefined by utilities such as LOAD REPLACE, REORG TABLESPACE, or RECOVER.

OPTIMIZE EXTENT SIZING field (MGEXTSZ subsystem parameter)

The MGEXTSZ subsystem parameter controls whether secondary extent allocations for DB2-managed data sets are to be sized according to a sliding scale. Sizing the secondary extents on a sliding scale optimizes the likelihood of reaching the maximum data set size before secondary extents are exhausted.

Acceptable values:	YES, NO
Default:	YES
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MGEXTSZ

YES DB2 automatically optimizes the secondary extent allocations of data sets for table spaces and index spaces that have a SECQTY value of greater than zero. When all secondary extents are exhausted for the first data set of a nonpartitioned table space or a nonpartitioned index space that has a SECQTY value of greater than zero, the primary space allocation of each subsequent data set is the larger of the SECQTY setting and the value that is derived from the sliding scale algorithm.

NO Secondary extent allocations are managed manually. For nonpartitioned table spaces and nonpartitioned index spaces, when all extents of the first data set are exhausted, the primary space allocation of each subsequent data set is always the PRIQTY setting.

When the sliding scale is used, secondary extent allocations that are allocated earlier are smaller than those allocated later, until a maximum allocation is reached. The maximum allocation is 127 cylinders for data sets with a maximum size of 16 GB or less, and 559 cylinders for data sets with a maximum size of 32 GB or 64 GB.

Secondary extent allocations for data sets in implicitly created table spaces are not influenced by this setting because they always use the sliding scale.

In migration mode, the input migration value for this field is overridden to YES, so you need to reset it to NO if you want to continue to disable secondary extent optimization.

REORDERED ROW FORMAT field (RRF subsystem parameter)

The value of this parameter specifies whether most newly created table spaces are to store data in reordered row format (RRF) or basic row format (BRF) by default.

Acceptable values:	ENABLE, DISABLE
Default:	ENABLE
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RRF

ENABLE
Newly created table spaces or newly added partitions that are created by ALTER ADD PARTITION statements on partition-by-growth table spaces will be created in RRF. Existing BRF table spaces will be converted to RRF by running LOAD REPLACE or REORG TABLESPACE.

DISABLE
Newly created table spaces, including universal table spaces, and newly added partitions that are created by ALTER ADD PARTITION statements on partition-by-growth table spaces will be created in BRF. Existing BRF table spaces will not be converted to RRF by LOAD REPLACE or REORG TABLESPACE.

The following additional restrictions apply, regardless of the setting of the REORDERED ROW FORMAT field:

- Newly created partitions for existing BRF table spaces that contain tables with an EDITPROC will always be in BRF format.
- XML table spaces will always be in RRF.
- Existing RRF table spaces will never be converted to BRF by LOAD REPLACE or REORG TABLESPACE except when requested explicitly with the ROWFORMAT keyword.
- Universal table spaces that are involved in a CLONE relationship will never be converted from BRF to RRF or from RRF to BRF by LOAD REPLACE or REORG TABLESPACE, regardless of the subsystem parameter or utility keyword setting.

DEFAULT PARTITION SEGSIZE field (DPSEGSZ subsystem parameter)

The DPSEGSZ subsystem parameter specifies the default segment size that is to be used for partitioned table spaces when the CREATE TABLESPACE statement does not include the SEGSIZE parameter.

This parameter is deprecated in Version 10.

Acceptable values:	0, 4, 8, 12, ..., 60, 64 (multiples of 4)
Default:	32
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP.DPSEGSZ

The type of table space that is created depends on whether the CREATE TABLESPACE statement specifies the MAXPARTITIONS and NUMPARTS clauses. The following table shows the relationship among the DPSEGSZ subsystem parameter, the MAXPARTITIONS and NUMPARTS clauses, and the type of table space that is created by the CREATE TABLESPACE statement.

Table 65. Effect of DPSEGSZ subsystem parameter, MAXPARTITIONS clause, and NUMPARTS clause on the type of table space that is created

DPSEGSZ setting	MAXPARTITIONS clause	NUMPARTS clause	Type of table space created
0	not specified	not specified	segmented table space with SEGSIZE 4
0	not specified	specified	classic partitioned table space
0	specified	not specified	partition-by-growth table space with SEGSIZE 32
0	specified	specified	partition-by-growth table space with SEGSIZE 32
$n > 0$	not specified	not specified	segmented table space with SEGSIZE 4
$n > 0$	not specified	specified	range-partitioned table space with SEGSIZE *n*
$n > 0$	specified	not specified	partition-by-growth table space with SEGSIZE *n*
$n > 0$	specified	specified	partition-by-growth table space with SEGSIZE *n*

The DPSEGSZ subsystem parameter has no effect on the creation of segmented table spaces and implicitly created table spaces. The table above includes segmented table spaces for completeness.

Work file database panel: DSNTIP9

The fields on the DSNTIP9 panel configure the 4 KB and 32 KB page size table spaces in the DB2 work file database.

The values in these fields determine the following:

- The total amount of space that is available for each type of table space.
- The number of table spaces that are to be created during the installation or migration process for each type of table space. In INSTALL mode, you must

specify at least 1 for each of the 4 types of work file table spaces: 4 KB sort work, 32 KB sort work, 3 KB temporary work, and 32 KB temporary work. In MIGRATE mode, you can specify 0 for any or all of these table spaces.

- The segment size of each type of table space. You can change the segment size in installation mode or if you are migrating from Version 9.
- The maximum amount of temporary storage in the work file database that a single agent can use at any given time for any type of usage. The usage can be for sort work files, created global temporary tables, declared global temporary tables, scrollable cursors result tables, and so on.

The settings on this panel are used to tailor installation job DSNTIJTM, which creates the work file database. For more information about job DSNTIJTM, see "Installation step 16: Create default storage group, define temporary work files, and bind DB2 REXX Language Support: DSNTIJTM" on page 532. To determine the storage requirements for work file database table spaces, see "Work file database storage requirements" on page 35.

The work file database is used as temporary space for SQL statements and triggers that require working storage. In particular, this includes statements that use:

GROUP BY or HAVING (without index)	IN (subselect)
ORDER BY (without index)	ANY (subselect)
DISTINCT (without index)	SOME (subselect)
UNION (except UNION ALL)	ALL (subselect)
EXISTS (subselect)	Some joins

From V8 Beginning in DB2 Version 9.1, the work file database is also used by declared global temporary tables and for static scrollable cursor implementation, which used the TEMP database in previous versions. If you are migrating from Version 8 and created a TEMP database in a previous version of DB2, it is not used in Version 10. You can drop it anytime after completing migration to DB2 Version 10 conversion mode from Version 8. If you drop it before entering new-function mode, you need to re-create it if you fall back to Version 8. From V8

Unlike temporary work table spaces, sort work table spaces cannot increase in size beyond the primary allocation. You can add a work file table space or change the size of an existing one by deleting and redefining it. DB2 needs to be started for these activities and you do not need to stop the work file database. All DB2 users share the work file database table spaces. You cannot use utilities on the work file database table spaces.

You can create additional work file table spaces at any time, including during migration. Creating additional work file table spaces can improve DB2 performance by reducing device contention among applications that require working storage.

For the work file database, the maximum number of data sets per table space is 32, and the maximum size of a data set is 2 GB in conversion mode (from both Version 8 and Version 9.1).

As of Version 10 new-function mode, the work file database can contain partition-by-growth table spaces to increase the available space for declared global temporary tables and reduce the possibility of receiving SQLCODE -904. A

partition-by-growth table space can grow up to 128 TB. The maximum number of partitions to which the partition-by-growth table space can grow and the maximum size of the table space are determined by the MAXPARTITIONS and DSSIZE values that are specified for the table space.

Recommendation: After migration to Version 10 new-function mode, drop work file table spaces and re-create them as partition-by-growth table spaces. (Existing table spaces cannot be altered to become partition-by-growth table spaces.)

Updating the parameters: You can alter the characteristics of the DB2 catalog, directory, work file databases, BSDS, and active and archive logs by using the methods described on "The update process" on page 501. You can change only the value of MAX TEMP STG/AGENT directly on the panel.

```
 DSNTIP9         INSTALL DB2 - WORK FILE DATABASE
 ===>

Enter options for creating sort work table spaces:
  1  4K SORT PRIMARY SPACE ===> 20         Total primary space (MB)
  2  4K SORT WORK TS'S     ===> 1          Number of table spaces to create
  3  4K SORT WORK SEG SIZE ===> 16         Segment size
  4  32K SORT PRIMARY SPACE===> 20         Total primary space (MB)
  5  32K SORT WORK TS'S    ===> 1          Number of table spaces to create
  6  32K SORT WORK SEG SIZE===> 16         Segment size
Enter options for creating temporary (DGTT) work file table spaces:
  7  4K TEMP PRIMARY SPACE ===> 20         Total primary space (MB)
  8  4K TEMP WORK TS'S     ===> 1          Number of table spaces to create
  9  4K TEMP WORK SEG SIZE ===> 16         Segment size
 10  32K TEMP PRIMARY SPACE===> 20         Total primary space (MB)
 11  32K TEMP WORK TS'S    ===> 1          Number of table spaces to create
 12  32K TEMP WORK SEG SIZE===> 16         Segment size
Enter other work file database options:
 13  MAX TEMP STG/AGENT    ===> 0          Maximum MB of temp storage per agent
 14  SEPARATE WORK FILES   ===> NO         Separate work according to TS type
 15  MAX TEMP RID          ===> NOLIMIT    Max RID blocks of temp storage per
                                           RID list (NOLIMIT, NONE, 1 -329166)
 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 26. Work file database panel: DSNTIP9

Related tasks:

Creating additional work file table spaces to reduce contention (DB2 Performance)

4K SORT PRIMARY SPACE field

The value of the 4K SORT PRIMARY SPACE field specifies the total amount of space for all 4 KB sort work table spaces in the work file database. This value is specified in MB, but you can enter the value in GB by specifying a suffix of G, for example `1024G`.

Unlike temporary work table spaces, sort work table spaces are always segmented table spaces and cannot increase in size.

Acceptable values:	for installation: 1 to 32,702,464; for migration: 0 to 32,702,464
Default:	for installation: 20; for migration: 0
Update:	none
DSNZP*xxx*:	none

The value of the 4K SORT PRIMARY SPACE field is divided by the value of the 4K SORT WORK TS'S field to determine the primary space quantity (PRIQTY) for creating each 4 KB type sort work file table space. For example, if you specify a value of 16 for the 4K SORT PRIMARY SPACE field and a value of 4 for the 4K SORT WORK TS'S field, the PRIQTY for each 4 KB type table space is 4096 KB.

The secondary space quantity (SECQTY) value for each 4 KB type table space is always 0.

In migration mode, the value of the 4K SORT PRIMARY SPACE field is used only when additional 4 KB type sort work file table spaces are created. DB2 does not take into account the space that is already used by existing 4 KB type work files.

4K SORT WORK TS'S field

The value of the 4K SORT WORK TS'S field specifies the number of 4 KB sort work file table spaces that are to be created.

Acceptable values:	for installation: 1 to 499; for migration: 0 to 500
Default:	for installation: 1 for migration: 0
Update:	none
DSNZP*xxx*:	none

The sum of the values of the 4K SORT WORK TS'S field, the 32K SORT WORK TS'S field, the 4K TEMP WORK TS'S field, and the 32K TEMP WORK TS'S field cannot exceed 500.

In migration mode, if you enter a non-zero value, the value is used for creating new 4 KB type sort work file table spaces. DB2 does not take into account the number of existing 4 KB type sort work file table spaces.

4K SORT WORK SEG SIZE field

The value of the 4K SORT WORK SEG SIZE field specifies the segment size that is to be used for 4 KB sort work file table spaces. The value must be a multiple of 4 between 4 and 64, inclusive.

Acceptable values:	4, 8, 12, ..., 60, 64 (multiples of 4)
Default:	16
Update:	none
DSNZP*xxx*:	none

From V8 If you are migrating from Version 8, the value of the 4K SORT WORK SEG SIZE field is always set to 16 and cannot be changed. From V8

The value of the 4K SORT WORK SEG SIZE field does not apply to existing 4 KB work file table spaces.

32K SORT PRIMARY SPACE field

The value of the 32K SORT PRIMARY SPACE field specifies the total amount of space for all 32 KB sort work table spaces in the work file database. This value is specified in MB but you can enter the value in GB by specifying a suffix of G, for example `1024G`.

Unlike temporary work table spaces, sort work table spaces are always segmented table spaces and cannot increase in size.

Acceptable values:	for installation: 1 to 32,702,464; for migration: 0 to 32,702,464
Default:	for installation: 20; for migration: 0
Update:	none
DSNZP*xxx*:	none

The value of the 32K SORT PRIMARY SPACE field is divided by the value of the 32K SORT WORK TS'S field to determine the primary space quantity (PRIQTY) for creating each 32 KB type sort work file table space. For example, if you specify a value of 16 for the 32K SORT PRIMARY SPACE field and a value of 4 for the 32K SORT WORK TS'S field, the PRIQTY for each 32 KB type table space is 4096 KB.

The secondary space quantity (SECQTY) value for each 32 KB type table space is always 0.

In migration mode, the value of the 32K SORT PRIMARY SPACE field is used only when additional 32 KB type sort work file table spaces are created. DB2 does not take into account the space that is already used by existing 32 KB type work files.

32K SORT WORK TS'S field

The value of the 32K SORT WORK TS'S field specifies the number of 32 KB sort work file table spaces that are to be created.

Acceptable values:	for installation: 1 to 499; for migration: 0 to 500
Default:	for installation: 1 for migration: 0
Update:	none
DSNZP*xxx*:	none

The sum of the values of the 4K SORT WORK TS'S field, the 32K SORT WORK TS'S field, the 4K TEMP WORK TS'S field, and the 32K TEMP WORK TS'S field cannot exceed 500.

In migration mode, if you enter a non-zero value, the value is used for creating new 32 KB type sort work file table spaces. DB2 does not take into account the number of existing 32 KB type sort work file table spaces.

32K SORT WORK SEG SIZE field

The value of the 32K SORT WORK SEG SIZE field specifies the segment size that is to be used for 32 KB sort work file table spaces. The value must be a multiple of 4 between 4 and 64, inclusive.

Acceptable values:	4, 8, 12, ..., 60, 64 (multiples of 4)
Default:	16
Update:	none
DSNZP*xxx*:	none

From V8 If you are migrating from Version 8, the value of the 32K SORT WORK SEG SIZE field is always set to 16 and cannot be changed. From V8

The value of the 32K SORT WORK SEG SIZE field does not apply to existing 32 KB work file table spaces.

4K TEMP PRIMARY SPACE field

The value of the 4K TEMP PRIMARY SPACE field specifies the total amount of space for all 4 KB temporary work table spaces in the work file database. This value is specified in MB, but you can enter the value in GB by specifying a suffix of G, for example `1024G`.

Unlike sort work table spaces, temporary work table spaces can increase in size beyond the primary allocation. In installation mode, temporary work table spaces, unlike sort work table spaces, are created as either partition-by-growth table spaces or segmented table spaces, depending on the amount of primary space per table space.

Acceptable values:	for installation: 1 to 8,388,608,000; for migration: 0 to 32,702,464
Default:	for installation: 20; for migration: 0
Update:	none
DSNZP*xxx*:	none

The value of the 4K TEMP PRIMARY SPACE field is divided by the value of the 4K TEMP WORK TS'S field to determine either of the following values:

- The primary space quantity (PRIQTY)
- The maximum number of partitions (MAXPARTITIONS) and the maximum size (DSSIZE) of each partition

The PRIQTY value is used for creating each 4 KB type temporary work table space as a segmented table space. The MAXPARTITIONS and DSSIZE values are used for creating each 4 KB type temporary work table space as a partition-by-growth table space.

In migration mode, only segmented table spaces can be created in the work file database. The value of the 4K TEMP PRIMARY SPACE field is used only when additional 4 KB type work file table spaces are created. DB2 does not take into account the space that is already used by existing 4 KB type work files. If necessary, the CLIST curtails the calculated space at 64 GB, which is the maximum for a 4 KB page size segmented table space.

In installation mode, when there is 1 GB or less per table space, segmented table spaces are created by using the PRIQTY parameter. Otherwise, partition-by-growth table spaces are created by using the DSSIZE and MAXPARTITIONS parameters. If necessary, the CLIST curtails the calculated space at 16,384 GB, which is the maximum for a 4 KB page size partition-by-growth table space.

For example:

- In either installation or migration mode, if you specify a value of 16 for the 4K TEMP PRIMARY SPACE field and a value of 4 for the 4K TEMP WORK TS’S field, the PRIQTY for each 4 KB type table space is 4096 KB.
- In installation mode, if you specify a value of 16G for the 4K TEMP PRIMARY SPACE field and a value of 4 for the 4K TEMP WORK TS’S field, the MAXPARTITIONS for each 4 KB type table space is 1 with a DSSIZE of 4.

- In migration mode, if you specify a value of 16G for the 4K TEMP PRIMARY SPACE field and a value of 4 for the 4K TEMP WORK TS'S field, the PRIQTY for each 4 KB type table space is 4,194,304 KB.

4K TEMP WORK TS'S field

The value of the 4K TEMP WORK TS'S field specifies the number of 4 KB temporary work file table spaces that are to be created.

Acceptable values:	for installation: 1 to 499; for migration: 0 to 500
Default:	for installation: 1; for migration: 0
Update:	none
DSNZP*xxx*:	none

The sum of the values of the 4K SORT WORK TS'S field, the 32K SORT WORK TS'S field, the 4K TEMP WORK TS'S field, and the 32K TEMP WORK TS'S field cannot exceed 500.

In migration mode, if you enter a non-zero value, the value is used when new 4 KB type temporary work file table spaces are created. DB2 does not take into account the number of existing 4 KB type temporary work file table spaces.

4K TEMP WORK SEG SIZE field

The value of the 4K TEMP WORK SEG SIZE field specifies the segment size that is to be used for 4 KB temporary work file table spaces. The value must be a multiple of 4 between 4 and 64, inclusive.

Acceptable values:	4, 8, 12, ..., 60, 64 (multiples of 4)
Default:	16
Update:	none
DSNZP*xxx*:	none

From V8 If you are migrating from Version 8, the value of the 4K TEMP WORK SEG SIZE field is always set to 16 and cannot be changed. The value of the field does not affect existing 4 KB temporary work file table spaces. From V8

32K TEMP PRIMARY SPACE field

The value of the 32K TEMP PRIMARY SPACE field specifies the total amount of space for all 32 KB temporary work table spaces in the work file database. This value is specified in MB, but you can enter the value in GB by specifying a suffix of G, for example `1024G`.

Unlike sort work table spaces, temporary work table spaces can increase in size beyond the primary allocation. In installation mode, temporary work table spaces, unlike sort work table spaces, are created as either partition-by-growth or segmented table spaces, depending on the amount of primary space per table space.

Acceptable values:	for installation: 1 to 67,108,864,000; for migration: 0 to 32,702,464
Default:	for installation: 20; for migration: 0
Update:	none
DSNZP*xxx*:	none

The value of the 32K TEMP PRIMARY SPACE field is divided by the value of the 32K TEMP WORK TS'S field to determine either of the following values:

- The primary space quantity (PRIQTY)
- The maximum number of partitions (MAXPARTITIONS) and the maximum size (DSSIZE) of each partition

The PRIQTY value is used for creating each 32 KB type temporary work table space as a segmented table space. The MAXPARTITIONS and DSSIZE values are used for creating each 32 KB type temporary work table space as a partition-by-growth table space.

In migration mode, only segmented table spaces can be created in the work file database. The value of TEMP 32K SPACE is used only when additional 32 KB work file table spaces are created. DB2 does not take into account the space that is already used by existing 32 KB work files. If necessary, the CLIST curtails the calculated space at 64 GB, which is the maximum for a 32 KB page size segmented table space.

In installation mode, when there is 1 GB or less per table space, segmented table spaces are created using the PRIQTY parameter. Otherwise, partition-by-growth table spaces are created using the DSSIZE and MAXPARTITIONS parameters. If necessary, the CLIST curtails the calculated space at 131,072 GB, which is the maximum for a 32 KB page size partition-by-growth table space.

For table spaces that are larger than 16,384 GB, data sets must be associated with a Data Facility Storage Management Subsystem (DFSMS) data class that has been specified with extended format and extended addressability.

For example:

- In either installation or migration mode, if you specify a value of 16 for the 32K TEMP PRIMARY SPACE field and a value of 4 for the 32K TEMP WORK TS'S field, the PRIQTY value for each 32 KB type table space is 4096 KB.
- In installation mode, if you specify a value of 16G for the 32K TEMP PRIMARY SPACE field and a value of 4 for the 32K TEMP WORK TS'S field, the MAXPARTITIONS value for each 32 KB type table space is 1 and the DSSIZE is 4.
- In migration mode, if you specify a value of 16G for the 32K TEMP PRIMARY SPACE field, the PRIQTY value for each 32 KB type table space is 4,193,304 KB.

32K TEMP WORK TS'S field

The value of the 32K TEMP WORK TS'S field specifies the number of 32 KB temporary work file table spaces that are to be created.

Acceptable values:	for installation: 1 to 499; for migration: 0 to 500
Default:	for installation: 1; for migration: 0
Update:	none
DSNZP*xxx*:	none

The sum of the values of the 4K SORT WORK TS'S field, the 32K SORT WORK TS'S field, the 4K TEMP WORK TS'S field, and the 32K TEMP WORK TS'S field cannot exceed 500.

In installation mode, you must enter a value of at least 1.

In migration mode, if you do not have an existing 32 KB work file table space, enter a non-zero value. If you enter a non-zero value, the value is used for creating new 32 KB type temporary work file table spaces. DB2 does not take into account the number of existing 32 KB type temporary work file table spaces.

32K TEMP WORK SEG SIZE field

The value of the 32K TEMP WORK SEG SIZE field specifies the segment size that is to be used for 32 KB temporary work file table spaces. The value must be a multiple of 4 between 4 and 64, inclusive.

Acceptable values:	4, 8, 12, ..., 60, 64 (multiples of 4)
Default:	16
Update:	none
DSNZP*xxx*:	none

From V8 If you are migrating from Version 8, the value of the 32K TEMP WORK SEG SIZE field is always set to 16 and cannot be changed. The value of 32K TEMP WORK SEG SIZE field does not affect existing 32 KB work file table spaces. From V8

MAX TEMP STG/AGENT field (MAXTEMPS subsystem parameter)

The MAXTEMPS subsystem parameter determines the maximum amount of temporary storage in the work file database that a single agent can use at any given time for any type of usage. The usage can be for sort work files, created global temporary tables, declared global temporary tables, scrollable cursors result tables, and so on.

Acceptable values:	0 to 2147483647
Default:	0
Update:	option 10 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAXTEMPS

An entry of 0 means that no limit is enforced.

By default, the specified value for MAXTEMPS is in MB, and in job DSNTIJUZ, the parameter value must be specified in MB. However, you can enter a GB value in the MAX TEMP STG/AGENT field of the installation CLIST by specifying a suffix of G, for example `4G`.

SEPARATE WORK FILES field (WFDBSEP subsystem parameter)

The WFDBSEP subsystem parameter specifies whether DB2 should provide an unconditional separation of table spaces in the work file database based on the allocation attributes of the table spaces.

Acceptable values:	YES, NO
Default:	NO
Update:	option 10 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM.WFDBSEP

YES DB2 always directs declared global temporary table work only to

DB2-managed (STOGROUP) work file table spaces that are defined with a non-zero SECQTY setting. DB2 always directs work file work only to other work file table spaces (DB2-managed table spaces that are defined with a zero SECQTY value or user-managed table spaces). If no table space with the preferred allocation type is available, DB2 issues an error message, a negative SQLCODE, or both.

NO DB2 attempts to direct declared global temporary table work to DB2-managed (STOGROUP) work file table spaces that are defined with a non-zero SECQTY value. DB2 attempts to direct work file work to any other work file table space (DB2-managed table spaces that are defined with a zero SECQTY value or user-managed table spaces). If no table space with the preferred allocation type is available, DB2 selects a table space with a non-preferred allocation type.

MAX TEMP RID field (MAXTEMPS_RID subsystem parameter)

The MAXTEMPS_RID subsystem parameter determines the maximum amount of temporary storage in the work file database that a single RID list can use at a time.

The work file storage is used for the RID list when the RID pool storage cannot be used to contain all the RIDs. In other words, when RID pool storage overflow occurs for a RID list, the DB2 subsystem attempts to store the RID list in work file storage instead of falling back to a relational scan. This subsystem parameter specifies the maximum number of RIDs (measured in RID blocks) that the subsystem is allowed to store in the work file. If the maximum number of RIDs is exceeded, the subsystem falls back to a relational scan.

The MAXTEMPS_RID subsystem parameter does not affect RID list processing for pair-wise join.

Acceptable values:	NONE, NOLIMIT, or 1 to 329166
Default:	NOLIMIT
Update:	option 10 on panel DSNTIPB
DSNZP*xxx*:	MAXTEMPS_RID

The MAXTEMPS_RID value is specified in number of RID blocks. An entry of NONE means that the RID list processing will not use work file storage. An entry of NOLIMIT means that the number of RID blocks that can be stored in a work file is unrestricted. This is the default behavior.

Each RID block that is stored in the work file occupies 32 KB of work file storage and contains 6524 RIDs. For example, a value of 10000 for MAXTEMPS_RID limits the number of RIDs that are allowed to be stored in the work file to 6,524,000, or approximately 312.5 MB.

Another subsystem parameter, MAXTEMPS, controls the maximum amount of temporary storage in the work file database that a single agent can use at a time for any type of usage. The scope of the parameter includes work file usage for storing RIDs. Therefore, if MAXTEMPS is set to a non-zero value, it overrides the setting of MAXTEMPS_RID in some cases.

For example, consider the following scenario:

MAXTEMPS is set to 2G, meaning that 2 GB of work file storage is allowed to be used by the agent. MAXTEMPS_RID is set to 20000, meaning that 20,000 RID

blocks are allowed to be stored in a work file, which translates to 625 MB of work file storage. Suppose that a RID list processing encountered a RID pool overflow, and therefore, the DB2 subsystem is using work file storage to store the RID list. Suppose that 104,384,000 RIDs are fetched into the RID list. The list therefore, takes up 16,000 RID blocks, which is below the MAXTEMPS_RID value of 20000. If no other activity is using much work file storage in the agent, the RID list should be successfully stored in a work file, and RID processing could continue. However, if other activities in the agent are using more than 1.5 GB of work file storage, less than 500 MB of work file storage is available to the agent (MAXTEMPS - 1.5 GB). The 16,000 RID blocks cannot fit in the 500 MB of available space. RID processing will not have enough work file storage and will fall back to a relational scan although the MAXTEMPS_RID value is well above the amount that the RID list needs.

Thread management panel: DSNTIPE

The entries on the DSNTIPE panel determine main storage sizes.

Updating the parameters: You can use the UPDATE mode of the CLIST to update any value on this panel.

```
DSNTIPE         INSTALL DB2 - THREAD MANAGEMENT
===> _

Check numbers and reenter to change:

 1  DATABASES              ===> 200        Concurrently in use
 2  MAX USERS              ===> 200        Concurrently running in DB2
 3  MAX REMOTE ACTIVE      ===> 200        Maximum number of active
                                               database access threads
 4  MAX REMOTE CONNECTED   ===> 10000      Maximum number of remote DDF
                                               connections that are supported
 5  MAX TSO CONNECT        ===> 50         Users on QMF or in DSN command
 6  MAX BATCH CONNECT      ===> 50         Users in DSN command or utilities
 7  MAX KEPT DYN STMTS     ===> 5000       Maximum number of prepared dynamic
                                           statements saved past commit points
 8  CONTRACT THREAD STG    ===> YES        Periodically free unused thread stg
 9  MANAGE THREAD STORAGE  ===> NO         Manage thread stg to minimize size
10  LONG-RUNNING READER    ===> 10         Minutes before read claim warning
11  PAD INDEXES BY DEFAULT ===> NO         Use PADDED for new indexes
12  MAX OPEN FILE REFS     ===> 100        Maximum concurrent open data sets

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 27. Thread management panel: DSNTIPE

DATABASES field for panel DSNTIPE

The value of the DATABASES field of panel DSNTIPE indicates the maximum number of databases that are to be open at one time.

Acceptable values:	1 to 800
Default:	200
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	none

The maximum number of concurrent databases is affected primarily by the DSMAX field on panel DSNTIPC, which specifies the maximum number of open data sets.

Related tasks:

Programming applications for performance (DB2 Performance)

Related reference:

"DSMAX field (DSMAX subsystem parameter)" on page 491

MAX USERS field (CTHREAD subsystem parameter)

The CTHREAD subsystem parameter controls the maximum number of allied threads that are to be allocated concurrently. *Allied threads* are threads that are started at the local subsystem.

Acceptable values:	1 to 20000
Default:	200
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CTHREAD

Count each of the following items as separate users:

- Each TSO user (whether running a DSN command or a DB2 request from DB2 QMF)
- Each batch job (whether running a DSN command or a DB2 utility)
- Each IMS region that can access DB2
- Each active CICS transaction that can access DB2
- Each utility (each utility uses one thread, plus one thread for each subtask)
- Each connection from users of CAF and RRSAF
- Each WebSphere application server or other application server that uses JDBC or ODBC has multiple connections to DB2. The number of connections is determined by the application server.

The maximum number of threads accessing data that can be allocated concurrently is the sum of the value of the MAX USERS field and the MAX REMOTE ACTIVE field on panel DSNTIPE. The maximum allowable value for this sum is 20000. When the number of users who are attempting to access DB2 exceeds the number that you specify, excess plan allocation requests are queued. In most situations, the amount of real storage determines the maximum number of threads that DB2 can handle.

Due to parallelism, DB2 utilities each use a minimum of one thread, plus an additional thread for each subtask. Therefore, a single utility might use many threads. Specify a thread value accordingly to accommodate parallelism within utilities. Consider using a value that is higher than the default value or the value that you specified in a previous version of DB2.

MAX REMOTE ACTIVE field (MAXDBAT subsystem parameter)

The value of the MAXDBAT subsystem parameter specifies the maximum number of database access threads (DBATs) that are allowed to be concurrently active.

Acceptable values:	0 to 19999
Default:	200
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MAXDBAT

The total number of threads accessing data concurrently is the sum of the value of the MAX USERS field and this field, MAX REMOTE ACTIVE. The maximum allowable value for this sum is 20000. If a request for a new connection to DB2 is received and MAX REMOTE ACTIVE has been reached, the resulting action depends on whether ACTIVE or INACTIVE is specified for the DDF THREADS field on panel DSNTIPR.

If DDF THREADS is ...	Action taken is ...
ACTIVE	The allocation request is allowed but any further processing for the connection is queued waiting for an active database access thread to terminate.
INACTIVE	The allocation request is allowed and is processed when DB2 can assign an unused database access thread slot to the connection.

Setting MAX REMOTE ACTIVE to 0

You can use a 0 in this field to restrict DDF server activity on a member of a data sharing group. A setting of 0 produces the following results:

- DDF does not register the member with WLM for member-specific Sysplex routing. This does not prevent the member from using WLM for enclave prioritization, but it prevents WLM from including this member in the Sysplex routing data that is sent to remote sites.
- In an SNA network, DDF does not register the member's LU name with the VTAM generic LU name during DDF startup. This causes VTAM generic resource connections to be directed to DB2 members that specify a MAX REMOTE ACTIVE value of greater than 0.
- In a TCP/IP network, DDF does not listen on the DRDA SQL port. This means TCP/IP SQL connections can be accepted only by members that specify MAX REMOTE ACTIVE greater than 0.
- DDF rejects requests for the Sysplex Routing TPN with the following sense code: SNA TPN not available.
- MAX REMOTE CONNECTED (CONDBAT) will be lowered to zero by default if it is not set explicitly.
- DB2 will still accept inbound Automatic Resynchronization requests.

MAX REMOTE CONNECTED field (CONDBAT subsystem parameter)

The CONDBAT subsystem parameter sets the maximum number of concurrent inbound DDF connections that are to be allowed.

Acceptable values:	0 to 150000
Default:	10000
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CONDBAT

This value must be greater than or equal to the value of MAX REMOTE ACTIVE. If MAX REMOTE ACTIVE is set to zero, MAX REMOTE CONNECTED is also set to zero. If a request to allocate a new connection to DB2 is received, and MAX REMOTE CONNECTED has been reached or MAX REMOTE CONNECTED is zero, the connection request is rejected.

Related tasks:

Setting thread limits for database access threads (DB2 Performance)

MAX TSO CONNECT field (IDFORE subsystem parameter)

The IDFORE subsystem parameter determines the maximum number of users that can be identified to DB2 from TSO foreground at the same time.

Acceptable values:	1 to 20000
Default:	50
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDFORE

Count each of the following items as a separate user:

- Each TSO foreground user that is executing a DSN command.
- Each TSO foreground user that is connected to DB2 through the CAF or RRSAF. This can include DB2 QMF users who are running in TSO foreground or user-written CAF or RRSAF applications running in TSO foreground.

When the number of TSO users who are attempting to access DB2 exceeds the number you specify, excess connection requests are rejected. No DB2 subsystem parameter controls the maximum concurrent connections for IMS and CICS. You can control those limits by using IMS and CICS facilities. For the CICS attachment, the maximum number of connections to DB2 by using the resource control table (RCT) TYPE=INIT THRDMAX value.

MAX BATCH CONNECT field (IDBACK subsystem parameter)

The IDBACK subsystem parameter determines the maximum number of concurrent connections that can be identified to DB2 from batch.

Acceptable values:	1 to 20000
Default:	50
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDBACK

Count each of the following items as a separate connection:

- Each DB2 utility.
- Each batch job that uses DB2 QMF.
- Each batch job that uses the DSN command processor.
- Each task that is connected to DB2 through the call attachment facility, which runs in batch. Among others, this can include:
  - Batch jobs that use DB2 QMF.
  - TCP/IP FTP connections.
- Each RRSAF connection that runs in batch.
- Each WebSphere application server or other application server that uses JDBC or ODBC has multiple connections to DB2. The number of connections is determined by the application server.

Batch job requests to access DB2 that exceed this limit are rejected.

REBUILD INDEX processing uses DB2 connections and might cause message DSNU397I to be issued. If you receive message DSNU397I indicating the REBUILD INDEX utility is constrained, increase the number of concurrent connections.

MAX KEPT DYN STMTS field (MAXKEEPD subsystem parameter)

The MAXKEEPD subsystem parameter specifies the maximum number of prepared, dynamic SQL statements that are to be saved past a commit point. This parameter applies to applications that run with the KEEPDYNAMIC(YES) bind option.

Acceptable values:	0 to 65535
Default:	5000
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAXKEEPD

This is a system-wide limit. This parameter does not limit the size of the dynamic cache itself.

When many applications that are bound with KEEPDYNAMIC(YES) run in a system that has the dynamic statement cache active, they can use a considerable amount of storage in the DBM1 address space. This parameter helps limit the amount of storage that these applications use by limiting the total number of prepared statements held by these applications past a commit point. If this limit is exceeded, DB2 honors the KEEPDYNAMIC(YES) behavior, but "implicit" prepares might be necessary to rebuild the executable version of some SQL statements when they are executed after a commit.

If you enter 0, DB2 cannot keep the executable version of dynamic SQL statements past commit points. To retain the KEEPDYNAMIC(YES) behavior after a commit point, DB2 implicitly prepares dynamic SQL statements to rebuild the executable version.

Related tasks:

Keeping prepared statements after commit points (DB2 Application programming and SQL)

Related reference:

BIND and REBIND options (DB2 Commands)

CONTRACT THREAD STG field (CONTSTOR subsystem parameter)

The CONTSTOR parameter controls whether DB2 is to periodically contract each thread's working storage area. If YES is used, DB2 examines threads at commit points and periodically returns storage that is no longer in use to the operating system. Storage that a thread acquires is normally allocated to that thread until deallocation.

Acceptable values:	YES, NO
Default:	YES
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CONTSTOR

Recommendation: For subsystems that have many long-running threads and that are constrained on storage in the DBM1 address space, specifying YES for this parameter can reduce the total amount of storage that is used in the DBM1 address space. For a subsystem that does not have storage constraints, specify NO for this parameter to achieve optimal performance. If you specify NO, monitor storage usage to ensure that no storage shortage occurs.

MANAGE THREAD STORAGE field (MINSTOR subsystem parameter)

The MINSTOR subsystem parameter controls whether DB2 is to use storage management algorithms that minimize the amount of working storage that is consumed by individual threads.

Acceptable values:	YES, NO
Default:	NO
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MINSTOR

LONG-RUNNING READER field (LRDRTHLD subsystem parameter)

The LRDRTHLD subsystem parameter specifies the number of minutes that an agent is to be allowed to hold a read claim. If a read claim is held longer than allowed, DB2 issues a warning message and writes a trace record to report it as a long-running reader.

Acceptable values:	0 to 1439 minutes
Default:	10
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM LRDRTHLD

If you specify a value of 0, DB2 will not report long-running readers.

PAD INDEXES BY DEFAULT field (PADIX subsystem parameter)

The PADIX subsystem parameter determines whether new indexes are to be padded by default.

Acceptable values:	NO, YES
Default:	NO
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PADIX

NO New indexes will not be padded unless the PADDED option is specified on the CREATE INDEX statement.

This is the default value.

YES New indexes will be padded unless the NOT PADDED option is specified on the CREATE INDEX statement.

This parameter affects only indexes that have at least one varying-length column.

Related concepts:

Indexes that are padded or not padded (Introduction to DB2 for z/OS)

MAX OPEN FILE REFS field (MAXOFILR subsystem parameter)

The MAXOFILR subsystem parameter specifies the maximum number of data sets that can be open concurrently for the processing of LOB file references.

Acceptable values:	0 to value specified in MAX USERS field
Default:	100
Update:	option 11 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MAXOFILR

Buffer pool sizes panel 1: DSNTIP1

The DSNTIP1 panel is the first of two panels on which you can specify the size of your virtual buffer pools.

Updating the buffer pool sizes: You can change your buffer pool sizes online with the ALTER BUFFERPOOL command, but you cannot change these sizes by running the DSNTINST CLIST in update mode.

```
 DSNTIP1         INSTALL DB2 - BUFFER POOL SIZES - PANEL 1
===> _


 1 DEFAULT  4-KB BUFFER POOL FOR USER DATA ===> BP0       BP0    - BP49
 2 DEFAULT  8-KB BUFFER POOL FOR USER DATA ===> BP8K0     BP8K0  - BP8K9
 3 DEFAULT 16-KB BUFFER POOL FOR USER DATA ===> BP16K0    BP16K0 - BP16K9
 4 DEFAULT 32-KB BUFFER POOL FOR USER DATA ===> BP32K     BP32K  - BP32K9
 5 DEFAULT BUFFER POOL FOR USER LOB DATA   ===> BP0       BP0    - BP32K9
 6 DEFAULT BUFFER POOL FOR USER XML DATA   ===> BP16K0    BP16K0 - BP16K9
 7 DEFAULT BUFFER POOL FOR USER INDEXES    ===> BP0       BP0    - BP32K9
Enter buffer pool sizes in number of pages.
 8 BP0  ==> 20000        18 BP10 ==> 0            28 BP20 ==> 0
 9 BP1  ==> 0            19 BP11 ==> 0            29 BP21 ==> 0
10 BP2  ==> 0            20 BP12 ==> 0            30 BP22 ==> 0
11 BP3  ==> 0            21 BP13 ==> 0            31 BP23 ==> 0
12 BP4  ==> 0            22 BP14 ==> 0            32 BP24 ==> 0
13 BP5  ==> 0            23 BP15 ==> 0            33 BP25 ==> 0
14 BP6  ==> 0            24 BP16 ==> 0            34 BP26 ==> 0
15 BP7  ==> 0            25 BP17 ==> 0            35 BP27 ==> 0
16 BP8  ==> 0            26 BP18 ==> 0            36 BP28 ==> 0
17 BP9  ==> 0            27 BP19 ==> 0            37 BP29 ==> 0

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 28. Buffer pool sizes panel 1: DSNTIP1

Related tasks:

Monitoring and tuning buffer pools using online commands (DB2 Performance)

Related reference:

"Buffer pool sizes panel 2: DSNTIP2" on page 342

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

DEFAULT 4-KB BUFFER POOL FOR USER DATA field (TBSBPOOL subsystem parameter)

The TBSBPOOL subsystem parameter specifies the default buffer pool to use for 4 KB page size table spaces that are created implicitly.

Acceptable values:	Any 4 KB buffer pool names
Default:	BP0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBPOOL

For a CREATE DATABASE statement, if the user does not specify the buffer pool name, the default name is the TBSBPOOL value. For a CREATE TABLESPACE statement, if the user does not specify a buffer pool name, the TBSBPOOL value is not used, and the default name is determined from the database. For a CREATE TABLE statement, if the user does not specify a table space name, DB2 implicitly creates a table space and uses the buffer pool name from the value of TBSBPOOL for a 4 KB page size table space.

Related concepts:

How SQL statements are invoked (DB2 SQL)

DEFAULT 8-KB BUFFER POOL FOR USER DATA field (TBSBP8K subsystem parameter)

The TBSBP8K subsystem parameter specifies the default buffer pool to use for 8 KB page size table spaces that are created implicitly.

Acceptable values:	Any 8 KB buffer pool names
Default:	BP8K0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBP8K

For a CREATE TABLESPACE statement, if the user does not specify a buffer pool name, the TBSBP8K value is not used, and the default name is determined from the database. For a CREATE TABLE statement, if the user does not specify a table space name, DB2 implicitly creates a table space and uses the buffer pool name from the TBSBP8K value for a 8 KB page size table space.

DEFAULT 16-KB BUFFER POOL FOR USER DATA field (TBSBP16K subsystem parameter)

The TBSBP16K subsystem parameter specifies the default buffer pool to use for 16 KB page size table spaces that are created implicitly.

Acceptable values:	Any 16 KB buffer pool names
Default:	BP16K0

Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBP16K

For a CREATE TABLESPACE statement, if the user does not specify a buffer pool name, the TBSBP16K value is not used, and the default name is determined from the database. For a CREATE TABLE statement, if the user does not specify a table space name, DB2 implicitly creates a table space and uses the buffer pool name from the TBSBP16K value for a 16 KB page size table space.

DEFAULT 32-KB BUFFER POOL FOR USER DATA field (TBSBP32K subsystem parameter)

The TBSBP32K subsystem parameter specifies the default buffer pool to use for 32 KB page size table spaces that are created implicitly.

Acceptable values:	Any 32 KB buffer pool names
Default:	BP32K
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBP32K

For a CREATE TABLESPACE statement, if the user does not specify a buffer pool name, the TBSBP32K value is not used, and the default name is determined from the database. For a CREATE TABLE statement, if the user does not specify a table space name, DB2 implicitly creates a table space and uses the buffer pool name from the TBSBP32K value for a 32 KB page size table space.

DEFAULT BUFFER POOL FOR USER LOB DATA field (TBSBPLOB subsystem parameter)

The TBSBPLOB subsystem parameter specifies the default buffer pool that is to be used for LOB table spaces that are created implicitly. This parameter also applies to LOB tables spaces that are created explicitly without the BUFFERPOOL clause.

Acceptable values:	Any 4 KB, 8 KB, 16 KB, or 32 KB buffer pool names
Default:	BP0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBPLOB

USE privilege is required on the specified buffer pool to use it by default.

Related concepts:

How SQL statements are invoked (DB2 SQL)

DEFAULT BUFFER POOL FOR USER XML DATA field (TBSBPXML subsystem parameter)

The TBSBPXML subsystem parameter specifies the default buffer pool that is to be used for XML table spaces that are created implicitly.

Acceptable values:	Any 16 KB buffer pool name
Default:	BP16K0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TBSBPXML

USE privilege is required on the specified buffer pool to use it by default.

Related concepts:

How SQL statements are invoked (DB2 SQL)

DEFAULT BUFFER POOL FOR USER INDEXES field (IDXBPOOL subsystem parameter)

The IDXBPOOL subsystem parameter identifies the default buffer pool that is to be used for indexes on user data.

DB2 uses the IDXBPOOL value if a value for INDEXBP is not specified on a CREATE DATABASE statement. DB2 does not use this value for a CREATE INDEX statement without the BUFFERPOOL option. In that case, DB2 uses the default index buffer pool for the database.

Acceptable values:	Any 4-KB, 8-KB, 16-KB, or 32-KB buffer pool names
Default:	BP0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDXBPOOL

BUFFERPOOL fields BP0 - BP29

The values of fields BP0 through BP29 on panel DSNTIP1 specify the number of 4 KB buffers in each of those virtual buffer pools (BP0 through BP29).

Acceptable values:	for BP0, 2000 to 250 000 000; for BP1 through BP29, 0 to 250 000 000
Default:	for BP0, 20000; for BP1 through BP29, 0
Update:	ALTER BUFFERPOOL command
DSNZP*xxx*:	none

Restriction: The sum of the storage that is available in all buffer pools cannot exceed 1 TB. The amount of storage configured on the subsystem can further limit the buffer pool sizes. If you specify more storage than the total real storage that is available, performance can suffer.

Buffer pool sizes panel 2: DSNTIP2

The DSNTIP2 panel is the second of the two panels that enables you to specify the size of your virtual buffer pools.

The total size of all buffer pools cannot exceed 1 TB.

Updating the buffer pool sizes: You can change your buffer pool sizes online with the ALTER BUFFERPOOL command. You cannot change these sizes by running the DSNTINST CLIST in update mode.

```
 DSNTIP2         INSTALL DB2 - BUFFER POOL SIZES - PANEL 2
===> _

Enter buffer pool sizes in number of pages.
 1 BP30  ==> 0          18 BP47   ==> 0          35 BP16K4 ==> 0
 2 BP31  ==> 0          19 BP48   ==> 0          36 BP16K5 ==> 0
 3 BP32  ==> 0          20 BP49   ==> 0          37 BP16K6 ==> 0
 4 BP33  ==> 0          21 BP8K0  ==> 2000       38 BP16K7 ==> 0
 5 BP34  ==> 0          22 BP8K1  ==> 0          39 BP16K8 ==> 0
 6 BP35  ==> 0          23 BP8K2  ==> 0          40 BP16K9 ==> 0
 7 BP36  ==> 0          24 BP8K3  ==> 0          41 BP32K  ==> 250
 8 BP37  ==> 0          25 BP8K4  ==> 0          42 BP32K1 ==> 0
 9 BP38  ==> 0          26 BP8K5  ==> 0          43 BP32K2 ==> 0
10 BP39  ==> 0          27 BP8K6  ==> 0          44 BP32K3 ==> 0
11 BP40  ==> 0          28 BP8K7  ==> 0          45 BP32K4 ==> 0
12 BP41  ==> 0          29 BP8K8  ==> 0          46 BP32K5 ==> 0
13 BP42  ==> 0          30 BP8K9  ==> 0          47 BP32K6 ==> 0
14 BP43  ==> 0          31 BP16K0 ==> 500        48 BP32K7 ==> 0
15 BP44  ==> 0          32 BP16K1 ==> 0          49 BP32K8 ==> 0
16 BP45  ==> 0          33 BP16K2 ==> 0          50 BP32K9 ==> 0
17 BP46  ==> 0          34 BP16K3 ==> 0

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 29. Buffer pool sizes panel 2: DSNTIP2

Related tasks:

Monitoring and tuning buffer pools using online commands (DB2 Performance)

Related reference:

"Buffer pool sizes panel 1: DSNTIP1" on page 339

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

BUFFERPOOL fields BP30 - BP32K9

The values of fields BP30 through BP32K9 on panel DSNTIP2 specify the number of buffers in each of those virtual buffer pools (BP30 through BP32K9).

Acceptable values:	for BP30 to BP49, 0 to 250 000 000; for BP8K0 to BP8K9, 0 to 125 000 000; for BP16K0 to BP16K9, 0 to 62 500 000; for BP32K, 250 to 31 250 000; for BP32K1 to BP32K9, 0 to 31 250 000
Default:	for BP8K0, 2000; for BP16K0, 500; for BP32K, 250; for all others, 0
Update:	ALTER BUFFERPOOL command
DSNZP*xxx*:	none

Restriction: The sum of the storage that is available in all buffer pools cannot exceed 1 TB. The amount of storage that is configured on the subsystem can further limit the buffer pool sizes.

Tracing parameters panel: DSNTIPN

The entries on the DSNTIPN panel affect the audit, global, accounting, and monitor traces and the checkpoint frequency.

Because space on this panel is limited, some input fields are not wide enough to display the entire contents of the field at one time. These fields are defined as ISPF scrollable fields, which let you see the field contents. A less-than symbol (<) or a greater-than (>) symbol denotes a scrollable field. A > symbol indicates that the

field can be scrolled to the right, and a < symbol indicates that the field can be scrolled to the left. You can position your cursor in the field and use ISPF scroll commands LEFT and RIGHT to scroll through the field. To see the entire contents of the field at one time, type EXPAND in the command field, position your cursor in the scrollable field, and press Enter. (Alternatively, you can assign your PF keys to be the LEFT, RIGHT, and EXPAND commands.)

```
DSNTIPN         INSTALL DB2 - TRACING PARAMETERS

===> _
Enter data below:

 1  AUDIT TRACE        ===> NO       > Audit classes to start. NO,YES,list
 2  TRACE AUTO START   ===> NO       > Global classes to start. YES,NO,list
 3  TRACE SIZE         ===> 64K        Trace table size in bytes. 4K-396K
 4  SMF ACCOUNTING     ===> 1        > Accounting classes to start. NO,YES,list
 5  SMF STATISTICS     ===> YES      > Statistics classes to start. NO,YES,list
 6  STATISTICS TIME    ===> 1          Time interval in minutes. 1-60
 7  STATISTICS SYNC    ===> NO         Synchronization within the hour. NO,0-59
 8  DATASET STATS TIME ===> 5          Time interval in minutes. 1-60
 9  MONITOR TRACE      ===> NO       > Monitor classes to start. NO,YES,list
10  MONITOR SIZE       ===> 1M         Default monitor buffer size. 1M-64M
11  UNICODE IFCIDS     ===> NO         Include UNICODE data when writing IFCIDS
12  DDF/RRSAF ACCUM    ===> 10         Rollup accting for DDF/RRSAF. NO, 2-64K
13  AGGREGATION FIELDS ===> 0          Rollup accting aggregation fields
14  COMPRESS SMF RECS  ===> OFF        Compress trace records destined for SMF

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 30. Tracing panel: DSNTIPN

Related concepts:

DB2 audit trace (Managing Security)

AUDIT TRACE field (AUDITST subsystem parameter)

The AUDITST parameter controls whether the audit trace is to start automatically when DB2 is started. You can specify the classes for which the audit trace is to start.

Acceptable values:	NO, YES, *list of classes*, an asterisk (*)
Default:	NO
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP AUDITST
Security parameter:	Yes

This is a scrollable field.

NO Specifies that there is to be no automatic start of the audit trace. If the audit trace is to be used, it must be started with the START TRACE command.

YES Specifies that the audit trace for the default class (class 1) starts whenever DB2 is started.

list of classes
Specifies that the audit trace for the specified classes starts whenever DB2 is started. To specify classes, list the numbers (any integer from 1 to 32) separated by commas.

an asterisk (*)
Specifies that the audit trace starts for all classes whenever DB2 is started.

Note: This is a security-related parameter. Audit traces should be enabled to collect information about DB2 security controls to ensure that data access is allowed only for authorized users.

Recommendation: Either AUDITST or audit policies should be enabled at DB2 startup.

Related concepts:

DB2 audit trace (Managing Security)

TRACE AUTO START field (TRACSTR subsystem parameter)

The TRACSTR subsystem parameter specifies whether the global trace is to start automatically when DB2 is started. This parameter also specifies the classes for which the global trace is to automatically start.

Acceptable values:	YES, NO, *list of classes*, an asterisk (*)
Default:	NO
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TRACSTR

This is a scrollable field.

YES Starts the global trace for the default classes (classes 1, 2, and 3) whenever DB2 is started, and it performs additional data consistency checks whenever a data or index is modified.

NO Specifies no automatic start of the global trace. If the global trace is to be used, it must be started with a special START TRACE command.

list of classes
Starts the global trace for the specified classes. Enter a list of class numbers (any integer from 1 to 32) separated by commas. Only classes 1 to 9 are defined by DB2.

asterisk (*)
Starts global trace for all classes.

The global trace is used to diagnose problems in DB2. Users with production systems that require high performance might consider turning off global trace. However, be aware that turning off global trace presents a serviceability exposure. In the event of a system failure, IBM Software Support might request that you turn on global trace and attempt to re-create the problem.

Related concepts:

DB2 audit trace (Managing Security)

Related tasks:

Minimizing the use of DB2 traces (DB2 Performance)

TRACE SIZE field (TRACTBL subsystem parameter)

The TRACTBL subsystem parameter specifies the size, in bytes, of the RES trace table. This table is the default destination for the global trace records in DB2. Most trace records require 32 byte entries. Events with more than three data items require 64 byte entries.

Acceptable values:	4K to 396K
Default:	64K
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TRACTBL

You can use the abbreviation K for multiples of 1024 bytes. The actual value is rounded up to a multiple of 4. If you use 50K, for example, the actual table size is 52 KB.

In the subsystem parameter, use a multiple of 4. For example, to get a 64 KB table, code TRACTBL=16.

Related concepts:

DB2 audit trace (Managing Security)

SMF ACCOUNTING field (SMFACCT subsystem parameter)

The SMFACCT subsystem parameter specifies whether DB2 is to send accounting data to SMF automatically when DB2 is started. This parameter also specifies which classes are to be sent.

Acceptable values:	NO, YES, *list of classes*, an asterisk (*)
Default:	1
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SMFACCT

This is a scrollable field.

NO Specifies no automatic start of classes.

YES Starts the trace for the default class (class 1). You might also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write the records.

list of classes
Starts the specified classes. Enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5, 7, and 8 are defined by DB2.

asterisk (*)
Starts all classes.

Related concepts:

Accounting trace (DB2 Performance)

Related tasks:

"Installation step 8: Record DB2 data to SMF (optional)" on page 519

SMF STATISTICS field (SMFSTAT subsystem parameter)

The SMFSTAT subsystem parameter specifies whether DB2 is to send statistical data to SMF automatically when DB2 is started. This parameter also specifies which classes are sent.

Acceptable values:	NO, YES, *list of classes*, an asterisk (*)
Default:	YES
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SMFSTAT

This is a scrollable field.

NO Specifies no automatic start.

YES Starts the trace for the default classes (classes 1, 3, 4, 5, and 6). You might also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write the records.

list of classes
Starts the specified classes. Enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5 are defined by DB2.

asterisk (*)
Starts all classes.

Related tasks:

"Installation step 8: Record DB2 data to SMF (optional)" on page 519

Monitoring concurrency and locks (DB2 Performance)

STATISTICS TIME field (STATIME subsystem parameter)

The STATIME subsystem parameter specifies the time interval, in minutes, between statistics collections. Statistics records are to be written at approximately the end of this interval.

In DB2 Version 10, the STATIME subsystem parameter applies only to IFCIDs 0105, 0106, 0199, and 0365. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME, and the corresponding trace records are written at fixed, one-minute intervals.

Acceptable values:	1 to 60
Default:	1
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STATIME

STATISTICS SYNC field (SYNCVAL subsystem parameter)

The SYNCVAL subsystem parameter specifies whether DB2 statistics recording is to be synchronized with some part of the hour, such as 15 minutes past the hour.

In DB2 Version 10, the SYNCVAL subsystem parameter applies only to IFCIDs 0105, 0106, and 0199. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by SYNCVAL, and the corresponding trace records are written at fixed, one-minute intervals.

Acceptable values:	NO, 0 to 59
Default:	NO
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SYNCVAL

NO Specifies that statistics recording is not synchronized. NO is the default.

0 to 59
DB2 statistics recording interval is synchronized with the beginning of the hour (0 minutes past the hour) or with any number of minutes past the hour up to 59. This parameter has no effect if STATIME is greater than 60.

Example

You want the DB2 statistics recording interval to have a length of 15 minutes and to be synchronized with 15 minutes past the hour. Thus, DB2 statistics are recorded at 15, 30, 45, and 60 minutes past the hour. To establish this interval, you would specify the following:

```
STATIME=15
SYNCVAL=15
```

DATASET STATS TIME field (DSSTIME subsystem parameter)

The DSSTIME subsystem parameter specifies the time interval, in minutes, that is to occur between the resetting of data set statistics for online performance monitors.

Acceptable values:	1 to 60
Default:	5
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DSSTIME

Online performance monitors can request DB2 data set statistics for the current interval with an IFI READS request for IFCID 0199.

MONITOR TRACE field (MON subsystem parameter)

The MON subsystem parameter specifies whether the monitor trace is to start automatically when DB2 is started. The trace can be started for all classes, a default class, or a specified list of classes.

Acceptable values:	NO, YES, *list of classes*, an asterisk (*)
Default:	NO
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MON

This is a scrollable field.

NO Specifies no automatic start.

YES Starts the trace for the default classes (class 1) whenever DB2 is started.

list of classes
: Start the trace automatically for the specified classes. Enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 8 are defined by DB2.

asterisk (*)
: Starts the trace automatically for all classes.

Related concepts:

Monitor trace (DB2 Performance)

MONITOR SIZE field (MONSIZE subsystem parameter)

The MONSIZE subsystem parameter determines the default buffer size that is to be used for monitor traces when data is sent to monitor destinations.

Acceptable values:	1048576 to 67108864
Default:	1048576

Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MONSIZE

You can enter the value in bytes (for example, 4194304) or use the abbreviation K for kilobytes (for example, 4096K) or M for megabytes (for example, 4M).

UNICODE IFCIDS field (UIFCIDS subsystem parameter)

The UIFCIDS subsystem parameter specifies whether output from IFC records is to include Unicode information.

Acceptable values:	NO, YES
Default:	NO
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP UIFCIDS

Only a subset of the character fields are encoded in Unicode. These fields are identified in the IFCID record definition by a `%U` in the comment area to the right of the field declaration in the DSNDQWxx copy files. The remaining fields maintain the same encoding of previous releases.

DDF/RRSAF ACCUM field (ACCUMACC subsystem parameter)

The ACCUMACC parameter determines whether DB2 accounting data is to be accumulated by the user for DDF and RRSAF threads.

Acceptable values:	NO, 2 to 65535
Default:	10
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ACCUMACC

NO DB2 writes an accounting record when a DDF thread is made inactive or when signon occurs for an RRSAF thread.

value between 2 and 65535
DB2 writes an accounting record every *n* accounting intervals for a given user, where *n* is the number that you specify for DDF/RRSAF ACCUM.

A user is identified by a concatenation of values, which is specified in the AGGREGATION FIELDS field. These values can be set by DDF threads through Server Connect and Set Client calls, and by RRSAF threads through the RRSAF SIGN, AUTH SIGNON, and CONTEXT SIGNON functions.

If you specify a value between 2 and 65535 for DDF/RRSAF ACCUM, an accounting record might be written prior to the *n*th accounting interval for a given user in the following cases:

- An internal storage threshold is reached for the accounting rollup blocks.
- The staleness threshold is reached. The end user has not rolled data into an internal block in approximately 10 minutes and DB2 considers the data stale.

AGGREGATION FIELDS field (ACCUMUID subsystem parameter)

The ACCUMUID subsystem parameter determines the aggregation fields that are to be used for DDF and RRSAF accounting rollup.

Acceptable values:	0 to 17
Default:	0
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ACCUMUID

The values are defined in the following table.

Table 66. Values for the aggregation fields that are used for DDF and RRSAF accounting rollup

Value	Rollup criteria	String of X'00' or string of X'40' considered for rollup?
0	End user ID, application name or transaction name, and workstation name	Yes[1]
1	End user ID	No
2	End user application name or transaction name	No
3	End user workstation name	No
4	End user ID and application name or transaction name	Yes[1]
5	End user ID and workstation name	Yes[1]
6	End user application name or transaction name and workstation name	Yes[1]
7	End user ID, application name or transaction name, and workstation name	No
8	End user ID and application name or transaction name	No
9	End user ID and workstation name	No
10	End user application name or transaction name and workstation name	No
11	End user ID, application name or transaction name, and workstation name	Yes
12	End user ID	Yes
13	End user application name or transaction name	Yes
14	End user workstation name	Yes
15	End user ID and application name or transaction name	Yes
16	End user ID and workstation name	Yes
17	End user application name or transaction name and workstation name	Yes

Note:
[1] At least one value in the set of criteria must have a value other than a string of X'00's or a string of X'40's.

For example, if a thread has the end user ID set to a string of X'00' and the workstation name set to myws, the thread qualifies for rollup if ACCUMUID is set to 5, but not if ACCUMUID is set to 9.

Assume that ACCUMUID is set to 5. The threads in the following table all qualify for rollup.

Table 67. Values for end user ID and workstation name

End user ID	Workstation name
myuser	myws
myuser	A string of X'40'
A string of X'40'	myws
myuser	A string of X'00'
A string of X'00'	myws

The thread with end user ID set to myuser and workstation name set to a string of X'40' and the thread with end user ID set to myuser and workstation name set to a string or X'00' are included in the same rollup record. The thread with end user ID set to a string of X'40' and workstation name set to myws and the thread with end user ID set to a string of X'00' and workstation name set to myws are included in the same rollup record.

DB2 writes individual accounting threads for threads that do not meet the criteria for rollup.

The value of ACCUMUID is ignored if the value of ACCUMACC is set to NO.

COMPRESS SMF RECS field (SMFCOMP subsystem parameter)

The SMFCOMP subsystem parameter specifies whether DB2 is to compress trace records that are to be written to SMF.

Acceptable values:	OFF, ON
Default:	OFF
Update:	option 14 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SMFCOMP

OFF SMF trace records are not compressed.

ON Trace records that are to be written to SMF are compressed. The z/OS compression service CSRCESRV compresses everything after the SMF header (SM100END, SM101END, or SM102END). See SM100, SM101, and SM102 for compression flags and the uncompressed size of compressed data.

Operator functions panel: DSNTIPO

The entries on the DSNTIPO panel affect various operator functions. These functions include write-to-operator route codes, automatic recall, and the maximum amount of CPU time that is to be allocated for a dynamic SQL statement.

```
DSNTIPO         INSTALL DB2 - OPERATOR FUNCTIONS
===> _

Enter data below:

 1  WTO ROUTE CODES    ===> 1
                                     Routing codes for WTORs
 2  RECALL DATABASE    ===> YES      Use DFHSM automatic recall. YES or NO
 3  RECALL DELAY       ===> 120      Seconds to wait for automatic recall
 4  RLF AUTO START     ===> NO       Resource Limit Facility. NO or YES
 5  RLST NAME SUFFIX   ===> 01       Resource Limit Spec. Table (RLST)
 6  RLST ACCESS ERROR  ===> NOLIMIT  NOLIMIT, NORUN, or 1-5000000
 7  AUTO BIND          ===> YES      Use automatic bind. YES, NO, or COEXIST
 8  EXPLAIN PROCESSING ===> YES      Explain allowed on autobind? YES or NO
 9  DPROP SUPPORT      ===> 1        1=NO 2=ONLY 3=ANY
10  SITE TYPE          ===> LOCALSITE   LOCALSITE or RECOVERYSITE
11  TRACKER SITE       ===> NO       Tracker DB2 system. NO or YES
12  READ COPY2 ARCHIVE ===> NO       Read COPY2 archives first. NO or YES
13  REAL TIME STATS    ===> 30       RST time interval in minutes 1-1440
 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 31. Operator functions panel: DSNTIPO

WTO ROUTE CODES field (ROUTCDE subsystem parameter)

The ROUTCDE subsystem parameter specifies the z/OS console routing codes that are to be assigned to messages that are not solicited from a specific console.

Acceptable values:	1 to 16 route codes. Specify at least one code. Separate multiple codes by commas only, with no blanks; for example: 1,3,5,7,9,10,11.
Default:	1
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ROUTCDE

Related reference:

Routing Codes (MVS System Messages)

Routing the Message (z/OS MVS Assembler Services Guide

RECALL DATABASE field (RECALL subsystem parameter)

The RECALL subsystem parameter controls whether DFSMShsm automatic recall is to be performed for DB2 databases.

Acceptable values:	YES, NO
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RECALL

YES Indicates that DFSMShsm is invoked to automatically recall migrated table spaces.

NO Indicates that a DB2 table space that has been migrated is considered to be an unavailable resource. It must be recalled explicitly before DB2 can use it.

RECALL DELAY field (RECALLD subsystem parameter)

The RECALLD subsystem parameter specifies the maximum length of time, in seconds, that a program can be delayed for a DFSMShsm recall.

Acceptable values:	0 to 32767
Default:	120
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RECALLD

If the recall is not completed within the specified number of seconds, the program receives an error message indicating that the set is unavailable, but that a recall was initiated. If you use 0 and the RECALL DATABASE field is YES, the recall is performed asynchronously. This field is ignored if the RECALL DATABASE field is NO.

The RECALL DELAY option is not used when running a DB2 utility against a DB2-migrated data set.

RLF AUTO START field (RLF subsystem parameter)

The RLF subsystem parameter specifies whether the resource limit facility (governor) is to automatically start each time DB2 is started.

Acceptable values:	YES, NO
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLF

Related concepts:

Controlling the resource limit facility (governor) (DB2 Administration Guide)

RLST NAME SUFFIX field (RLFTBL subsystem parameter)

The RLFTBL subsystem parameter specifies the suffix that is to be used for the default resource limit specification table (RLST). The default RLST is used when the resource limit facility (governor) is automatically started or when the governor is started without a specified suffix.

Acceptable values:	any 2 alphanumeric characters; national characters are not allowed
Default:	01
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFTBL

RLST ACCESS ERROR field (RLFERR subsystem parameter)

The RLFERR subsystem parameter specifies what DB2 is to do if the governor encounters a condition that prevents it from accessing the resource limit specification table. This setting also applies if DB2 cannot find an applicable row in the resource limit specification table.

An applicable row is one that applies to the authorization ID, plan or package name, and name of the logical unit of work of the query user.

Acceptable values:	NOLIMIT, NORUN, 1 to 5000000
Default:	NOLIMIT
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFERR

NOLIMIT
Allows all dynamic SQL statements to run without limit.

NORUN
Terminates all dynamic SQL statements immediately with an SQL error code.

1 to 5000000
Specifies the default limit for the number of resource limit specification table access errors. If the limit is exceeded, the SQL statement is terminated.

Related reference:

DSNRLSTxx (DB2 Performance)

AUTO BIND field (ABIND subsystem parameter)

The ABIND subsystem parameter determines whether plans or packages are to be automatically rebound (or autobound) at run time in certain situations.

Automatic rebinds can improve availability and administration of plans and packages because you do not need to explicitly rebind some plans and packages. However, automatic rebinds can also hinder performance because they require access to the DB2 directory and catalog.

Acceptable values:	YES, NO, COEXIST
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ABIND

In a non-data sharing environment, set the ABIND subsystem parameter to YES. In a data-sharing environment, set the parameter to YES or COEXIST. These settings can trigger automatic rebinds during fallback in non-data sharing environments or when a Version 9.1 member executes a Version 10 package in data-sharing environments.

YES Allows automatic rebind operations to be performed at run time when a plan or package meets one of the following criteria:

- The package is invalid; the VALID column of SYSPACKAGE contains "N".
- The plan or package was last bound in DB2 Version 10, but it is now running on a previous version of DB2.

  This autobind is performed on the previous version of DB2 because that is where the user is attempting to run the plan or package. This autobind is performed only for the first attempt to run the plan or package on DB2 Version 8 or Version 9.1.
- The plan or package was last autobound on DB2 Version 8 or Version 9.1, but it is now running again on DB2 Version 10.

  This autobind is performed on DB2 Version 10 because that is where the user is attempting to run the plan or package. This autobind is performed only for the first attempt to run the plan or package on Version 10 after an autobind on Version 8 or Version 9.1. If the plan or package is later run again on DB2 Version 8 or Version 9.1, the autobind process starts again as previously described.

NO Prevents DB2 from performing automatic rebind operations under any circumstances. This means that when any of the following conditions are true, you must explicitly rebind the plan or package before it can be run again:

- You want to run a package that is invalid; the VALID column of SYSPACKAGE contains “N”.
- You are attempting to run a plan or package on a previous version that was last bound (explicitly or automatically) in DB2 Version 10.

If you attempt to run any plan or package in one of the situations described above, you receive SQLCODE -908 SQLSTATE 23510.

COEXIST
Allows automatic rebind operations to be performed in a DB2 data sharing environment only when the plan or package meets one of the following criteria:

- The package is invalid; the VALID column of SYSPACKAGE contains “N”.
- The package or plan was last bound on DB2 Version 10 and is now running on DB2 Version 8 or Version 9.1.

  For this case, DB2 performs an automatic rebind of the plan or package on Version 8 or Version 9.1 before running it there. This automatic rebind is performed only for the first attempt to run the plan or package on DB2 Version 8 or Version 9.1.

In a data sharing environment, an automatic rebind operation is not performed for any Version 10 plan or package that was last automatically rebound on the previous version and is now running again on Version 10. The plan or package runs on Version 10 as a Version 8-bound or Version 9.1-bound plan or package.However, no Version 10-only features will be used because the plan or package was not automatically rebound on DB2 Version 10.
The value COEXIST is relevant only for DB2 subsystems in data sharing mode. If COEXIST is specified in a non-data-sharing environment, DB2 ignores the value and uses the default value of YES when determining whether an automatic rebind can be done. YES allows a Version 8-to-Version 10 or Version 9.1-to-Version 10 automatic rebind on Version 10 after a previous Version 10-to-Version 8 or Version 10-to-Version 9.1 automatic rebind completes successfully.
To reduce the rate of automatic rebinds in this type of data sharing environment, consider specifying COEXIST.

Related concepts:

“Automatic rebinds in coexistence” on page 726

Changes that invalidate packages (DB2 Application programming and SQL)

Related tasks:

“Disallowing the automatic remigration rebind” on page 728

EXPLAIN PROCESSING field (ABEXP subsystem parameter)

The ABEXP subsystem parameter controls whether EXPLAIN processing is to occur during an automatic rebind.

Acceptable values:	YES, NO
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ABEXP

YES EXPLAIN processing occurs during the automatic rebind of a plan or package when the bind option EXPLAIN(YES) is specified. If the

PLAN_TABLE does not exist, the automatic rebind continues, but generates no EXPLAIN output. EXPLAIN processing does not occur during the automatic rebind of plans or packages with the bind option EXPLAIN(NO).

NO EXPLAIN processing does not occur during the automatic rebind of a plan or package.

DPROP SUPPORT field (EDPROP and CHGDC subsystem parameters)

The value of the DPROP SUPPORT field determines whether IMS DataPropagator is to be used to propagate SQL changes to tables defined with DATA CAPTURE CHANGES.

Acceptable values:	1, 2, 3
Default:	1
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDPROP, DSN6SPRM CHGDC

1 Specifies that you do not intend to propagate SQL changes.

2 Specifies that you intend to use IMS DataPropagator to propagate SQL changes, and that changes made to tables defined with DATA CAPTURE CHANGES are allowed **only** when all of the following conditions are met:

- Monitor trace class 6 is active.
- IMS DataPropagator is installed.
- The DB2 application is running in an IMS environment.

If you specify 2 and monitor trace class 6 is not active, IMS DataPropagator is not installed, or the IMS DataPropagator application is not running in an IMS environment, then no changes to the DB2 table are permitted.

3 Specifies that data propagation occurs when all of the following conditions are met:

- Monitor trace class 6 is active.
- IMS DataPropagator is installed.
- The DB2 application is running in an IMS environment.

The ANY option for IMS DataPropagator Support is intended for subsystems that need to propagate some data with IMS DataPropagator and need to propagate some data with a different program.

If you specify 3, an application that is not running in an IMS environment can update DB2 tables that are defined with DATA CAPTURE CHANGES. However, these changes are not propagated to IMS. You might have tables in your DB2-IMS environment that you want to be updated only by DB2 applications. You can protect these tables by using any of the following methods:

- Use the ENABLE parameter on BIND to specify a specific attachment facility through which updates to data propagation tables can be made.
- Define a validation procedure for data propagation tables to allow only certain plans to update those tables.
- Use a group authorization ID to assign update authority for data propagation tables to a group of authorization IDs that can run only in the IMS environment.

The DPROP SUPPORT field values correspond with the CHGDC and EDPROP subsystem parameters as indicated in the following table.

Table 68. Relationship between DPROP SUPPORT field and CHGDC and EDPROP subsystem parameters

DPROP SUPPORT field value	CHGDC setting	EDPROP setting
1	NO	NO
2	YES	YES
3	YES	NO

Related concepts:

IMS DataPropagator

SITE TYPE field (SITETYP subsystem parameter)

The SITETYP subsystem parameter specifies whether the current system is at a local site or a recovery site.

Acceptable values:	LOCALSITE, RECOVERYSITE
Default:	LOCALSITE
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SITETYP

LOCALSITE
Specifies that the current system is at the site where the multiple image copies are made and are operational.

RECOVERYSITE
Specifies that the current system is at the site that is named as an alternative for recovery purposes.

The RECOVER utility uses this value to determine what site the current system is on. It then recovers everything from the copies of data that are registered at that site. The RECOVER and MERGECOPY utilities use this value to determine whether COPYDDN or RECOVERDDN is allowed with NEWCOPY NO.

TRACKER SITE field (TRKRSITE subsystem parameter)

The value of the TRKRSITE subsystem parameter indicates whether the subsystem that is being installed is to be used as a remote tracker site. A remote tracker site is used by another DB2 subsystem in case of a disaster.

Acceptable values:	NO, YES
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM TRKRSITE

Related tasks:

Recovering from disasters by using a tracker site (DB2 Administration Guide)

READ COPY2 ARCHIVE field (ARC2FRST subsystem parameter)

The ARC2FRST subsystem parameter determines whether COPY2 archives are to be read first when the DB2 subsystem is started.

Acceptable values:	NO, YES
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP ARC2FRST

REAL TIME STATS field (STATSINT subsystem parameter)

The STATSINT subsystem parameter specifies the time, in minutes, that DB2 is to wait before attempting to write out page set statistics to the real-time statistics tables.

Acceptable values:	1 to 1440
Default:	30
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATSINT

Default startup modules panel: DSNTIPO3

The DSNTIPO3 panel enables you to specify the default member names of certain modules that are to be loaded at DB2 startup. These modules are the system parameter module, the application defaults module, and the exit modules.

```
DSNTIPO3     INSTALL DB2 - DEFAULT STARTUP MODULES
===> _

Enter data below:

 1  PARAMETER MODULE    ===> DSNZPARM   DB2 subsystem parameter module name
 2  APPL DEFAULTS       ===> DSNHDECP   Application defaults module name
 3  ACCESS CONTROL      ===> DSNX@XAC   Access Control Exit module name
 4  IDENTIFY/AUTH       ===> DSN3@ATH   Identify/Authorization Exit module name
 5  SIGNON              ===> DSN3@SGN   Signon Exit module name









 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 32. Default startup modules panel: DSNTIPO3

PARAMETER MODULE field

The value of the PARAMETER MODULE field specifies the member name of the load module that is to be used for DB2 subsystem parameters.

Acceptable values:	1 to 8 characters
Default:	DSNZPARM
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	none

The parameter module typically resides in library *prefix*.SDSNEXIT. To avoid conflict with members of *prefix*.SDSNLOAD, use DSNZP*xxx*, where *xxx* is any set

of three alphanumeric characters. DB2 puts this name in the startup JCL procedure in SYS1.PROCLIB, but you can override this value by using the START DB2 command.

APPL DEFAULTS field

The value of the APPL DEFAULTS field specifies the member name of the load module that is to be used for DB2 application default values.

Acceptable values:	for installation: 1 - 8 characters; for migration: DSNHDECP
Default:	DSNHDECP
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	none

This module typically resides in library *prefix*.SDSNEXIT. The application default values that are specified within this module are defined by panels DSNTIPK, DSNTIPM, DSNTIPF, and DSNTIP4. The value that is entered in this field will be assigned to the DECP parameter of the *ssnm*MSTR startup JCL procedure in SYS1.PROCLIB. The DECP parameter specifies the name of the applications defaults module that is to be loaded by default when DB2 is started. You can override this value by using the DECP option of the START DB2 command. If you use a name that starts with DSN, use DSNHDE*xx*, where *xx* is any set of two alphanumeric characters. This naming convention avoids conflict with members of *prefix*.SDSNLOAD.

If you specify a name other than DSNHDECP, the DSNTIJUZ and DSNTIJNG jobs create a module with that name, with an alias of DSNHDECP. If alias DSNHDECP already exists, it is replaced. If a module that is named DSNHDECP already exists, the link-edit request fails. In this case, you must delete the DSNHDECP module or remove the ALIAS directive for DSNHDECP from the link-edit in job step DSNTIZQ of DSNTIJUZ and DSNTIJNG.

If you run the installation CLIST in MIGRATE mode, this field is always set to DSNHDECP and cannot be modified. The following restrictions apply to the application defaults member name during migration:

- Use of an application defaults member name other than DSNHDECP requires DB2 Version 10 new-function mode and DB2 Version 10 early (ERLY) code. If these conditions are satisfied after migration, you can optionally create an application defaults member with a different name, and specify it in the DECP parameter of the DB2 startup command. You can also modify the setting of the DECP parameter of the *ssnm*MSTR startup JCL procedure to use the new name by default.
- Use of an application defaults member name other than DSNHDECP prior to Version 10 new-function mode causes DB2 startup to fail with abend 04E, reason code 00E3001C.
- Use of an application defaults member name other than DSNHDECP under pre-Version 10 ERLY code causes DB2 startup to fail with abend 04E, reason code 00E3001D.

This parameter cannot be changed online.

ACCESS CONTROL field (ACCESS_CNTL_MODULE subsystem parameter)

The value of this field specifies the member name of the load module that is to be used for the DB2 access control exit routine.

Acceptable values:	for installation: 1 - 8 characters; for migration: DSNX@XAC
Default:	DSNX@XAC
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ACCESS_CNTL_MODULE

This module typically resides in library *prefix*.SDSNEXIT. If you specify a name that starts with DSN, use DSNX@*xxx*, where *xxx* is any unused set of three alphanumeric characters. This naming convention avoids conflict with members of *prefix*.SDSNLOAD.

If you specify a name other than DSNX@XAC, the DSNTIJEX job creates a module with that name, with an alias of DSNX@XAC. If alias DSNX@XAC already exists, it is replaced. If a module that is named DSNX@XAC already exists, the link-edit request fails. In this case, you must delete the DSNX@XAC module or remove the ALIAS directive for DSNX@XAC from the link-edit in job step JEX0003 of DSNTIJEX.

If you run the installation CLIST in MIGRATE mode, this field is always set to DSNX@XAC and cannot be modified. The following restrictions apply to the member name for the access control exit routine during migration:

- Use of a member name other than DSNX@XAC for the access control exit routine requires DB2 Version 10 new-function mode and DB2 Version 10 early (ERLY) code. If these conditions are satisfied after migration, you can optionally create an access control exit member with a different name, and specify that name as the setting of the ACCESS_CNTL_MODULE subsystem parameter.
- Use of a member name other than DSNX@XAC for the access control exit routine prior to Version 10 new-function mode causes DB2 startup to fail with abend 04E, reason code 00E3001C.
- Use of a member name other than DSNX@XAC for the access control exit routine under pre-Version 10 ERLY code causes DB2 startup to fail with abend 04E, reason code 00E3001D.

This parameter cannot be changed online.

IDENTIFY/AUTH field (IDAUTH_MODULE subsystem parameter)

The value of this field specifies the member name of the load module that is to be used for the DB2 connection authorization exit routine.

Acceptable values:	for installation: 1 - 8 characters; for migration: DSN3@ATH
Default:	DSN3@ATH
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDAUTH_MODULE

This module typically resides in library *prefix*.SDSNEXIT. If you specify a name that starts with DSN, use DSN3@*xxx*, where *xxx* is any unused set of three alphanumeric characters. This naming convention avoids conflict with members of *prefix*.SDSNLOAD.

If you specify a name other than DSN3@ATH, the DSNTIJEX job creates a module with that name, with an alias of DSN3@ATH. If alias DSN3@ATH already exists, it is replaced. If a module that is named DSN3@ATH already exists, the link-edit request fails. In this case, you must delete the DSN3@ATH module or remove the ALIAS directive for DSN3@ATH from the link-edit in job step JEX0002 of DSNTIJEX.

If you run the installation CLIST in MIGRATE mode, this field is always set to DSN3@ATH and cannot be modified. The following restrictions apply to the member name for the connection authorization exit routine during migration:

- Use of a member name other than DSN3@ATH for the connection authorization exit routine requires DB2 Version 10 new-function mode and DB2 Version 10 early (ERLY) code. If these conditions are satisfied after migration, you can optionally create a connection authorization member with a different name, and specify that name as the setting of the IDAUTH_MODULE subsystem parameter.
- Use of a member name other than DSN3@ATH for the connection authorization exit routine prior to Version 10 new-function mode causes DB2 startup to fail with abend 04E, reason code 00E3001C.
- Use of a member name other than DSN3@ATH for the connection authorization exit routine under pre-Version 10 ERLY code causes DB2 startup to fail with abend 04E, reason code 00E3001D.

This parameter cannot be changed online.

SIGNON field (SIGNON_MODULE subsystem parameter)

The value of this field specifies the member name of the load module that is to be used for the DB2 sign-on exit routine.

Acceptable values:	for installation: 1 - 8 characters; for migration: DSN3@SGN
Default:	DSN3@SGN
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SIGNON_MODULE

This module typically resides in library *prefix*.SDSNEXIT. If you specify a name that starts with DSN, use DSN3@*xxx*, where *xxx* is any unused set of three alphanumeric characters. This naming convention avoids conflict with members of *prefix*.SDSNLOAD.

If you specify a name other than DSN3@SGN, the DSNTIJEX job creates a module with that name, with an alias of DSN3@SGN. If alias DSN3@SGN already exists, it is replaced. If a module that is named DSN3@SGN already exists, the link-edit request fails. In this case, you must delete the DSN3@SGN module or remove the ALIAS directive for DSN3@SGN from the link-edit in job step JEX0001 of DSNTIJEX.

If you run the installation CLIST in MIGRATE mode, this field is always set to DSN3@SGN and cannot be modified. The following restrictions apply to the member name for the sign-on exit routine during migration:

- Use of a member name other than DSN3@SGN for the sign-on exit routine requires DB2 Version 10 new-function mode and DB2 Version 10 early (ERLY) code. If these conditions are satisfied after migration, you can optionally create a connection authorization member with a different name, and specify that name as the setting of the SIGNON_MODULE subsystem parameter.
- Use of a member name other than DSN3@SGN for the sign-on exit routine prior to Version 10 new-function mode causes DB2 startup to fail with abend 04E, reason code 00E3001C.
- Use of a member name other than DSN3@SGN for the sign-on exit routine under pre-Version 10 ERLY code causes DB2 startup to fail with abend 04E, reason code 00E3001D.

This parameter cannot be changed online.

Application programming defaults panel 1: DSNTIPF

The entries on the DSNTIPF panel and the DSNTIP4 panel define application programming default values. These values are used as default values by the program preparation panels, the program preparation CLIST (DSNH), and the precompiler. These values can also be used as default values by other programs, such as DB2 QMF.

Migrating or updating the parameters: If you alter parameter values for which a change during migration or update is "not recommended," this change can invalidate the syntax of existing SQL statements or affect the way that application programs run. Update is allowed, but must be handled with caution.

Most of the values that are set here and on the DSNTIP4 panel are contained in the application defaults load module, *dsnhdecp*. The load module is located in library *prefix*.SDSNEXIT, which can be loaded and accessed by application programs. When modifying the application defaults load module, do so only by changing and running the installation CLIST.

Important: You should always use the CLIST to modify installation parameters. Do not directly modify the data in the application defaults load module. If you modify any installation parameters by changing job DSNTIJUZ directly, these values are not recorded for later updates, new installations, or migrations. In addition, these values are not checked for validity. If you do not use the CLIST to modify these parameters, DB2 may not start.

Many of the fields on this panel involve the selection of coded character set identifiers (CCSIDs). Here is some information that can help you choose values for these fields:

- If you choose YES for the MIXED DATA field, you must specify a mixed data CCSID from Table 172 on page 947 or Table 173 on page 948. An error occurs if you do not specify a CCSID or if the CCSID you specify is not listed in the table.
- If you specify an incorrect CCSID, data can become corrupted. For example, assume that the coded character set used at your site is 37, but you specify 500 as the system CCSID. If DB2 receives data with a CCSID of 500, the data can become corrupted because character conversion does not occur. Conversely, if DB2 receives data with a CCSID other than 500 and a conversion is made from that CCSID **to** 500, the data can become corrupted because character conversion occurs.

Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.

- If you need to convert to a CCSID that supports the euro symbol, you can correct it by altering the CCSID field for your default encoding scheme.
- During code conversion, DB2 first looks in the SYSSTRINGS table to see if a conversion is defined. If DB2 finds a conversion, it is used. If DB2 does not find a conversion, it uses z/OS Unicode Services. In some cases, z/OS Unicode Services is used instead of the value in SYSSTRINGS. If the conversion is not available, an error occurs.
- Converting statements to Unicode for parsing depends on having the correct input CCSID specified. The system CCSIDs must be set up correctly at installation time.

  During connect processing, a requester and server provide default CCSIDs for character data sent on the connection.

  The DB2 requester uses the application encoding scheme for its default CCSIDs. If an application provides character data that is not in the CCSID that the application encoding scheme identifies, DB2 overrides the default CCSID by tagging each field with the actual CCSID prior to sending to the server. For applications that use the Unicode encoding scheme, the DB2 requester overrides the application encoding scheme CCSIDs with the system EBCDIC CCSIDs and converts the Unicode data to EBCDIC if DB2 determines that the server does not support Unicode character data. If the server cannot accept character data in these CCSIDs, connect fails with a -332 SQLCODE.

  The DB2 server uses the system default encoding scheme to determine the default CCSID values for character data that is to be returned to the requester. For servers using the encoding scheme of Unicode, the DB2 server overrides the Unicode encoding scheme CCSIDs with the system EBCDIC CCSIDs and converts the Unicode data to EBCDIC if DB2 determines that the requester does not support Unicode character data. If the requester cannot accept character data in these CCSIDs, connect fails with a -332 SQLCODE.

```
DSNTIPF         INSTALL DB2 - APPLICATION PROGRAMMING DEFAULTS PANEL 1
===> _

Enter data below:

 1  LANGUAGE DEFAULT     ===> IBMCOB   ASM,C,CPP,IBMCOB,FORTRAN,PLI
 2  DECIMAL POINT IS     ===> .        . or ,
 3  STRING DELIMITER     ===> DEFAULT  DEFAULT, " or ' (COBOL or COB2 only)
 4  SQL STRING DELIMITER ===> DEFAULT  DEFAULT, " or '
 5  DIST SQL STR DELIMTR ===> '        ' or "
 6  MIXED DATA           ===> NO       NO or YES for mixed DBCS data
 7  EBCDIC CCSID         ===>          CCSID of SBCS or mixed data. 1-65533.
 8  ASCII CCSID          ===>          CCSID of SBCS or mixed data. 1-65533.
 9  UNICODE CCSID        ===> 1208     CCSID of UNICODE UTF-8 data.
10  DEF ENCODING SCHEME  ===> EBCDIC   EBCDIC, ASCII, or UNICODE
11  APPLICATION ENCODING ===> EBCDIC   EBCDIC, ASCII, UNICODE, ccsid (1-65533)
12  LOCALE LC_CTYPE      ===>
13  DECFLOAT ROUNDING MODE===> ROUND_HALF_EVEN

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 33. Application programming defaults panel: DSNTIPF

Related concepts:

"Euro symbol support" on page 948

Related tasks:

Processing SQL statements (DB2 Application programming and SQL)

Related reference:

"Application programming defaults panel 2: DSNTIP4" on page 372

Code set converters supplied (XL C/C++ Programming Guide)

LANGUAGE DEFAULT field (DEFLANG DECP value)

The DEFLANG DECP value identifies the default programming language that is to be used for your site.

Acceptable values:	ASM, C, CPP, FORTRAN, IBMCOB, PLI
Default:	IBMCOB
Update:	not recommended
dsnhdecp::	DEFLANG

ASM
Specifies High Level Assembler/MVS as the default programming language.

C Specifies C as the default programming language. If you specify C, you can fold SQL identifiers to uppercase. However, this is not a default from any installation panel.

CPP
Specifies C++ as the default programming language. If you specify CPP, you can fold SQL identifiers to uppercase. However, this is not a default from any installation panel.

FORTRAN
Specifies Fortran as the default programming language.

IBMCOB
Specifies Enterprise COBOL for z/OS as the default programming language.

PLI
Specifies PL/I as the default programming language.

Related concepts:

Precompiler option list format (DB2 Application programming and SQL)

DECIMAL POINT IS field (DECIMAL DECP value)

The DECIMAL DECP value determines whether the decimal point for numbers is to be a comma (,) or a period (.).

Acceptable values:	. (period) or , (comma)
Default:	. (period)
Update:	recommended only to recover an error
dsnhdecp:	DECIMAL

For example, some nations customarily signify the number "one and one-half," as 1.5; other nations use 1,5 for the same value.

This parameter is used in the following cases:

- For running dynamic SQL statements with DYNAMICRULES:

- Whether the value of field DECIMAL POINT IS is COMMA or PERIOD, DB2 recognizes the value as the decimal point for numbers.

- For binding, defining, or invoking dynamic SQL statements with DYNAMICRULES:
  - If the value of field USE FOR DYNAMICRULES is NO, DB2 does not use the value in field DECIMAL POINT IS if you specify the option COMMA or PERIOD when you precompile the application that contains the dynamic SQL statements. DB2 uses the precompiler option to determine the decimal point for numbers.
  - If the value of field USE FOR DYNAMICRULES is YES, and the value of field DECIMAL POINT IS is PERIOD or COMMA, DB2 recognizes the value as the decimal point for numbers.
- For static SQL statements in COBOL programs, DECIMAL POINT IS specifies the default precompiler option (PERIOD or COMMA).

This parameter is the default for binds at this DB2 site that are requested by a remote system that does not indicate whether the period or the comma is used to represent a decimal point. In most cases, however, requesting systems give DB2 this information.

STRING DELIMITER field (DELIM DECP value)

The DELIM DECP value determines the string delimiter that is to be used for COBOL.

Acceptable values:	DEFAULT, " (quotation mark), ' (apostrophe)
Default:	DEFAULT
Update:	not recommended
dsnhdecp:	DELIM

This option is effective for all varieties of COBOL.

DEFAULT
: The string delimiter is the quotation mark.

"
: The string delimiter is the quotation mark.

'
: The string delimiter is the apostrophe.

See the SQL STRING DELIMITER field for a description of how to use this field to get the desired set of character string delimiters for COBOL and SQL.

SQL STRING DELIMITER field (SQLDELI DECP value)

The SQLDELI DECP value specifies the character that is to be used as the SQL string delimiter to delimit character strings in dynamic SQL.

Acceptable values:	DEFAULT, " (quotation mark), ' (apostrophe)
Default:	DEFAULT
Update:	not recommended
dsnhdecp:	SQLDELI

This option is effective for all varieties of COBOL.

The value in this field also determines which character is the escape character for delimited identifiers in dynamic SQL. If you specify an apostrophe in this field,

you get a quotation mark for your SQL escape character. If you specify a quotation mark in this field, you get an apostrophe for your SQL escape character.

For SQL statements that are embedded in COBOL programs, COBOL precompiler options specify which character is the SQL string delimiter and which character is the SQL escape character. If you specify DEFAULT in this field, a quotation mark is passed to the precompiler as the default SQL string delimiter.

Some applications might require a particular value for the SQL STRING DELIMITER. Determine the required values for those applications before installing DB2.

The following table shows the different combinations of character string delimiters that result from specifying different values in the STRING DELIMITER field and the SQL STRING DELIMITER field.

Table 69. Effect of STRING DELIMITER field and SQL STRING DELIMITER field on SQL and COBOL string delimiters

When you want this combination of character string delimiters...			Specify this in STRING DELIMITER field	Specify this in SQL STRING DELIMITER field
COBOL	**Dynamic SQL**	**Embedded SQL**		
"	'	"	DEFAULT	DEFAULT
'	'	'	'	'
"	"	"	"	"
"	'	'	"	'

The values that you specify in the STRING DELIMITER field and the SQL STRING DELIMITER field are also used by the program preparation panels, the DSNH CLIST, and the precompiler. The following table shows why you might specify different combinations of values in these fields.

Table 70. Effect of the STRING DELIMITER field and the SQL STRING DELIMITER field on precompiler options

Purpose	STRING DELIMITER field	SQL STRING DELIMITER field
Force APOST default (even in COBOL) and provide a default similar to APOST in DB2 Server for VSE & VM.	'	'
Change dynamic query string delimiter to the quotation mark. This is helpful if you use COBOL with the QUOTE option. It allows queries to be tested with dynamic SQL and moved into the program more easily.	"	"
Enables compatibility with the DB2 Server for VSE & VM option.	"	'

DIST SQL STR DELIMTR field (DSQLDELI DECP value)

The DSQLDELI DECP value specifies whether an apostrophe or quotation mark is to be used as the SQL string delimiter for bind operations at this DB2 site. This delimiter is to be used when the requester does not give DB2 that information.

In most cases, requesters tell DB2 whether the apostrophe or the quotation mark is to be used as the SQL string delimiter.

Acceptable values:	' (apostrophe) or " (quotation mark)
Default:	' (apostrophe)
Update:	not recommended
dsnhdecp:	DSQLDELI

MIXED DATA field (MIXED DECP value)

The value of the MIXED DECP field indicates how the EBCDIC CCSID and ASCII CCSID fields are to be interpreted by DB2.

The MIXED DATA option has no effect on the UNICODE CCSID field. Regardless of the setting for MIXED DATA, UNICODE UTF-8 data is considered mixed data and is processed according to the rules for mixed data.

Acceptable values:	YES, NO
Default:	NO
Update:	not recommended
dsnhdecp:	MIXED

YES
The CCSID that is specified in the EBCDIC CCSID and ASCII CCSID field must be the mixed CCSID for the encoding scheme. From this, DB2 determines the associated SBCS and DBCS CCSIDs for the encoding scheme. MIXED DATA YES allows EBCDIC and ASCII mixed-character data and graphic data to be defined.

Important: MIXED DATA applies to both the EBCDIC CCSID and ASCII CCSID. If you choose MIXED DATA YES, you must select mixed CCSIDs for EBCDIC and ASCII.

NO The CCSID specified in the EBCDIC CCSID or ASCII CCSID field must be the CCSID for the encoding scheme. MIXED DATA NO does not allow for EBCDIC or ASCII mixed character data or graphic data to be defined.

For EBCDIC data, specify whether the code points X'0E' and X'0F' have special meaning as the shift-out and shift-in controls for character strings that include double-byte characters.

- NO indicates that these code points have no special meaning. Therefore, all character strings are single-byte character set (SBCS) data.
- YES indicates that these code points have the special meaning described above. Therefore, character strings can be either SBCS or MIXED data.

Related reference:

"EBCDIC and ASCII support" on page 945

EBCDIC CCSID field (SCCSID, MCCSID, and GCCSID DECP values)

The value of the EBCDIC CCSID field specifies the default CCSID for EBCDIC-encoded character data that is stored in your DB2 subsystem or data sharing system.

DB2 uses this value to perform conversion of character data that is received from external sources including other database management systems. Choose this value carefully to avoid loss of data integrity. The values that you choose for EBCDIC CCSID and ASCII CCSID are closely related.

Acceptable values:	1 to 65533
Default:	(none)
Update:	not recommended; data integrity may be compromised
dsnhdecp:	SCCSID (single-byte), MCCSID (mixed), GCCSID (graphic)

If you specify NO for the MIXED DATA field, the MCCSID and GCCSID values are 65534. If you specify YES for the MIXED DATA field, ensure that you use the correct single-byte CCSID and MCCSID.

Recommendation: Use this parameter to specify an MCCSID. By doing so, you also receive system CCSIDs for your SBCS and GRAPHIC data. If you edit DSNHDECP directly, you will not receive the system CCSIDs for your SBCS and GRAPHIC data.

Conversions are determined in the following order:

1. SYSIBM.SYSSTRINGS.
2. z/OS Unicode conversion services.

Considerations for mixed data:

- If the MIXED DATA field is set to YES, you must specify a MCCSID from Table 172 on page 947. An error occurs if you do not specify a CCSID or if the CCSID you specify is not listed in the table.
- If you specify 930, 1390, or 5026, Katakana characters are allowed in ordinary identifiers, and letters are not changed to uppercase.

If you specify a CCSID that is recognized by DB2 but is inappropriate for your site, data might be corrupted. For example, assume that the coded character set at your site is 37, but you specify 500 as the system CCSID. If DB2 receives data with a CCSID of 37, the data might be corrupted because character conversion does not occur. Conversely, if DB2 receives data with a CCSID other than 500 and a conversion is made from that CCSID to 500, the data may be corrupted because character conversion does occur.

Altering CCSIDs can be very disruptive to a system. Converting to a CCSID that supports the euro symbol is potentially less disruptive because specific pre-euro CCSIDs map to specific CCSIDs for the euro. Converting to a different CCSID for other reasons, particularly when a DB2 system has been operating with the wrong CCSID, could render data unusable and unrecoverable.

Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.

Related concepts:

"Euro symbol support" on page 948

Related reference:

"EBCDIC and ASCII support" on page 945

Related information:

Character Data Representation Architecture Reference

ASCII CCSID field (ASCCSID, AMCCSID, AGCCSID DECP values)

The ASCII CCSID field identifies the default coded character set identifier (CCSID) for ASCII-encoded character data that is to be stored in your DB2 subsystem or data sharing system.

DB2 uses the value of this field to convert character data that is received from external sources, including other database management systems. You must specify a value for this field, even if you do not have or plan to create ASCII-encoded objects. Choose this value carefully to prevent loss of data integrity.

Acceptable values:	1 to 65533
Default:	(none)
Update:	not recommended; data integrity may be compromised
dsnhdecp:	ASCCSID (single-byte), AMCCSID (mixed), AGCCSID (graphic)

Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.

Related reference:

"EBCDIC and ASCII support" on page 945

"EBCDIC CCSID field (SCCSID, MCCSID, and GCCSID DECP values)" on page 367

UNICODE CCSID field (USCCSID, UMCCSID, and UGCCSID DECP values)

The value of the UNICODE CCSID field specifies the CCSID that is to be used for Unicode data.

Acceptable values:	1208
Default:	1208
Update:	not recommended; data integrity may be compromised
dsnhdecp:	USCCSID (367 for single-byte), UMCCSID (1208 for mixed), UGCCSID (1200 for graphic)

Accept the default CCSID for Unicode.DB2 currently allows specification of only CCSID 1208 for this value. DB2 automatically chooses the CCSIDs for double-byte and single-byte data. Do not change CCSID values after they have been specified. SQL results might be unpredictable if you do not accept the default.

DEF ENCODING SCHEME field (ENSCHEME DECP value)

The ENSCHEME DECP value specifies the default format in which data is to be stored in DB2.

Acceptable values:	EBCDIC, ASCII, UNICODE
Default:	EBCDIC
Update:	not recommended
dsnhdecp:	ENSCHEME

If you set DEF ENCODING SCHEME to ASCII or EBCDIC and MIXED DATA is set to YES, specify a mixed CCSID.

The DDL uses the default encoding scheme in the following cases:

- CREATE DATABASE
- CREATE DISTINCT TYPE
- CREATE FUNCTION
- CREATE GLOBAL TEMPORARY TABLE
- DECLARE GLOBAL TEMPORARY TABLE
- CREATE TABLESPACE (in DSNDB04 database)

APPLICATION ENCODING field (APPENSCH DECP value)

The APPENSCH DECP value determines the system's default application encoding scheme, which affects how DB2 interprets data coming into DB2.

For example, if your default application encoding scheme is set to 37, and your EBCDIC coded character is set to 500, DB2 converts all data coming into the system to 500 from 37 before using it. This includes, but is not limited to, SQL statement text and host variables.

Acceptable values:	ASCII, EBCDIC, UNICODE, or CCSID (1 to 65533)
Default:	EBCDIC
Update:	not recommended
dsnhdecp:	APPENSCH

The following statements set the value of the host variable and do not require the package to be bound into the plan:

```
SET CURRENT PACKAGE SET = :HV ,
SET :HV = CURRENT PACKAGE SET ,
SET :HV = CURRENT PACKAGE PATH ,
SET CURRENT PACKAGE PATH = :HV
```

The host variable uses the system's default application encoding scheme, even when the application is bound with the ENCODING(EBCDIC/UNICODE) bind option.

The default value, EBCDIC, causes DB2 to retain the behavior of previous versions of DB2. (Assume that all data is in the EBCDIC system CCSID.)

LOCALE LC_CTYPE field (LC_TYPE DECP value)

The LC_TYPE DECP value specifies the system LOCALE LC_CTYPE. A *locale* is the part of your system environment that depends on language and cultural conventions. An *LC_CTYPE* is a subset of a locale that applies to character functions.

Acceptable values:	A valid locale of 0 to 50 characters.
Default:	Blank
Update:	not recommended
dsnhdecp::	LC_TYPE

The UPPER, LOWER, and TRANSLATE scalar functions use the CURRENT LOCALE LC_CTYPE system default or special register. The results of these functions can vary, depending on the setting of the locale.

Recommendation: Use the default value for LOCALE LC_CTYPE unless you need to execute the UPPER, LOWER, or TRANSLATE functions for data that must be interpreted by using the rules provided by specific locales. For example, specify En_US for English in the United States or Fr_CA for French in Canada.

Related reference:

SET CURRENT LOCALE LC_CTYPE (DB2 SQL)

LC_CTYPE category (XL C/C++ Programming Guide)

DECFLOAT ROUNDING MODE field (DEF_DECFLOAT_ROUND_MODE DECP value)

The DEF_DECFLOAT_ROUND_MODE DECP value specifies the system default action that is to be used for rounding decimal floating point values.

Acceptable values:	ROUND_CEILING, ROUND_DOWN, ROUND_FLOOR, ROUND_HALF_DOWN, ROUND_HALF_EVEN, ROUND_HALF_UP, ROUND_UP
Default:	ROUND_HALF_EVEN
Update:	not recommended
dsnhdecp:	DEF_DECFLOAT_ROUND_MODE

ROUND_CEILING
Rounds towards +infinity. If all of the discarded digits are zero or if the sign is negative, DB2 removes the discarded digits and the rest of the result remains the same. In all other cases, DB2 increments the result coefficient by 1.

ROUND_DOWN
Rounds towards 0. DB2 ignores the discarded digits.

ROUND_FLOOR
Rounds towards -infinity. If all of the discarded digits are zero or if the sign is positive, DB2 removes the discarded digits and the rest of the result remains the same. In all other cases, DB2 increments the result coefficient by 1.

ROUND_HALF_DOWN
Rounds to the nearest number. If the discarded digits are greater than 0.5 of the value of a one in the next left position, DB2 increments the result coefficient by 1. If the discarded digits are equal or less than 0.5, DB2 ignores the discarded digits.

ROUND_HALF_EVEN
Rounds to the nearest number. If the discarded digits are greater than 0.5, the value of a one in the next left position, DB2 increments the result coefficient by 1. If the discarded digits are less than 0.5, DB2 ignores the discarded digits. If the discarded digits are 0.5 and the rightmost digit is

even, DB2 does not alter the result coefficient. If the discarded digits are 0.5 and the rightmost digit is odd, DB2 increments the result coefficient by 1 to make it an even digit.

ROUND_HALF_UP
Rounds to the nearest number. If the discarded digits are greater than or equal to 0.5 of the value of a one in the next left position, DB2 increments the result coefficient by 1. In all other cases, DB2 ignores the discarded digits.

ROUND_UP
Rounds away from 0. If all of the discarded digits are zero, DB2 removes the discarded digits and the rest of the result remains the same. In all other cases, DB2 increments the result coefficient by 1.

Related reference:

- BIND and REBIND options (DB2 Commands)
- SET CURRENT DECFLOAT ROUNDING MODE (DB2 SQL)
- Syntax and options of the UNLOAD control statement (DB2 Utilities)

Application programming defaults panel 2: DSNTIP4

The DSNTIP4 panel is a continuation of the DSNTIPF panel, and it is used to set application programming default values. The values that you specify on this panel are used as default values by the program preparation panels, the program preparation CLIST (DSNH), and the precompiler.

The values that you specify on this panel can also be used as default values by other programs, such as DB2 QMF.

```
DSNTIP4         INSTALL DB2 - APPLICATION PROGRAMMING DEFAULTS PANEL 2
===> _

Enter data below:

 1  MINIMUM DIVIDE SCALE ===> NO       NO or YES for a minimum of 3 digits
                                       to right of decimal after division
 2  DECIMAL ARITHMETIC   ===> DEC15    DEC15, DEC31, 15, 31 or DPP.S
 3  USE FOR DYNAMICRULES ===> YES      YES or NO
 4  DESCRIBE FOR STATIC  ===> YES      Allow DESCRIBE for STATIC SQL. NO or YES
 5  DATE FORMAT          ===> ISO      ISO, JIS, USA, EUR, LOCAL
 6  TIME FORMAT          ===> ISO      ISO, JIS, USA, EUR, LOCAL
 7  LOCAL DATE LENGTH    ===> 0        10-254 or 0 for no exit
 8  LOCAL TIME LENGTH    ===> 0         8-254 or 0 for no exit
 9  IMPLICIT TIME ZONE   ===> CURRENT  Implicit time zone for TIMESTAMP values
                                        (CURRENT, SESSION , or -12:59 to +14:00)
10  STD SQL LANGUAGE     ===> NO       NO or YES
11  PAD NUL-TERMINATED   ===> NO       NO or YES

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 34. Application programming defaults panel: DSNTIP4

Related tasks:

Processing SQL statements (DB2 Application programming and SQL)

Related reference:

"Application programming defaults panel 1: DSNTIPF" on page 362

MINIMUM DIVIDE SCALE field (DECDIV3 subsystem parameter)

The DECDIV3 subsystem parameter determines whether to retain at least three digits to the right of the decimal point after any decimal division.

Acceptable values:	YES, NO
Default:	NO
Update:	option 18 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DECDIV3

Certain accounting applications might need this option.

YES Retains at least three digits to the right of the decimal point after any decimal division.

NO Does not retain three digits to the right of the decimal point after decimal division. Use NO, the default, to accept the usual rules for decimal division in SQL.

Related concepts:

Arithmetic with two decimal operands (DB2 SQL)

DECIMAL ARITHMETIC field (DECARTH DECP value)

The DECARTH DECP value specifies the rules that are to be used when both operands in a decimal operation have precisions of 15 or less.

Acceptable values:	DEC15, DEC31, 15, 31, or DPP.S
Default:	DEC15
Update:	not recommended; cannot be changed during migration
dsnhdecp:	DECARTH

DEC15 or 15
Specifies the rules that do not allow a precision greater than 15 digits.

DEC31 or 31
Specifies the rules that allow a precision of up to 31 digits. The rules for DEC31 are always used if either operand has a precision greater than 15. If you chose DEC15 for your previous installation, choosing DEC31 can produce different results for operations on existing data.

D*PP.S* "PP" is either 15 or 31 and "S" is a number between 1 and 9. If the form "DPP.S" is used, "PP" represents the precision that is to be used with the rules that are used for DEC15 or DEC31, and "S" represents the minimum divide scale to use for division operations. The separator that is used in the form "DPP.S" can be either the "." or the "," character, regardless of the setting of the default decimal point. An error occurs if any other values are specified.

This installation option applies to dynamic SQL by becoming the initial value for the CURRENT PRECISION special register, and it provides the default for the DEC

precompiler option. DEC15 is sufficient for most sites. Do not choose DEC31 unless you are certain that you need the extra precision. If you use DEC31, you are more likely to get a bind error, particularly in division operations.

Related concepts:

Arithmetic with two decimal operands (DB2 SQL)

USE FOR DYNAMICRULES field (DYNRULS DECP value)

The DYNRULS DECP value specifies whether DB2 is to use the application programming default values that are specified on this panel. The other option is to use the values of the DB2 precompiler options for dynamic SQL statements that are bound by using DYNAMICRULES bind, define, or invoke behavior.

Acceptable values:	YES or NO
Default:	YES
Update:	not recommended
dsnhdecp:	DYNRULS

YES DB2 uses the application programming defaults for these fields regardless of the DYNAMICRULES option:

- DECIMAL POINT IS
- STRING DELIMITER
- SQL STRING DELIMITER
- MIXED DATA
- DECIMAL ARITHMETIC

NO DB2 uses the DB2 precompiler values for dynamic SQL statements in plans or packages that are bound by using the DYNAMICRULES bind, define, or invoke behavior.

DESCRIBE FOR STATIC field (DESCSTAT subsystem parameter)

The DESCSTAT subsystem parameter controls whether DB2 is to build a DESCRIBE SQL descriptor area (SQLDA) when binding static SQL statements. DESCRIBE SQLDAs allow DESCRIBE requests for static SQL statements to execute successfully.

Normally, a DESCRIBE request cannot be issued against a static SQL statement, with the following exceptions:

- In a distributed environment, where DB2 for z/OS is the server, and the requester supports extended dynamic SQL. In this scenario, a DESCRIBE request that is executed on an SQL statement in the extended dynamic package appears to DB2 as a DESCRIBE on a static SQL statement in the DB2 package.
- When an application uses a stored procedure result set, and the application must allocate a cursor for that result set. The application can describe that cursor by using a DESCRIBE CURSOR statement. The SQL statement that is actually described is the one for which the cursor is declared in the stored procedure. If that statement is static, this requires that a static SQL statement must be described.

Acceptable values:	NO or YES
Default:	YES
Update:	option 18 on panel DSNTIPB

NO DB2 does not generate a DESCRIBE SQLDA at bind time for static SQL statements. If a DESCRIBE request is received at execution time, DB2 generates an error. However, if the describe request comes from a DESCRIBE CURSOR statement, DB2 satisfies the request but is able to provide only data type and length information. Column names are not provided.

YES DB2 generates a DESCRIBE SQLDA at bind time so that DESCRIBE requests for static SQL can be satisfied during execution. You must rebind the package after this value has been set to YES. Specifying YES increases the size of some packages because the DESCRIBE SQLDA is now stored with each statically bound SQL SELECT statement.

YES is the default value.

If your DB2 for z/OS subsystem or DB2 for Linux, UNIX, and Windows systems use the IBM Data Server Driver for JDBC and SQLJ, or if your DB2 for Linux, UNIX, and Windows systems use the CLI driver, you must set the DESCSTAT parameter to YES.

DATE FORMAT field (DATE DECP value)

The DATE DECP value controls the default format that is to be used to represent dates in output.

Acceptable values:	ISO, USA, EUR, JIS, LOCAL
Default:	ISO
Update:	not recommended
dsnhdecp:	DATE

The date formats that are associated with the acceptable values for this field are shown in the table below.

Table 71. Date formats

Format name	Abbreviation	Format	Example
International Standards Organization	ISO	yyyy-mm-dd	2003-12-23
IBM USA standard	USA	mm/dd/yyyy	12/23/2003
IBM European standard	EUR	dd.mm.yyyy	23.12.2003
Japanese Industrial Standard Christian Era	JIS	yyyy-mm-dd	2003-12-23
Locally defined (by an installation exit routine)	LOCAL	your choice	

DB2 accepts dates in any format as input. DB2 interprets the input date based on the punctuation, and then provides date output in the format that you specify for this parameter. If you use LOCAL, you must provide a date exit routine to perform date formatting.

Related concepts:

Date and time routines (DB2 Administration Guide)

TIME FORMAT field (TIME DECP value)

The value of the TIME FORMAT field specifies which format is to be used as a default output to represent times.

Acceptable values:	ISO, USA, EUR, JIS, LOCAL
Default:	ISO
Update:	not recommended
dsnhdecp:	TIME

Table 72. Time formats

Format name	Abbreviation	Time format	Example
International Standards Organization[1]	ISO[1]	hh.mm.ss	13.30.05
IBM USA standard	USA	hh:mm AM or PM	1:30 PM
IBM European standard	EUR	hh.mm.ss	13.30.05
Japanese Industrial Standard Christian Era	JIS	hh:mm:ss	13:30:05
Installation-defined	LOCAL	Any installation-defined form	—

Note: 1. This is an earlier version of the ISO format. JIS can be used to get the current ISO format.

In the USA format:

- The minutes can be omitted, thereby specifying 00 minutes. For example, 1 PM is equivalent to 1:00 PM.
- The letters A, M, and P can be lowercase.
- A single blank must precede the AM or PM.
- The hour must not be greater than 12 and cannot be 0 except for the special case of 00:00 AM.

Using the ISO format of the 24-hour clock, the correspondence between the USA format and the 24-hour clock is as follows:

- 12:01 AM through 12:59 AM correspond to 00.01.00 through 00.59.00.
- 01:00 AM through 11:59 AM correspond to 01.00.00 through 11.59.00.
- 12:00 PM (noon) through 11:59 PM correspond to 12.00.00 through 23.59.00.
- 12:00 AM (midnight) corresponds to 24.00.00.
- 00:00 AM (midnight) corresponds to 00.00.00.

DB2 can accept a time in any format as input. DB2 interprets the input time based on the punctuation and then provides time output in the format that you specify for this parameter. If you use LOCAL, you must provide a time exit routine to perform time formatting.

Related concepts:

Date and time routines (DB2 Administration Guide)

LOCAL DATE LENGTH field (DATELEN DECP value)

If you use a locally defined date exit routine, the DATELEN DECP value defines the length of the longest field that is required to hold a date. To use your own default date format, you must also enter LOCAL for the DATE FORMAT field on panel DSNTIP4.

Acceptable values:	0, 10 to 254
Default:	0
Update or migrate:	not recommended
dsnhdecp:	DATELEN

Accept the default value of `0` if you want to use one of the IBM-supplied date formats (ISO, JIS, USA, or EUR). This value indicates that no user-defined date format exists in your system.

LOCAL TIME LENGTH field (TIMELEN DECP value)

Accept the default value of **0** if you want to use one of the IBM-supplied time formats (ISO, JIS, USA, or EUR).

Acceptable values:	0, 8 to 254
Default:	0
Update or migrate:	not recommended
dsnhdecp:	TIMELEN

This indicates that no user-defined time format exists in your system. If you use a locally defined time exit routine, enter the length of the longest field that is required to hold a time. If you want your own time format to be the default, enter LOCAL for field 2 on this panel.

IMPLICIT TIME ZONE field (IMPLICIT_TIMEZONE DECP value)

The IMPLICIT_TIMEZONE DECP value determines the implicit time zone that is to be used when a time zone is not provided. This parameter applies to DB2 table columns and routing parameters that are declared with TIMESTAMP WITH TIME ZONE data types.

Acceptable values:	CURRENT, SESSION, -12:59 to +14:00
Default:	CURRENT
Update:	not recommended[1]
dsnhdecp:	IMPLICIT_TIMEZONE

[1]Given the way that the implicit time zone is defined, a query might return different results before and after the IMPLICIT_TIMEZONE setting is changed. Or, if IMPLICIT_TIMEZONE is set to SESSION and an application changes the value of SESSION TIME ZONE through a SET SESSION TIME ZONE statement, a query might also return a different result. These situations apply to both static and dynamic applications.

CURRENT
Uses the setting of the CURRENT TIME ZONE special register as the implicit time zone. This is the default setting.

SESSION
Uses the setting of the SESSION TIME ZONE special register as the implicit time zone.

-12:59 to +14:00
Uses the value of the string as the implicit time zone. The format of the value is *th*:*tm*, where *th* is the time zone hour (-12 to +14), and *tm* is the time zone minute (00 to 59). This value is an offset of hours and minutes from the coordinated universal time (UTC). If the string does not indicate '+' or '-', a '+' value is assumed

STD SQL LANGUAGE field (STDSQL DECP value)

The STD SQL LANGUAGE field specifies the SQL standard that is to be used in writing application programs. Specifically, the parameter specifies whether the SQL language conforms to the portions of the 1992 ANSI SQL standard that are implemented by DB2.

Acceptable values:	YES, NO
Default:	NO
Update:	not recommended
dsnhdecp:	STDSQL

YES The SQL language that is used in application programs conforms to the portions of the 1992 ANSI SQL standard that are implemented by DB2.

NO Programs are written in accordance with the SQL language that is defined by DB2.

PAD NUL-TERMINATED field (PADNTSTR DECP value)

The PADNTSTR subsystem parameter determines whether output host variables that are nul-terminated strings are to be padded with blanks and a nul-terminator.

Acceptable values:	NO, YES
Default:	NO
Update:	not recommended
dsnhdecp:	PADNTSTR

NO Nul-terminated output host variables have the nul-terminator placed at the end of actual data that is returned in the host variable.

YES Nul-terminated output host variables have the nul-terminator placed at the end of the string, after the string has been padded with blanks from the end of the actual data to the length of the output host variable.

Performance and optimization panel: DSNTIP8

The DSNTIP8 panel is a continuation of the DSNTIP4 panel. It is used to set application programming default values pertaining to performance and optimization.

```
DSNTIP8         INSTALL DB2 - PERFORMANCE AND OPTIMIZATION
 ===> _

 Enter data below:

  1  CURRENT DEGREE        ===> 1        1 or ANY
  2  CACHE DYNAMIC SQL     ===> YES      NO or YES
  3  OPTIMIZATION HINTS    ===> NO       Enable optimization hints. NO or YES
  4  MAX DEGREE            ===> 0        Maximum degree of parallelism. 0-254
  5  PARALLELISM EFFICIENCY===> 50       Efficiency of parallelism. 0-100
  6  IMMEDIATE WRITE       ===> NO       NO, YES
  7  EVALUATE UNCOMMITTED  ===> NO       Evaluate uncommitted data. NO or YES
  8  SKIP UNCOMM INSERTS   ===> NO       Skip uncommitted inserts. NO or YES
  9  CURRENT REFRESH AGE   ===> 0        0 or ANY
 10  CURRENT MAINT TYPES   ===> SYSTEM   NONE, SYSTEM, USER, ALL
 11  STAR JOIN QUERIES     ===> DISABLE  DISABLE, ENABLE, 1-32768
 12  MAX DATA CACHING      ===> 20       0-512
 13  PLAN MANAGEMENT       ===> EXTENDED OFF, BASIC, EXTENDED
 14  PLAN MANAGEMENT SCOPE ===> STATIC   ALL, STATIC, DYNAMIC
 15  RANDOMIZE XML DOCID   ===> NO       NO or YES



 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 35. Performance and optimization panel: DSNTIP8

CURRENT DEGREE field (CDSSRDEF subsystem parameter)

The CDSSRDEF subsystem parameter determines the default value that is to be used for the CURRENT DEGREE special register. The default value is used when a degree is not explicitly set in the SQL statement SET CURRENT DEGREE.

Acceptable values:	1, ANY
Default:	1
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CDSSRDEF

1 Specifies that when a query is dynamically prepared, the execution of that query will not use parallelism.

If this value is specified, DB2 does not use any optimization hints for parallelism.

ANY Specifies that when a query is dynamically prepared, the execution of that query can involve parallelism.

Recommendation: In almost all situations, accept the default value of 1. You should use parallelism selectively where it provides value, rather than globally. Although parallelism can provide a substantial reduction in elapsed time for some queries with only a modest overhead in processing time, parallelism does not always provide the intended benefit. For some queries and in many other situations, query parallelism does not provide an improvement, or it uses too many resources. If you are using nearly all of your CPU, I/O, or storage resources,

parallelism is more likely to cause degradation of performance. Use parallelism only where it is most likely to provide benefits.

CACHE DYNAMIC SQL field (CACHEDYN subsystem parameter)

The CACHEDYN subsystem parameter controls whether prepared, dynamic SQL statements are to be cached for later use by eligible application processes.

Acceptable values:	YES, NO
Default:	YES
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHEDYN
Security parameter:	Yes

YES Causes prepared, dynamic SQL statements to be cached in the EDM dynamic statement cache. If you specify YES, consider this usage when you calculate your EDM pool size. If you specify YES, you must also specify YES for the USE PROTECTION field on panel DSNTIPP.

NO Causes prepared, dynamic SQL statements to not be cached.

Note: This is a security-related parameter. When RACF authorization is used and this parameter is set, the changes that are made in RACF profiles are not immediately reflected. This setting might result in a user (cached AUTHID) being able to issue the dynamic statements until the cache is refreshed.

Related tasks:

"Calculating EDM pool space for the prepared-statement cache" on page 56

Related reference:

"USE PROTECTION field (AUTH subsystem parameter)" on page 410

OPTIMIZATION HINTS field (OPTHINTS subsystem parameter)

The OPTHINTS subsystem parameter specifies whether DB2 applies optimization hints when static SQL statements are bound or dynamic SQL statements are prepared.

DB2 applies matching optimization hints for SQL statements only when the value of the OPTHINTS subsystem parameter is set to YES.

Acceptable values:	NO, YES
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM OPTHINTS

If you accept the default value, you cannot use optimization hints.

When the value is set to YES, DB2 applies hints in the following situations:

- For rows in a PLAN_TABLE instance that contain a value in the OPTHINT column that matches the value of the OPTHINT bind option or the CURRENT OPTIMIZATION HINT special register.
- For any instance of a statement whose application environment and text matches rows in the SYSIBM.SYSQUERY catalog table.

Related concepts:

CURRENT OPTIMIZATION HINT (DB2 SQL)

Related tasks:

Enabling optimization hints (DB2 Performance)

Influencing access path selection by using optimization hints (DB2 Performance)

Related reference:

PLAN_TABLE (DB2 Performance)

SYSIBM.SYSQUERY table (DB2 SQL)

BIND and REBIND options (DB2 Commands)

MAX DEGREE field (PARAMDEG subsystem parameter)

The PARAMDEG subsystem parameter specifies the maximum degree of parallelism that is to be allowed for a parallel group. When you specify a value for this parameter, you limit the degree of parallelism so that DB2 cannot create too many parallel tasks that use virtual storage.

Acceptable values:	0 to 254
Default:	0
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PARAMDEG

The default value of 0 means that DB2 will choose a maximum degree of parallelism that is based on the system configuration.

When optimization hints for parallelism are used, the value of the PARAMDEG subsystem parameter does not limit the degree of parallelism at bind time. However, the value of the PARAMDEG subsystem parameter is enforced at execution time. So, if the value of the PARAMDEG subsystem parameter is lower than the degree of parallelism that is specified at bind time, then the degree of parallelism is reduced at execution time.

Related concepts:

Optimization hints (DB2 Performance)

Related tasks:

Enabling parallel processing (DB2 Performance)

Disabling query parallelism (DB2 Performance)

PARALLELISM EFFICIENCY field (PARA_EFF subsystem parameter)

The PARA_EFF subsystem parameter controls the efficiency that DB2 assumes for parallelism when DB2 chooses an access path. The integer value that is used for this parameter represents a percentage efficiency.

Acceptable values:	0 – 100
Default:	50
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PARA_EFF

Related tasks:

Tuning parallel processing (DB2 Performance)

IMMEDIATE WRITE field (IMMEDWRI subsystem parameter)

The IMMEDWRI subsystem parameter determines when updates to group buffer pool-dependent buffers are to be written to the coupling facility.

Acceptable values:	YES, NO
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6GRP IMMEDWRI

YES DB2 immediately writes the page to the coupling facility after the update occurs.

NO DB2 does not immediately write the change buffer to the coupling facility. Instead, DB2 waits until phase 1 of commit.

If either the IMMEDWRITE bind option or the IMMEDWRI subsystem parameter is set to YES, the value of the immediate write option at run time is also YES.

EVALUATE UNCOMMITTED field (EVALUNC subsystem parameter)

The EVALUNC subsystem parameter controls whether predicate evaluation is to be allowed on uncommitted data of other transactions. This parameter applies only to stage 1 predicate processing that uses table access for queries with isolation level RS or CS. Table access includes table space scan, index-to-data access, and RID-list processing.

Acceptable values:	NO, YES
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EVALUNC

NO Predicate evaluation occurs only on committed data (or on the application's own uncommitted changes). NO ensures that all qualifying data is always included in the answer set.

This is the default value.

YES Predicate evaluation can occur on uncommitted data of other transactions. With YES, data might be excluded from the answer set. Data that does not satisfy the predicate during evaluation but then, because of undo processing (ROLLBACK or statement failure), reverts to a state that does satisfy the predicate is missing from the answer set. A value of YES enables DB2 to take fewer locks during query processing. The number of avoided locks depends on:

- The query's access path
- The number of evaluated rows that do not satisfy the predicate
- The number of those rows that are on overflow pages

Recommendation: Specify YES to improve concurrency if your applications can tolerate returned data that might falsely exclude any data that would be included as the result of undo processing (ROLLBACK or statement failure).

Although the option influences whether predicate evaluation can occur on uncommitted data, it does not influence whether uncommitted data is returned to an application. Queries with isolation level RS or CS return only committed data. They never return the uncommitted data of other transactions, even if predicate evaluation occurs. If data satisfies the predicate during evaluation, the data is locked as needed, and the predicate is re-evaluated as needed before the data is returned to the application.

SKIP UNCOMM INSERTS field (SKIPUNCI subsystem parameter)

The SKIPUNCI subsystem parameter specifies whether statements are to ignore a row that was inserted by another transaction if the row has not been committed or aborted.

Acceptable values:	NO, YES
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SKIPUNCI

SKIP UNCOMM INSERTS applies only to statements running with row-level locking and read stability or cursor stability isolation level.

NO DB2 waits for the inserted row to be committed or rolled back. It then processes the row as a qualifying row if the insert commits, or it moves on to find another row if the insert is rolled back. If a transaction performs one or more inserts, and then spawns a second transaction, specify NO for SKIP UNCOMM INSERTS if the first transaction needs the second transaction to wait for the outcome of the inserts.

This is the default value.

YES
DB2 behaves as though the uncommitted row has not yet arrived and the row is skipped. Specifying a value of YES offers greater concurrency than the default value of NO.

CURRENT REFRESH AGE field (REFSHAGE subsystem parameter)

The REFSHAGE subsystem parameter specifies the default value for the CURRENT REFRESH AGE special register. The default value is used when no value is explicitly set by using the SET CURRENT REFRESH AGE statement.

Acceptable values:	0, ANY
Default:	0
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM REFSHAGE

Accepting the default value of 0 disables query rewrite using deferred materialized query tables.

CURRENT MAINT TYPES field (MAINTYPE subsystem parameter)

The MAINTYPE subsystem parameter specifies the default value for the CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION special register.

This value is to be used when no value is explicitly set by using the SQL statement SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION.

Acceptable values:	NONE, SYSTEM, USER, ALL
Default:	SYSTEM
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAINTYPE

NONE
Does not allow query rewrite.

SYSTEM
Allows query rewrite by using system-maintained materialized query tables (SYSTEM) when CURRENT REFRESH AGE is set to ANY.

USER Allows query rewrite by using user-maintained materialized query tables when CURRENT REFRESH AGE is set to ANY.

ALL Allows query rewrite by using both system-maintained and user-maintained materialized query tables.

STAR JOIN QUERIES field (STARJOIN subsystem parameter)

The STARJOIN subsystem parameter specifies whether star join processing is to be enabled.

Acceptable values:	DISABLE, ENABLE, 1 to 32768
Default:	DISABLE
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STARJOIN

A value of 1 indicates that the fact table will be the largest table in a star join query that does not have fact/dimension ratio checking. A value of 2 to 32768 indicates that DB2 should use the ratio of the star join table and the largest dimension table.

Related concepts:

Star schema access (DB2 Performance)

MAX DATA CACHING field (MXDTCACH subsystem parameter)

The MXDTCACH subsystem parameter specifies the maximum amount of memory, in MB, that is to be allocated for data caching per thread.

Acceptable values:	0 to 512
Default:	20
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MXDTCACH

If you specify 0, DB2 does not use data caching during query execution and you can apply sparse index only. If you specify a value between 1 and 512, DB2 allocates memory from above the 2 GB bar pool up to the specified size for data caching.

PLAN MANAGEMENT field (PLANMGMT subsystem parameter)

The PLANMGMT subsystem parameter specifies which default plan management policy to use when the PLANMGMT option is not explicitly specified for the bind or rebind of a package.

Acceptable values:	OFF, BASIC, EXTENDED
Default:	EXTENDED
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PLANMGMT

OFF
DB2 does not store access path information for SQL queries.

BASIC
DB2 stores access path information for SQL queries in the repository. In addition, one old access path is retained. This access path is referred to as the PREVIOUS copy.

EXTENDED
DB2 stores access path information for SQL queries in the repository. In addition, two old access paths are retained. These access paths are referred to as the PREVIOUS and ORIGINAL copies. This is the default setting of the PLANMGMT subsystem parameter.

PLAN MANAGEMENT SCOPE field (PLANMGMTSCOPE subsystem parameter)

The PLANMGMTSCOPE subsystem parameter specifies the default plan management scope to use when the PLANMGMTSCOPE option is not explicitly specified for the bind or rebind of a package.

The value in this field is meaningful only when the value of the PLAN MANAGEMENT field is BASIC or EXTENDED.

Acceptable values:	STATIC
Default:	STATIC
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PLANMGMTSCOPE

STATIC
DB2 stores access path information for only static SQL queries. This is the default value for the PLANMGMTSCOPE parameter.

RANDOMIZE XML DOCID field (XML_RANDOMIZE_DOCID subsystem parameter)

Specifies whether DB2 is to sequentially or randomly generate the DOCID values for XML columns.

This parameter does not affect existing tables that have XML columns. Those tables continue to generate DOCIDs in the order that was specified by XML_RANDOMIZE_DOCID when the table was created or in sequential order if the parameter was not specified.

Acceptable values:	NO, YES

Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP XML_RANDOMIZE_DOCID

NO DB2 generates the DOCID values for XML columns sequentially.

YES DB2 generates the DOCID values for XML columns randomly. Setting this parameter to YES can be beneficial for performance when concurrent inserts are expected.

Once a table starts to randomly generate DOCID values for XML columns, it cannot be changed to generate the value sequentially. Changing this parameter back to NO does not have any effect on tables that have had columns added with a randomized DOCID.

IRLM panel 1: DSNTIPI

The entries on the DSNTIPI panel affect the installation of the internal resource lock manager (IRLM).

You must use one IRLM for each DB2 subsystem.

```
DSNTIPI         INSTALL DB2 - IRLM PANEL 1
===> _

Enter data below:

 1  INSTALL IRLM       ===> YES        IRLM is required for DB2. Should the
                                       IRLM distributed with DB2 be installed?
 2  SUBSYSTEM NAME     ===> IRLM       IRLM MVS subsystem name
 3  RESOURCE TIMEOUT   ===> 30         Seconds to wait for unavailable resource
 4  AUTO START         ===> YES        Start IRLM if not up. YES or NO
 5  PROC NAME          ===> IRLMPROC   Name of start procedure for IRLM
 6  TIME TO AUTOSTART  ===> 120        Time DB2 will wait for IRLM autostart
 7  U LOCK FOR RR/RS   ===> YES        Lock mode for update cursor with
                                       RR or RS isolation. YES or NO
 8  X LOCK FOR SEARCHED U/D ===> NO    Use X lock for searched updates or
                                       deletes. NO or YES or TARGET.
 9  START IRLM CTRACE  ===> NO         Start IRLM component traces at startup?
                                       NO or YES
10  IMS BMP TIMEOUT    ===> 4          Timeout multiplier for BMP. 1-254
11  DL/I BATCH TIMEOUT ===> 6          Timeout multiplier for DL/I. 1-254
12  RETAINED LOCK TIMEOUT ===> 0       Retained lock timeout multiplier. 0-254
PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 36. IRLM panel 1: DSNTIPI

Related concepts:

"IRLM address space (IRLMPROC) storage requirements" on page 47

Performance monitoring and tuning for data sharing environments (DB2 Data Sharing Planning and Administration)

Related reference:

"IRLM panel 2: DSNTIPJ" on page 391

INSTALL IRLM field

The INSTALL IRLM field determines whether IRLM subsystem entries are to be provided in job DSNTIJMV and whether an IRLM procedure is to be built.

Acceptable values:	YES, NO
Default:	YES
Update:	Edit the associated parameter in job DSNTIJUZ, the IRLM start procedure, and CLIST input member. Then, execute DSNTIJUZ and restart DB2.
DSNZP*xxx*:	none

YES The required entries are provided, and the IRLM procedure is built. .

NO No IRLM procedure is produced. On installation panel DSNTIPJ, all values are ignored except the LOCKS PER TABLE(SPACE) field and the LOCKS PER USER field.

If you do not have a new IRLM procedure created, ensure that your old IRLM procedure is updated with any new keywords that were added.

SUBSYSTEM NAME field (IRLMSID subsystem parameter)

The IRLMSID subsystem parameter specifies the name by which z/OS is to know the IRLM subsystem.

Acceptable values:	1 to 4 characters. First character must be A - Z, #, $, or @. Other characters must be A - Z, 1 - 9, #, $, or @.
Default:	IRLM
Update:	Edit the associated parameter in job DSNTIJUZ, the IRLM start procedure, and CLIST input member. Then, execute DSNTIJUZ and restart DB2.
DSNZP*xxx*:	DSN6SPRM IRLMSID

The name is used for communication between DB2 and the IRLM. This name is included in the z/OS subsystem table IEFSSN*xx*, where *xx* is the value of the SUBSYSTEM MEMBER field on installation panel DSNTIPM.

If you installed the IRLM for IMS, DB2's IRLM name must be different. Two IRLMs that reside in the same z/OS system must have unique z/OS subsystem names. If you already have IRLM installed, use the z/OS subsystem name for that IRLM. Otherwise, accept the default value, IRLM.

The associated IRLM PROC parameter is IRLMNM.

Related reference:

"COMMAND PREFIX field" on page 423

RESOURCE TIMEOUT field (IRLMRWT subsystem parameter)

The IRLMRWT subsystem parameter controls the number of seconds that are to elapse before a resource timeout is detected.

Acceptable values:	1 to 3600
Default:	30
Update:	Edit the associated parameter in job DSNTIJUZ, the IRLM start procedure, and CLIST input member.Then, execute DSNTIJUZ and restart DB2.
DSNZP*xxx*:	DSN6SPRM IRLMRWT

Timeout means that a lock request has waited for a resource (or for claims on a resource for a particular claim class to be released) longer than the number of

seconds specified on this option. The value that is specified for this option must be a multiple of the DEADLOCK TIME on installation panel DSNTIPJ because IRLM uses its deadlock timer to initiate timeout detection and deadlock detection. This value is rarely the actual time. For data sharing, the actual timeout period is longer than the timeout value.

Related concepts:

Global timeout processing (DB2 Data Sharing Planning and Administration)

Related tasks:

Specifying the amount of inactive time before a timeout (DB2 Performance)

Related reference:

"DEADLOCK TIME field" on page 394

AUTO START field (IRLMAUT subsystem parameter)

The IRLMAUT subsystem parameter defines whether DB2 is to automatically start and stop the IRLM.

Acceptable values:	YES, NO
Default:	YES
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM IRLMAUT

YES When DB2 starts, it tries to start the IRLM if the IRLM is not already started. When DB2 stops, IRLM automatically stops if the IRLM was started by DB2. However, IRLM will not automatically terminate if it is defined with DISCONNECT IRLM = NO.

NO DB2 terminates if the IRLM is not started when DB2 comes up.

Recommendation: Use YES if you use the IRLM for only a single DB2 subsystem.

When IRLM initializes, it is registered with the z/OS automatic restart manager (ARM). IRLM deregisters from the ARM when the IRLM is shut down normally. When IRLM terminates, it sends DB2 the registration information so that DB2 can determine whether IRLM was terminated normally. DB2 then deregisters from the ARM to prevent unwanted restarts. Otherwise, IRLM might be restarted with DB2, if AUTO START is YES.

Related concepts:

"IRLM names" on page 160

Related tasks:

Controlling the IRLM (DB2 Administration Guide)

PROC NAME field (IRLMPRC subsystem parameter)

The IRLMPRC subsystem parameter specifies the name of the IRLM procedure that z/OS is to invoke if the AUTO START field is set to YES.

Acceptable values:	1 to 8 characters
Default:	IRLMPROC
Update:	Edit the associated parameter in job DSNTIJUZ, the IRLM start procedure, and CLIST input member. Then, execute DSNTIJUZ and restart DB2
DSNZP*xxx*:	DSN6SPRM IRLMPRC

The name that is specified in the PROC NAME field **cannot** be the same as the subsystem name that is specified in the SUBSYSTEM NAME field.

The procedure is created during installation or migration by job DSNTIJMV and is placed in SYS1.PROCLIB. You can review the procedure by examining DSNTIJMV.

TIME TO AUTOSTART field (IRLMSWT subsystem parameter)

The IRLMSWT subsystem parameter specifies the IRLM wait time in seconds. This is the amount of time that DB2 is to wait for the IRLM to start during autostart. If the time expires, DB2 abends.

Acceptable values:	1 to 3600
Default:	120
Update:	Edit the associated parameter in job DSNTIJUZ, the IRLM start procedure, and CLIST input member. Then, execute DSNTIJUZ and restart DB2.
DSNZP*xxx*:	DSN6SPRM IRLMSWT

U LOCK FOR RR/RS field (RRULOCK subsystem parameter)

When operations use repeatable read (RR) or read stability (RS) isolation levels to access a table, a specified locking method is used. The RRULOCK subsystem parameter specifies whether DB2 is to use U (UPDATE) locks or S (SHARE) locks.

Acceptable values:	YES, NO
Default:	YES
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RRULOCK

YES
The lock mode for operations with RR or RS isolation levels is U. If your applications make frequent updates with repeatable-read isolation, the U-lock might provide greater concurrency than the S-lock. However, applications that require high concurrency are almost always more efficient if they use cursor stability (CS) isolation.

NO The lock mode for operations with RR or RS isolation levels is S. If the cursor in your applications includes the FOR UPDATE OF clause, but updates are infrequent, S-locks generally provide better performance.

X LOCK FOR SEARCHED U/D field (XLKUPDLT subsystem parameter)

The XLKUPDLT subsystem parameter specifies the locking method that is to be used when DB2 executes a searched update or delete.

Acceptable values:	YES, NO, TARGET
Default:	NO
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM XLKUPDLT

YES
DB2 uses an X-lock on qualifying rows or pages. For ISOLATION(CS), the lock is released if the rows or pages are not updated or deleted. For ISOLATION(RS) or ISOLATION(RR), an X-lock is retained until the next commit point. A value of YES is beneficial in a data sharing environment when

most or all searched updates and deletes use an index. If YES is specified and searched updates or deletes result in a table space scan, the likelihood of timeouts and deadlocks greatly increases.

NO DB2 uses an S- or U-lock when scanning for qualifying rows. For any qualifying rows or pages, the lock is upgraded to an X-lock before performing the update or delete. For non-qualifying rows or pages the lock is released if ISOLATION(CS) is used. For ISOLATION(RS) or ISOLATION(RR), an S-lock is retained on the rows or pages until the next commit point. Use this option to achieve higher rates of concurrency.

TARGET
DB2 combines YES and NO behavior. DB2 uses an X-lock on qualifying rows or pages of the specific table that is targeted by the update or delete statement. DB2 uses an S- or U-lock when scanning for rows or pages of other tables that are referenced by the query (for example, tables that are referenced only in the WHERE clause of the query). For non-qualifying rows or pages the lock is released if ISOLATION(CS) is used. For ISOLATION(RS) or ISOLATION(RR), an S-lock is retained on the rows or pages until the next commit point.

START IRLM CTRACE field

The value of the START IRLM CTRACE field specifies whether the IRLM component traces are to be activated when IRLM is started. The DB2-provided IRLM procedure in DSNTIJMV is tailored according to this value.

Acceptable values:	NO, YES
Default:	NO
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	none

NO IRLM starts only the low-activity subtraces EXP, INT, and XIT.

YES
IRLM starts with all its subtraces active. Starting all IRLM subtraces has a slight impact on performance demand for ECSA storage, but it improves serviceability.

IMS BMP TIMEOUT field (BMPTOUT subsystem parameter)

The BMPTOUT subsystem parameter is used to determine the amount of time that an IMS BMP connection is to wait for a lock to be released. The value of this field is multiplied by the value specified in the RESOURCE TIMEOUT field to determine the wait time.

For example, if you use the default value, an IMS BMP connection can wait for a resource for four times as long as the resource timeout value. This option gives you flexibility in tuning your system to avoid timeouts.

Acceptable values:	1 to 254
Default:	4
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM BMPTOUT

Related tasks:

Setting installation options for wait times (DB2 Performance)

DL/I BATCH TIMEOUT field (DLITOUT subsystem parameter)

The DLITOUT subsystem parameter controls the number of resource timeout values that a DL/I batch connection is to wait for a lock to be released. The resource timeout value is specified in the RESOURCE TIMEOUT field.

Acceptable values:	1 to 254
Default:	6
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DLITOUT

For example, if you use the default value of 6, a DL/I batch application can wait 6 times the resource timeout value for a resource. This option gives you flexibility in tuning your system to avoid timeouts.

Related tasks:

Setting installation options for wait times (DB2 Performance)

Related reference:

"RESOURCE TIMEOUT field (IRLMRWT subsystem parameter)" on page 387

RETAINED LOCK TIMEOUT field (RETLWAIT subsystem parameter)

When another DB2 subsystem in a data sharing group has failed, it can hold an incompatible lock on a resource. Locks that are held by failed DB2 members are called *retained locks*. The RETLWAIT subsystem parameter is used to calculate how long a transaction is to wait for a retained lock.

Acceptable values:	0 to 254
Default:	0
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RETLWAIT

This value is of importance only in a data sharing environment.

The value that you use is a multiplier that is applied to the normal timeout value of the connection. For example, imagine that the retained lock multiplier is 2. The timeout period for a call attachment connection that is waiting for a retained lock is 1 * 2 (1 for the normal CAF timeout period, 2 for the additional time that is specified for retained locks).

If you use the default value, 0, applications do not wait for incompatible retained locks. Instead, the lock request is immediately rejected, and the application receives a resource unavailable SQLCODE.

Related concepts:

Retained locks (DB2 Data Sharing Planning and Administration)

IRLM panel 2: DSNTIPJ

The entries on the DSNTIPJ panel affect several characteristics of IRLM time-sharing fields and other locking options.

The default values are adequate for most sites under ordinary conditions. DB2 and IRLM group names must start with a letter.

```
DSNTIPJ         INSTALL DB2 - IRLM PANEL 2
===> _

Enter data below:

 1  PAGE PROTECT          ===> YES      Page protect common modules (YES,NO)
 2  MAX STORAGE FOR LOCKS ===> 2160     Control block storage in MB (2048-102400)
 3  LOCKS PER TABLE(SPACE)===> 2000     Maximum before lock escalation (0-100M)
 4  LOCKS PER USER        ===> 10000    Max before resource unavailable (0-100M)
 5  DEADLOCK TIME         ===> 1        Detection interval (1-5 seconds or
                                         100-5000 milliseconds)


 For DB2 data sharing ONLY enter data below:

 6  DEADLOCK CYCLE        ===> 1        Number of LOCAL cycles before GLOBAL
 7  MEMBER IDENTIFIER     ===> 1        Member ID for this IRLM (1-255)
 8  IRLM XCF GROUP NAME   ===> DXRGROUP Name of IRLM XCF group
 9  LOCK ENTRY SIZE       ===> 2        Initial allocation, in bytes (2,4,8)
10  NUMBER OF LOCK ENTRIES===> 0        Lock table entries (0-2048)
11  DISCONNECT IRLM       ===> YES      Disconnect automatically (YES, NO)


PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 37. IRLM panel 2: DSNTIPJ

Related concepts:

"IRLM address space (IRLMPROC) storage requirements" on page 47

Performance monitoring and tuning for data sharing environments (DB2 Data Sharing Planning and Administration)

Related reference:

"IRLM panel 1: DSNTIPI" on page 386

PAGE PROTECT field

The value of the PAGE PROTECT field specifies whether IRLM is to load its common storage modules into page-protected storage.

Acceptable values:	NO, YES
Default:	YES
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

NO Indicates that common storage modules are to be loaded into CSA or ECSA without page protecting that memory.

YES Indicates that modules located in common storage are to be loaded into page-protected storage to prevent programs from overlaying the instructions. YES is recommended because it requires no additional overhead after the modules are loaded, and the protection can prevent code-overlay failures.

This is the default value.

The associated IRLM PROC parameter is PGPROT.

MAX STORAGE FOR LOCKS field

The MAX STORAGE FOR LOCKS field specifies, in MB, the maximum amount of private storage above the 2 GB bar for the IRLM lock control block structure.

Acceptable values:	2048 to 102400
Default:	2160
Update:	edit IRLM start procedure
DSNZP*xxx*	none

This value becomes the setting of the MLMT parameter for the IRLM address space procedure. The IRLM address space procedure sets the z/OS MEMLIMIT value for the address space.

Ensure that you set this value high enough so that IRLM does not reach the limit. The value that you choose should provide space for possible retained locks. IRLM only gets storage as it needs it, so choose a large value. You can also change the value dynamically by using the z/OS command `MODIFY irlmproc,SET,MLT`.

The associated IRLM PROC parameter is MLMT.

Related concepts:

"Common service area storage requirements" on page 49

Related reference:

"IRLM storage use" on page 192

LOCKS PER TABLE(SPACE) field (NUMLKTS subsystem parameter)

The NUMLKTS subsystem parameter specifies the default maximum number of page, row, or LOB locks that an application can hold simultaneously in a table or table space. If a single application exceeds the maximum number of locks in a single table or table space, lock escalation occurs.

Acceptable values:	0 to 104857600
Default:	2000
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM NUMLKTS

You can enter the number of locks as an integer or you can enter a value with a suffix of K or M. If you enter a value with a suffix of K, the number of locks is that value multiplied by 1024. If you enter a value with a suffix of M, the number of locks is that value multiplied by 1024 × 1024, or 1,048,576.

The value that you specify for this field must be less than the value that is specified for the LOCKS PER USER field (except when LOCKS PER USER is set to 0).

The value of the LOCKS PER TABLE(SPACE) field becomes the default value (SYSTEM) for the LOCKMAX clause of the SQL statements CREATE TABLESPACE and ALTER TABLESPACE. A value of 0 indicates that there is no limit to the number of data and row locks that a program can acquire.

Recommendation: Do not set the value to 0, because it can cause the IRLM to experience storage shortages.

Related tasks:

Specifying a default value for the LOCKMAX option (DB2 Performance)

Related reference:

"LOCKS PER USER field (NUMLKUS subsystem parameter)"

CREATE TABLESPACE (DB2 SQL)

ALTER TABLESPACE (DB2 SQL)

LOCKS PER USER field (NUMLKUS subsystem parameter)

The NUMLKUS subsystem parameter specifies the maximum number of page, row, or LOB locks that a single application can hold concurrently for all table spaces.

Acceptable values:	0 to 104857600
Default:	10000
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM NUMLKUS

You can enter the number of locks as an integer or you can enter a value with a suffix of K or M. If you enter a value with a suffix of K, the number of locks is that value multiplied by 1024. If you enter a value with a suffix of M, the number of locks is that value multiplied by 1024 × 1024, or 1,048,576.

The maximum number includes locks on data pages, index pages, and rows that the program acquires when it accesses table spaces. The limit applies to all table spaces that are defined with the LOCKSIZE PAGE, LOCKSIZE ROW, or LOCKSIZE ANY options. A value of 0 indicates that there is no limit to the number of data and row locks that a program can acquire.

DB2 assumes that each lock requires 540 bytes of storage. If you define referential constraints between values, you might want to select a higher value for this field.

To avoid exhausting the IRLM's storage for locks, follow these guidelines:

- Do not specify 0 or a very large value unless it is specifically required to run an application.
- Consider the design of your applications. Long-running applications, particularly those that perform row-level locking, have few or infrequent commit points, or use repeatable-read isolation may use substantial amounts of lock storage. You should perform frequent commits to release locks.

Important: These values are constraints for a single application. Each concurrent application can hold the maximum number of locks specified here.

Check panel DSNTIPC to ensure that the required storage for the IRLM does not exceed the available region size for the IRLM.

Related tasks:

Specifying the maximum number of locks that a single process can hold (DB2 Performance)

DEADLOCK TIME field

The DEADLOCK TIME field controls the amount of time for which local deadlock detection cycles are to run. A *deadlock* is a situation in which two or more

requesters are waiting for resources that are held by another requester. *Deadlock detection* is the procedure that identifies deadlocks and its participants.

Acceptable values:	1 to 5, 100 to 5000
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

DB2 interprets values between 1 and 5 as seconds and values between 100 and 5000 as milliseconds. Depending on the value that you enter, IRLM might substitute a smaller maximum value.

The associated IRLM PROC parameter is DEADLOK.

DEADLOCK CYCLE field

The value of the DEADLOCK CYCLE field specifies the number of local deadlock cycles that must expire before the IRLM does global deadlock detection processing.

Acceptable values:	1
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

The DEADLOCK CYCLE field is used only for DB2 data sharing.

The associated IRLM PROC parameter is DEADLOK.

MEMBER IDENTIFIER field

The value of the MEMBER IDENTIFIER field specifies an ID number that uniquely names this IRLM member within an IRLM data sharing group.

Acceptable values:	1 to 255
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Recommendation: Correlate the IRLM member ID with the DB2 member name. For example, for DB2 member DSN1, specify an IRLM member ID of 1.

The IRLM ID that you specify does not relate directly to the limit of IRLM members that can be in the data sharing group. That limit is determined by the current hardware limits (32). If you edit the IRLMPROC directly, you can specify a value from 1 to 255.

This option is used only for DB2 data sharing.

The associated IRLM PROC parameter is IRLMID.

Related information:

DB2 and related commands (DB2 Commands)

IRLM XCF GROUP NAME field

The value of the IRLM XCF GROUP NAME field is the name that is to be used for the IRLM group. This name must be different from the DB2 group name.

Acceptable values:	1 to 8 characters
Default:	DXRGROUP
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Recommendation: Begin this name with DXR. All members in the DB2 group must have the same IRLM XCF group name.

This option is used only for DB2 data sharing.

To avoid names that IBM uses for its cross-system coupling facility (XCF) groups, the first character must be an uppercase letter J - Z unless the name begins with DXR. Do not use SYS as the first three characters, and do not use UNDESIG as the group name.

The associated IRLM PROC parameter is IRLMGRP

LOCK ENTRY SIZE field

The value of the LOCK ENTRY SIZE field specifies the initial size, in bytes, of individual lock entries in the lock table portion of the lock structure.

Acceptable values:	2, 4, 8
Default:	2
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Recommendation: If you have less than seven members in your data sharing group, use the default value for the most efficient use of coupling lock structure space. If you have more than seven members, use a value of 4 to avoid an automatic rebuild for maximum connections.

DB2 converts the value for LOCK ENTRY SIZE to a corresponding value for the IRLM parameter MAXUSRS as shown in the table below.

Table 73. Converting lock entry size to MAXUSRS values

LOCK ENTRY SIZE	MAXUSRS value
2	7
4	23
8	32

NUMBER OF LOCK ENTRIES field

The value of the NUMBER OF LOCK ENTRIES field specifies the number of lock table entries that are to be in the coupling facility lock structure.

Acceptable values:	1 to 2048

Default:	0
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

This value must be a power of 2 in the range of 1 to 2048, with each increment representing 1,048,576 lock table entries. The default value, 0, indicates that IRLM is to determine the number of lock table entries based on the lock structure size that is specified in CFRM policy and the number of users (MAXUSRS). If you specify a value for lock structure size in CFRM policy that is greater than 2048 MB, IRLM limits the number of lock table entries to a maximum of 2048 MB. If you want to control the number of lock table entries, enter a non-zero value within the accepted range.

The number of lock table entries has a direct effect on cross-system extended services (XES) contention. You should therefore monitor this number to find the optimum values for your installation.

The associated IRLM PROC parameter is LTE.

DISCONNECT IRLM field

The value of the DISCONNECT IRLM field determines whether IRLM is to automatically disconnect from the data sharing group when DB2 is not identified to it.

Acceptable values:	YES, NO
Default:	YES
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

YES Causes IRLM to disconnect from the data sharing group when DB2 is stopped normally or it stops as the result of a DB2 failure.

If you specify YES for the DISCONNECT IRLM field and YES for the AUTO START field on panel DSNTIPI, stopping IRLM does not require manual intervention.

NO IRLM remains connected to the data sharing group even when DB2 is stopped. In this case, you must explicitly stop IRLM to bring it down.

This option has less impact on other systems when a DB2 fails because z/OS is not required to perform certain recovery actions that it normally performs when IRLM comes down. Specifying NO can also mean that DB2 restarts more quickly after a DB2 normal or abnormal termination because it does not have to wait for IRLM to rejoin the IRLM data sharing group.

The associated IRLM PROC parameter is SCOPE.

DB2 utilities parameters panel 1: DSNTIP6

The fields on the DSNTIP6 panel configure the default behavior of enhancements to the BACKUP SYSTEM, RESTORE SYSTEM, RECOVER, and other utilities.

```
DSNTIP6            INSTALL DB2 - DB2 UTILITIES PARAMETERS 1
===>

Enter system-level backup options for RESTORE SYSTEM and RECOVER below:
 1 SYSTEM-LEVEL BACKUPS  ===> NO        As a recovery base: NO or YES
 2 RESTORE/RECOVER       ===> NO        From dump: NO or YES
 3 DUMP CLASS NAME       ===>           For RESTORE/RECOVER from dump
 4 MAXIMUM TAPE UNITS    ===> NOLIMIT   For RESTORE SYSTEM: NOLIMIT or 1-255
Enter default settings for the DB2 Utilities backup options below:
 5 FAST REPLICATION      ===> REQUIRED  DSScopy replication type for CHECK
                                        utilities: PREFERRED or REQUIRED
 6 FAST RESTORE          ===> PREFERRED For RECOVERY: NONE, PREFERRED, REQUIRED
 7 FLASHCOPY PPRC        ===> REQUIRED  FLASHCOPY peer to peer remote copy:
                                        blank, NONE, PREFERRED, or REQUIRED
 8 DEFAULT TEMPLATE      ===> HLQ.&DB..&SN..N&DSNUM..&UQ.

 9 COPY                  ===> NO        Use FLASHCOPY defaults for COPY
10 LOAD                  ===> NO        Use FLASHCOPY defaults for LOAD
11 REORG TABLESPACE      ===> NO        Use FLASHCOPY defaults for REORG TS
12 REBUILD INDEX         ===> NO        Use FLASHCOPY defaults for REBUILD IX
13 REORG INDEX           ===> NO        Use FLASHCOPY defaults for REORG IX

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 38. DB2 utilities parameters panel 1: DSNTIP6

SYSTEM-LEVEL BACKUPS field (SYSTEM_LEVEL_BACKUPS subsystem parameter)

The RECOVER utility uses image copies and concurrent copies for object-level recoveries. The SYSTEM_LEVEL_BACKUPS subsystem parameter specifies whether the RECOVER utility is to also use system-level backups as a recovery base for object-level recoveries.

Acceptable values:	NO, YES
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSTEM_LEVEL_BACKUPS

Specify the default of NO if you do not take system-level backups with the BACKUP SYSTEM utility. Specify YES if you take system-level backups with the BACKUP SYSTEM utility.

Related reference:

RECOVER (DB2 Utilities)

RESTORE/RECOVER field (RESTORE_ RECOVER_FROMDUMP subsystem parameter)

The RESTORE_ RECOVER_FROMDUMP subsystem parameter controls whether the system-level backup for the RESTORE SYSTEM and the RECOVER utilities is to occur from the dump on tape.

Acceptable values:	YES or NO
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RESTORE_ RECOVER_FROMDUMP

YES System-level backup for the RESTORE SYSTEM and the RECOVER utilities occurs from the dump on tape.

NO System-level backup for the RESTORE SYSTEM and the RECOVER utilities occurs from the disk copy of the system-level backup.

You can override the setting of RESTORE/RECOVER by executing the RESTORE SYSTEM utility statement or the RECOVER utility statement with the FROMDUMP keyword.

DUMP CLASS NAME field (UTILS_DUMP_CLASS_NAME subsystem parameter)

A system-level backup can occur from a dump on tape or from a disk copy. For backups that occur from the dump on tape, the UTILS_DUMP_CLASS_NAME subsystem parameter specifies the name of the DFSMShsm dump class that is to be used.

Acceptable values:	blank or a valid DFSMShsm dump class name not exceeding 8 characters in length
Default:	blank
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM UTILS_DUMP_CLASS_NAME

This dump class is to be used by the RECOVER utility and the RESTORE SYSTEM utility to restore objects from a system-level backup that has been dumped to tape. The setting for this field is applicable only when you specify YES in the RESTORE/RECOVER field on panel DSNTIP6. You can override the setting for DUMP CLASS NAME by executing the RESTORE SYSTEM utility statement or the RECOVER utility statement with the DUMPCLASS keyword.

Related concepts:

Dump Tasks (z/OS DFSMShsm Storage Administration Reference)

MAXIMUM TAPE UNITS field (RESTORE_TAPEUNITS subsystem parameter)

When the RESTORE SYSTEM utility restores a system-level backup that has been dumped to tape, it allocates a number of tape units or tape drives. The RESTORE_TAPEUNITS subsystem parameter specifies the maximum number of tape units or tape drives that the RESTORE SYSTEM utility can allocate.

Acceptable values:	NOLIMIT or 1 - 255
Default:	NOLIMIT
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RESTORE_TAPEUNITS

The default value, NOLIMIT, means that the RESTORE SYSTEM utility will allocate as many tape units as necessary to restore the system-level backup. You can override the setting for MAXIMUM TAPE UNITS by executing the RESTORE SYSTEM utility statement with the TAPEUNITS keyword.

FAST REPLICATION field (CHECK_FASTREPLICATION subsystem parameter)

The CHECK_FASTREPLICATION parameter specifies the type of replication that DSS COPY uses to copy objects to shadow data sets when it is invoked by the DB2 CHECK utilities. The CHECK utilities can stipulate fast replication as PREFERRED or REQUIRED.

Acceptable values:	PREFERRED, REQUIRED
Default:	REQUIRED
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM.CHECK_FASTREPLICATION

This parameter applies to the CHECK utilities in all DB2 modes (conversion mode through new-function mode)

PREFERRED
The CHECK utilities direct DSS COPY to use fast replication only if FlashCopy® support is available.

REQUIRED
The CHECK utilities force DSS COPY to use fast replication, ensuring that object copies occur as quickly as possible. This reduces the opportunity for resource contention and unavailability. This option causes CHECK utilities to fail if FlashCopy cannot be used.

Related concepts:

Subsystem parameters for refining DFSMSdss COPY operation with utilities (DB2 Utilities)

Related tasks:

Checking data consistency with the CHECK utilities (Managing Security)

FAST RESTORE field (REC_FASTREPLICATION subsystem parameter)

The REC_FASTREPLICATION parameter specifies whether the RECOVER utility should use FlashCopy to recover from a FlashCopy image copy. This parameter applies only to Version 10 new-function mode.

Acceptable values:	NONE, PREFERRED, REQUIRED
Default:	PREFERRED
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM.REC_FASTREPLICATION

NONE
The RECOVER utility uses standard I/O to restore a FlashCopy image copy.

PREFERRED
The RECOVER utility uses FlashCopy to recover from a FlashCopy image copy if FlashCopy support is available.

REQUIRED
The RECOVER utility forces the use of FlashCopy to recover from a FlashCopy image copy to ensure that recovery occurs as quickly as possible. This option causes RECOVERY to fail if FlashCopy cannot be used.

If you use BACKUP SYSTEM to create system-level backups, using FlashCopy for a restore might cause BACKUP SYSTEM to fail because bi-directional FlashCopy is not supported.

Related concepts:

Subsystem parameters for refining DFSMSdss COPY operation with utilities (DB2 Utilities)

FLASHCOPY PPRC field (FLASHCOPY_PPRC subsystem parameter)

The FLASHCOPY_PPRC subsystem parameter specifies the behavior for DFSMSdss FlashCopy requests when the target disk storage volume is the primary device in a peer-to-peer remote copy (Metro Mirror) relationship.

Specifically, FLASHCOPY_PPRC determines the following:

- Whether DFSMSdss preserves mirroring while processing a DB2 utilities request
- Whether the target device pair is allowed to go to duplex pending state

The FLASHCOPY_PPRC parameter applies to the CHECK DATA, CHECK INDEX, and CHECK LOB utilities in all DB2 modes (conversion mode through new-function mode). The parameter also applies to the COPY, REORG TABLESPACE, REORG INDEX, REBUILD INDEX, LOAD, and RECOVER utilities in new-function mode.

Important: Set this field to blank if PPRC is not available on the subsystem.

Acceptable values:	NONE, PREFERRED, REQUIRED, blank
Default:	REQUIRED
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM.FLASHCOPY_PPRC

NONE
DFSMSdss does not preserve mirroring while processing a DB2 utilities request, even if all the configuration requirements for preserving the mirror operation are met.

PREFERRED
DFSMSdss allows a volume pair to go into duplex pending state while processing a DB2 utilities request only if it is required.

REQUIRED
DFSMSdss does not permit a PPRC volume pair to go into duplex pending state while processing a DB2 utilities request.

blank No preserve mirror command is passed to DFSMSdss.

Related concepts:

Subsystem parameters for refining DFSMSdss COPY operation with utilities (DB2 Utilities)

DEFAULT TEMPLATE field (FCCOPYDDN subsystem parameter)

The FCCOPYDDN subsystem parameter defines the default value that is to be used for the FCCOPYDDN parameter of the FLASHCOPY option of DB2 utilities

control statements. This parameter applies to the COPY, LOAD, REBUILD INDEX, REORG INDEX, and REORG TABLESPACE utilities.

FCCOPYDDN specifies the template for deriving the data set name for the FlashCopy image copy.

FlashCopy technology can be used to create VSAM copies of objects almost instantaneously. It creates a separate data set for each piece of the object that is being copied. FlashCopy technology can also be used to recover the object if necessary, which reduces recovery time and therefore provides greater availability.

Important: You must replace "HLQ" in the default value with a high-level qualifier that is appropriate for your subsystem.

Acceptable values:	valid DB2 utilities template
Default:	HLQ.&DB..&SN..N&DSNUM..&UQ.
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FCCOPYDDN

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

If you assign the FCCOPYDDN setting directly in DSNTIJUZ, you need to repeat each ampersand (&) character to conform with IBM High-Level Assembler rules. For example, you would need to assign the default value above as FCCOPYDDN=HLQ.&&DB..&&SN..N&&DSNUM..&&UQ. (notice the trailing period).

COPY field (FLASHCOPY_COPY subsystem parameter)

The value of the FLASHCOPY_COPY parameter specifies whether the FLASHCOPY option of the COPY utility is to be used by default.

Acceptable values:	NO, YES
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FLASHCOPY_COPY

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

NO Specifies that the FLASHCOPY option of the COPY utility is not used by default.

YES
Specifies that the COPY utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement. Specify YES only if the DB2 data sets are on FlashCopy Version 2 DASD.

LOAD field (FLASHCOPY_LOAD subsystem parameter)

The value of the FLASHCOPY_LOAD subsystem parameter specifies whether the FLASHCOPY option of the LOAD utility is to be used by default.

Acceptable values: NO, YES

Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FLASHCOPY_LOAD

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

NO Specifies that the FLASHCOPY option of the LOAD utility is not used by default.

YES
Specifies that the LOAD utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement. Specify YES only if the DB2 data sets are on FlashCopy Version 2 DASD.

REORG TABLESPACE field (FLASHCOPY_REORG_TS subsystem parameter)

The value of the FLASHCOPY_REORG_TS subsystem parameter specifies whether the FLASHCOPY option of the REORG TABLESPACE utility is to be used by default.

Acceptable values:	NO, YES
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FLASHCOPY_REORG_TS

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

NO Specifies that the FLASHCOPY option of the REORG TABLESPACE utility is not used by default.

YES
Specifies that the REORG TABLESPACE utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement. Specify YES only if the DB2 data sets are on FlashCopy Version 2 DASD.

REBUILD INDEX field (FLASHCOPY_REBUILD_INDEX subsystem parameter)

The value of the FLASHCOPY_REBUILD_INDEX subsystem parameter specifies whether the FLASHCOPY option of the REBUILD INDEX utility is to be used by default.

Acceptable values:	NO, YES
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FLASHCOPY_REBUILD_INDEX

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

NO Specifies that the FLASHCOPY option of the REBUILD INDEX utility is not used by default.

YES
Specifies that the REBUILD INDEX utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement. Specify YES only if the DB2 data sets are on FlashCopy Version 2 DASD.

REORG INDEX field (FLASHCOPY_REORG_INDEX subsystem parameter)

The value of the FLASHCOPY_REORG_INDEX subsystem parameter specifies whether the FLASHCOPY option of the REORG INDEX utility is to be used by default.

Acceptable values:	NO, YES
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM FLASHCOPY_REORG_INDEX

This parameter applies to new-function mode only. The parameter is ignored in conversion mode (from both Version 8 and Version 9.1) and enabling-new-function mode (from both Version 8 and Version 9.1).

NO Specifies that the FLASHCOPY option of the REORG INDEX utility is not used by default.

YES
Specifies that the REORG INDEX utility uses FlashCopy technology when the FLASHCOPY option is not specified in the control statement. Specify YES only if the DB2 data sets are on FlashCopy Version 2 DASD.

DB2 utilities parameters panel 2: DSNTIP61

The fields on the DSNTIP61 panel are used to configure the default behavior of various DB2 utilities.

```
DSNTIP61            INSTALL DB2 - DB2 UTILITIES PARAMETERS 2
===>

Enter other DB2 Utilities options below:
 1 TEMP DS UNIT NAME     ===> SYSDA     Device for temporary utility data sets
 2 UTIL TEMP STORCLAS    ===>           For temporary shadow data sets
                                        by utilities
 3 STATISTICS HISTORY    ===> NONE      Default for collection of stats history
 4 STATISTICS ROLLUP     ===> YES       Allow statistics aggregation: NO or YES
 5 UTILITY TIMEOUT       ===> 6         Utility wait time multiplier
 6 UT SORT DS ALLOCATION ===> YES       Predictable sort disk space allocation
 7 IGNORE SORTNUM STMT   ===> NO        Ignore SORTNUM keyword in UT stmt
 8 SET CHECK PENDING     ===> NO        Set inconsistent objects in CHECK
                                        PENDING status: NO or YES
 9 UT DB2 SORT USE       ===> ENABLE    ENABLE/DISABLE utility use of DB2 Sort




PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 39. DB2 utilities parameters panel 2: DSNTIP61

TEMP DS UNIT NAME field (VOLTDEVT subsystem parameter)

The VOLTDEVT subsystem parameter specifies the device type or unit name that is to be used by the DB2 COPY CONCURRENT and CHECK utilities for dynamically allocating temporary data sets. Specifying a value for VOLTDEVT removes the need to specify the FILTERDDN DD in the utility job.

Acceptable values:	valid device type or unit name
Default:	SYSDA
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM VOLTDEVT

The temporary data sets are deleted when the utility execution is complete.

UTIL TEMP STORCLAS field (UTIL_TEMP_STORCLAS subsystem parameter)

The UTIL_TEMP_STORCLAS subsystem parameter specifies the storage class that the CHECK INDEX, CHECK DATA, and CHECK LOB utilities are to use when allocating temporary shadow data sets. (These utilities allocate shadow data sets when you specify the SHRLEVEL CHANGE option.)

Acceptable values:	blank or a valid SMS management class name
Default:	blank
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM UTIL_TEMP_STORCLAS

The default value of blank indicates that the shadow data sets are to be defined in the same storage class as the production page set.

Recommendation: Specifying an explicit storage class for this subsystem parameter can improve performance, especially when the system uses paired disk volumes for peer to peer remote copy.

In a data sharing group, this parameter has group scope.

STATISTICS HISTORY field (STATHIST subsystem parameter)

The STATHIST subsystem parameter specifies which inserts and updates are to be recorded in catalog history tables.

Acceptable values:	SPACE, NONE, ALL, ACCESSPATH
Default:	NONE
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATHIST

SPACE
All inserts and updates that DB2 makes to space-related catalog statistics are recorded.

NONE
None of the changes that DB2 makes in the catalog are recorded. NONE is the default value.

ALL All inserts and updates that DB2 makes in the catalog are recorded.

ACCESSPATH
All inserts and updates that DB2 makes to ACCESSPATH-related catalog statistics are recorded.

Related tasks:

Collecting history statistics (DB2 Performance)

Collecting statistics history (DB2 Utilities)

Related reference:

History statistics (DB2 Performance)

STATISTICS ROLLUP field (STATROLL subsystem parameter)

The STATROLL subsystem parameter specifies whether the RUNSTATS utility is to aggregate the partition-level statistics, even though some parts might not contain data.

Acceptable values:	YES, NO
Default:	YES
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATROLL

For DB2 subsystems that have large partitioned table spaces and indexes, it is recommended that you specify YES for STATISTICS ROLLUP. Specifying YES for STATISTICS ROLLUP enables the aggregation of partition-level statistics and helps the optimizer to choose a better access path.

UTILITY TIMEOUT field (UTIMOUT subsystem parameter)

The UTIMOUT subsystem parameter specifies how long, in number of resource values, that a utility or utility command is to wait for a resource. The utility or utility command waits until a lock or all claims on a resource of a particular claim class is released.

Acceptable values:	1 to 254
Default:	6
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM UTIMOUT

For example, if you use the default value of six, a utility can wait six times longer than an SQL application for a resource. This option allows utilities to wait longer than SQL applications to access a resource. For more information about setting the resource timeout for SQL applications, see the discussion of the RESOURCE TIMEOUT field on DSNTIPI.

The value of UTILITY TIMEOUT is used as the default value for the RETRY parameter of DB2 Utilities such as CHECK INDEX and online REBUILD INDEX.

Related tasks:

Specifying the amount of inactive time before a timeout (DB2 Performance)

Related reference:

"RESOURCE TIMEOUT field (IRLMRWT subsystem parameter)" on page 387

UT SORT DATA SET ALLOCATION field (UTSORTAL subsystem parameter)

The UTSORTAL subsystem parameter specifies whether DB2 is to use real-time statistics to determine the sort work data set sizes if real-time statistics data is available.

Acceptable values:	YES or NO
Default:	YES
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM UTSORTAL

YES

DB2 uses real-time statistics to determine the sort work data set sizes if real-time statistics data is available. If you specify YES and real-time statistics are unavailable, DB2 utilities that invoke sorts use a space prediction algorithm for dynamically allocated sort work data sets. These utilities are CHECK, LOAD, REBUILD, REORG, and RUNSTATS.

The following table illustrates the result of using UTSORTAL=YES in combination with the IGNSORTN subsystem parameter and real-time statistics.

Table 74. Results of using UTSORTAL=YES in combination with IGNSORTN and real-time statistics

UTSORTAL setting	IGNSORTN setting	Real-time statistics available?	Result
YES	NO	YES	DB2 uses the real-time statistics to estimate the sizes of the sort work data sets. The SORTNUM clause of the utility control statement is used to estimate the number of temporary data sets that are to be dynamically allocated by the sort program.
YES	NO	NO	DB2 uses catalog information to estimate the sizes of the sort work data sets. The SORTNUM clause of the utility control statement is used to estimate the number of temporary data sets that are to be dynamically allocated by the sort program.

Table 74. Results of using UTSORTAL=YES in combination with IGNSORTN and real-time statistics (continued)

UTSORTAL setting	IGNSORTN setting	Real-time statistics available?	Result
YES	YES	YES	DB2 uses the real-time statistics to estimate the sizes of the sort work data sets. DB2 also determines the number of temporary data sets that are to be dynamically allocated by the sort program, regardless of the SORTNUM clause.
YES	YES	NO	DB2 uses catalog information to estimate the sizes of the sort work data sets. DB2 also determines the number of temporary data sets that are to be dynamically allocated by the sort program, regardless of the SORTNUM clause.

NO DB2 does not use real-time statistics to determine the sort work data set sizes. If you specify NO, you either need to specify the SORTNUM clause in the control statement when you run these utilities or allocate sort work data sets explicitly using JCL.

IGNORE SORTNUM STAT field (IGNSORTN subsystem parameter)

The IGNSORTN subsystem parameter determines whether occurrences of the SORTNUM clause in utility control statements are to be ignored. The value of the IGNORE SORTNUM STAT field is meaningful only when YES is specified in the UT SORT DATA SET ALLOCATION field.

Acceptable values:	YES or NO
Default:	NO
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM IGNSORTN

YES Indicates that DB2 is to ignore the use of the SORTNUM clause in utility control statements.

NO Specifies that occurrences of the SORTNUM clause are to be honored.

SET CHECK PENDING field (CHECK_SETCHKP subsystem parameter)

The CHECK_SETCHKP subsystem parameter specifies whether the CHECK DATA and CHECK LOB utilities are to place inconsistent objects in CHECK PENDING status.

Acceptable values:	YES or NO
Default:	NO
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CHECK_SETCHKP

When the CHECK DATA and CHECK LOB utilities detect an inconsistency in an object, they write diagnostic messages and end with a return code. The return code matches the return code that is associated with the diagnostic messages. If several messages have been issued, the highest return code is used.

YES The CHECK DATA and CHECK LOB utilities place inconsistent objects in CHECK PENDING status. If an object is in CHECK PENDING status, it is unavailable for further processing until you correct the inconsistency and rerun the CHECK utility. There is no impact if the CHECK utilities are running in mode SHRLEVEL CHANGE because the utilities are running on a shadow copy and a restrictive state is not set in this mode.

NO The CHECK DATA and CHECK LOB utilities do not place objects in CHECK PENDING status if an inconsistency is detected. If the object was already in CHECK PENDING status, it remains in that status.

To change the CHECK_SETCHKP parameter without using the panel interface:

1. Edit member DSN6SPRM of the *prefix*.SDSNMACS library and locate the entry for CHECK_SETCHKP.
2. Change the setting for CHECK_SETCHKP and save your changes.
3. Run the first two steps of your customized copy of job DSNTIJUZ to rebuild your DB2 system parameter (DSNZP*xxx*) module.
4. After rebuilding the DB2 system parameter module, refresh the system parameter module by issuing the SET SYSPARM command.

UT DB2 SORT USE field (DB2SORT subsystem parameter)

The DB2SORT subsystem parameter specifies whether DB2 utilities are to use DB2 Sort instead of DFSORT for utility sort processing when DB2 Sort is installed.

Acceptable values:	ENABLE, DISABLE
Default:	ENABLE
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DB2SORT

ENABLE
DB2 utilities use DB2 Sort when it is installed. If DB2 Sort is not installed, DFSORT is used instead.

DISABLE
DB2 utilities use DFSORT for utility sort processing, regardless of whether DB2 Sort is installed.

Protection panel: DSNTIPP

The entries on the DSNTIPP panel are related to security matters.

```
DSNTIPP         INSTALL DB2 - PROTECTION
 ===> _

 Enter data below:

  1  ARCHIVE LOG RACF   ===> NO        RACF protect archive log data sets
  2  USE PROTECTION     ===> YES       DB2 authorization enabled. YES or NO
  3  PLAN AUTH CACHE    ===> 3072      Size in bytes per plan (0 - 4096)
  4  PACKAGE AUTH CACHE ===> 5M        Global - size in bytes (0-10M)
  5  ROUTINE AUTH CACHE ===> 5M        Global - size in bytes (0-10M)
  6  AUTH EXIT LIMIT    ===> 10        Access control exit shutdown threshold

 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 40. Protection panel: DSNTIPP

ARCHIVE LOG RACF field (PROTECT subsystem parameter)

The PROTECT subsystem parameter specifies whether archive log data sets are to be protected with individual profiles with RACF when they are created.

Acceptable values:	YES, NO
Default:	NO
Update:	see “Protection panel: DSNTIPP”; not during migration
DSNZP*xxx*:	DSN6ARVP PROTECT

If you specify YES, RACF protection must be active for DB2. However, a value of YES also means that you cannot use RACF generic profiles for archive log data sets. In addition, RACF class TAPEVOL must be active if your archive log is on tape. Otherwise, the offload fails.

Related concepts:

Managing access through RACF (Managing Security)

USE PROTECTION field (AUTH subsystem parameter)

The AUTH subsystem parameter controls whether DB2 is to check authorizations.

Acceptable values:	YES, NO
Default:	YES
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AUTH
Security parameter:	Yes

YES Enables authorization checking in DB2.

If you specified YES for the value of CACHE DYNAMIC SQL on panel DSNTIP8, you must also specify YES for this value.

Recommendation: Specify YES.

NO Disables all authorization checking in DB2 and disables the GRANT statement. In this case, every privilege is granted to PUBLIC.

Note: This is a security-related parameter. If it is set to NO, individual grants cannot be performed and all privileges are granted to PUBLIC, which allows any user to access user data.

PLAN AUTH CACHE field (AUTHCACH subsystem parameter)

The AUTHCACH subsystem parameter specifies the size (in bytes per plan) of the authorization cache that is to be used if no CACHESIZE is specified on the BIND PLAN subcommand.

Acceptable values:	0 to 4096 in multiples of 256
Default:	3072
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AUTHCACH
Security parameter:	Yes

0 Specifies that no authorization cache is to be used.

256 to 4096 (integers in multiples of 256)
For an authorization cache, you need 32 bytes of overhead plus (8 bytes of storage times the number of concurrent users).

Note: This is a security-related parameter.

Related tasks:

Determining the authorization cache size for plans (DB2 Application programming and SQL)

Related reference:

BIND PLAN (DSN) (DB2 Commands)

PACKAGE AUTH CACHE field (CACHEPAC subsystem parameter)

The CACHEPAC subsystem parameter determines the amount of storage in bytes to allocate for caching package authorization information for all packages on this DB2 member. The cache is stored in the DSN1DBM1 address space.

Acceptable values:	0 to 10M
Default:	5M
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHEPAC
Security parameter:	Yes

Note: This is a security-related parameter. When RACF authorization is used and this parameter is set, the changes that are made in RACF profiles are not immediately reflected. This setting might result in a user (cached AUTHID) being able to run a package.

ROUTINE AUTH CACHE field (CACHERAC subsystem parameter)

The CACHERAC subsystem parameter determines the amount of storage in bytes to allocate for caching routine authorization information for all routines on this DB2 member. Routines include stored procedures and user-defined functions.

Acceptable values:	0 to 10M

Default:	5M
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHERAC
Security parameter:	Yes

Note: This is a security-related parameter. When RACF authorization is used and this parameter is set, the changes that are made in RACF profiles are not immediately reflected. This setting might result in a user (cached AUTHID) being able to invoke a routine or function.

AUTH EXIT LIMIT field (AEXITLIM subsystem parameter)

The AEXITLIM subsystem parameter controls the number of abends of the DB2 access control authorization exit routine that are to be tolerated before it is shut down.

Acceptable values:	0 to 32767
Default:	10
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AEXITLIM

After the exit routine shuts down, it can be reactivated only by restarting DB2. A very low setting for the AUTH EXIT LIMIT field might cause the exit routine to shut down in response to routine abends such as timeouts. A very high setting, on the other hand, may mask a problem with the exit routine environment that can result in degraded DB2 or system performance.

Protection panel 2: DSNTIPP1

Use the DSNTIPP1 panel to specify security options, including the authorization IDs that are required for system administrators and system operators, and the authorization IDs or roles that are required for security administrators. You can also specify the authorization IDs that are to be used to install, configure, and validate DB2-supplied routines.

Because space on this panel is limited, some input fields are not wide enough to display the entire contents of the field at one time. These fields are defined as ISPF scrollable fields, which let you see the field contents. A less-than symbol (<) or a greater-than (>) symbol denotes a scrollable field. A > symbol indicates that the field can be scrolled to the right, and a < symbol indicates that the field can be scrolled to the left. You can position your cursor in the field and use ISPF scroll commands LEFT and RIGHT to scroll through the field. To see the entire contents of the field at one time, type EXPAND in the command field, position your cursor in the scrollable field, and press Enter. (Alternatively, you can assign your PF keys to be the LEFT, RIGHT, and EXPAND commands.)

```
DSNTIPP1         INSTALL DB2 - PROTECTION PANEL 2
 ===> _

 Enter data below:

  1  SYSTEM ADMIN 1    ===> SYSADM      Authid of system administrator
  2  SYSTEM ADMIN 2    ===> SYSADM      Authid of system administrator
  3  SYSTEM OPERATOR 1 ===> SYSOPR      Authid of system operator
  4  SYSTEM OPERATOR 2 ===> SYSOPR      Authid of system operator
  5  SECURITY ADMIN 1  ===> SECADM    > Authid or role of security administrator
  6  SEC ADMIN 1 TYPE  ===> AUTHID      Security administrator type. AUTHID, ROLE
  7  SECURITY ADMIN 2  ===> SECADM    > Authid or role of security administrator
  8  SEC ADMIN 2 TYPE  ===> AUTHID      Security administrator type. AUTHID,ROLE
  9  SEPARATE SECURITY ===> NO          SYSADM/SYSCTRL can perform GRANT/REVOKE
 10  UNKNOWN AUTHID    ===> IBMUSER     Authid of default (unknown) user
 11  RESOURCE AUTHID   ===> SYSIBM      Authid of Resource Limit Table creator
 12  BIND NEW PACKAGE  ===> BINDADD     Authority required: BINDADD or BIND
 13  DBADM CREATE AUTH ===> NO          DBA can create views/aliases for others
 14  REVOKE DEP PRIV   ===> SQLSTMT     Include dependent privileges on REVOKE
                                        (NO, YES, or SQLSTMT)
 15  ROUTINES CREATOR  ===> SYSADM      Authid to create and bind DB2 routines
 16  SEC DEF CREATOR   ===> SYSADM      Authid for routines w/ SECURITY DEFINER
                                        (must be a primary authorization ID)
 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 41. Protection panel 2: DSNTIPP1

SYSTEM ADMIN 1 field (SYSADM subsystem parameter)

The SYSADM subsystem parameter specifies the first of two authorization IDs that are to have installation SYSADM authority.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSADM
Security parameter:	Yes

The two users with installation SYSADM authority are permitted access to DB2 in all cases.

Note: This is a security-related parameter. The authorization ID that is specified for this parameter can manage the DB2 subsystem even when the DB2 catalog is unavailable. This authorization ID can access all user data and can run any application.

Related reference:

Installation SYSADM (Managing Security)

SYSTEM ADMIN 2 field (SYSADM2 subsystem parameter)

The SYSADM2 subsystem parameter specifies the second of two authorization IDs that are to have installation SYSADM authority.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSADM2

Security parameter:	Yes

If you leave this field blank, the value is set to the value of the SYSTEM ADMIN 1 field.

Note: This is a security-related parameter. The authorization ID that is specified for this parameter can manage the DB2 subsystem even when the DB2 catalog is unavailable. This authorization ID can access all user data and can run any application.

SYSTEM OPERATOR 1 field (SYSOPR1 subsystem parameter)

The SYSOPR1 subsystem parameter specifies the first of two authorization IDs that are to have installation SYSOPR authority.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSOPR
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSOPR1
Security parameter:	Yes

The two users with installation SYSOPR authority are permitted access to DB2 even if the DB2 catalog is unavailable.

If you leave this field blank, the value is set to the value of the SYSTEM ADMIN 1 field.

Recommendation: Set the value of this field or the value of the SYSTEM OPERATOR 2 field to SYSOPR. Doing so ensures that DB2 commands that are issued from the console can be processed correctly when the DB2 catalog is unavailable.

Note: This is a security-related parameter. The authorization ID that is specified for this parameter can issue most of the DB2 commands such as BIND QUERY, CANCEL THREAD, and STOP DB2. This ability might provide the authorization ID with information about user data. Also, the authorization ID can run the DSN1SDMP utility and terminate any utility job.

Related reference:

Installation SYSOPR (Managing Security)

SYSTEM OPERATOR 2 field (SYSOPR2 subsystem parameter)

The SYSOPR2 subsystem parameter specifies the second of two system operators that are to have installation SYSOPR authority.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSOPR
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSOPR2
Security parameter:	Yes

If you leave this field blank, the value is set to the value of the SYSTEM OPERATOR 1 field.

Recommendation: Set the value of this field or the value of the SYSTEM OPERATOR 1 field to SYSOPR. Doing so ensures that DB2 commands that are issued from the console can be processed correctly when the DB2 catalog is unavailable.

Note: This is a security-related parameter. The authorization ID that is specified for this parameter can issue most of the DB2 commands such as BIND QUERY, CANCEL THREAD, and STOP DB2. This ability might provide the authorization ID with information about user data. Also, the authorization ID can run the DSN1SDMP utility and terminate any utility job.

SECURITY ADMIN 1 field (SECADM1 subsystem parameter)

The SECADM1 subsystem parameter specifies the first of two authorization IDs or roles that are to have DB2 security administrator authority. In the SEC ADMIN 1 TYPE field, specify whether this entry is an authorization ID or a role.

Acceptable values:	If SEC ADMIN 1 TYPE is AUTHID: 1 to 8 characters, starting with an alphabetic character. If SEC ADMIN 1 TYPE is ROLE: an ordinary SQL identifier (up to 128 bytes) that designates a role. The role identifier cannot begin with "SYS" and cannot be ACCESSCTRL, DATAACCESS, DBADM, DBCTRL, DBMAINT, NONE, NULL, PACKADM, PUBLIC, SECADM, or SQLADM.
Default:	SECADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*	DSN6SPRM SECADM1
Security parameter:	Yes

If you want to separate DB2 security administrator duties from system administrator duties for this subsystem, set at least one SECADM system parameter to an authorization ID, or create the necessary trusted contexts and roles before setting the SEPARATE SECURITY field to YES. If you specify YES for SEPARATE SECURITY, system administrator authority can no longer be used to perform security tasks, and the SECADM authority is required to manage security objects such as trusted contexts and roles. If both SECADM system parameters are set to roles and those roles have not been created, no one will have the authority to manage security objects..

If the access control authorization exit routine (DSNX@XAC) is active, then the exit routine is called to check for SECADM authorization and this system parameter is not checked.

Note: This is a security-related parameter. A user that has SECADM authority can manage security-related objects such as trusted contexts, roles, and column masks. The user can also grant privileges and revoke privileges that are granted by others.

Related concepts:

Access control authorization exit routine (Managing Security)

Related tasks:

"Installation step 9: Establish subsystem security (optional)" on page 520

Related reference:

"SEPARATE SECURITY field (SEPARATE_SECURITY subsystem parameter)" on page 417

Related information:

Authorization IDs and roles (Managing Security)

SEC ADMIN 1 TYPE field (SECADM1_TYPE subsystem parameter)

The SECADM1_TYPE subsystem parameter specifies whether the entry in the SECURITY ADMIN 1 field is an authorization ID or a role.

Acceptable values:	AUTHID, ROLE
Default:	AUTHID
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SECADM1_TYPE
Security parameter:	Yes

Note: This is a security-related parameter. The value specifies whether the SECADM1 parameter is an authorization ID or a role. Changing the value of this field changes the SECADM1 parameter from an authorization ID to a role, or from a role to an authorization ID.

Related information:

Authorization IDs and roles (Managing Security)

Managing access through trusted contexts (Managing Security)

SECURITY ADMIN 2 field (SECADM2 subsystem parameter)

The SECADM2 subsystem parameter specifies the second of two authorization IDs or roles that are to have DB2 security administrator authority. In the SEC ADMIN 2 TYPE field, specify whether this entry is an authorization ID or a role.

Acceptable values:	If SEC ADMIN 2 TYPE is AUTHID: 1 to 8 characters, starting with an alphabetic character. If SEC ADMIN 2 TYPE is ROLE: an ordinary SQL identifier (up to 128 characters) that designates a role. The role identifier cannot begin with "SYS" and cannot be ACCESSCTRL, DATAACCESS, DBADM, DBCTRL, DBMAINT, NONE, NULL, PACKADM, PUBLIC, SECADM, or SQLADM.
Default:	SECADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SECADM2
Security parameter:	Yes

If you leave this field blank, the value is set to the value of the SECURITY ADMIN 1 field.

If you want to separate DB2 security administrator duties from system administrator duties for this subsystem, set at least one SECADM system parameter to an authorization ID, or create the necessary trusted contexts and roles before setting the SEPARATE SECURITY field to YES. If you specify YES for SEPARATE SECURITY, system administrator authority can no longer be used to perform security tasks, and the SECADM authority is required to manage security objects such as trusted contexts and roles. If both SECADM system parameters are set to roles and those roles have not been created, no one will have the authority to manage security objects.

If the access control authorization exit (DSNX@XAC) is active, then the exit is called to check for SECADM authorization and this parameter is not checked.

Note: This is a security-related parameter. A user that has SECADM authority can manage security-related objects such as trusted contexts, roles, and column masks. The user can also grant privileges and revoke privileges that are granted by others.

Related concepts:

Access control authorization exit routine (Managing Security)

Related tasks:

"Installation step 9: Establish subsystem security (optional)" on page 520

Related information:

Authorization IDs and roles (Managing Security)

SEC ADMIN 2 TYPE field (SECADM2_TYPE subsystem parameter)

The SECADM2_TYPE subsystem parameter specifies whether the entry in the SECURITY ADMIN 2 field is an authorization ID or a role.

Acceptable values:	AUTHID, ROLE
Default:	AUTHID
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SECADM2_TYPE
Security parameter:	Yes

Note: This is a security-related parameter. The value specifies whether the SECADM2 parameter is an authorization ID or a role. Changing the value of this field changes the SECADM2 parameter from an authorization ID to a role, or from a role to an authorization ID.

Related information:

Authorization IDs and roles (Managing Security)

Managing access through trusted contexts (Managing Security)

SEPARATE SECURITY field (SEPARATE_SECURITY subsystem parameter)

The SEPARATE_SECURITY subsystem parameter specifies whether DB2 security administrator duties are to be separated from system administrator duties for this subsystem.

Acceptable values:	NO, YES
Default:	NO

Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SEPARATE_SECURITY
Security parameter:	Yes

NO Overlaps DB2 security administrator duties and system administrator duties. If you are migrating and want SYSADM and SYSCTRL authorities to remain unchanged, use this setting. Users with SYSADM authority can manage all security objects, perform grants, and revoke privileges that are granted by others. Users with SYSCTRL authority can manage roles, perform most grants, and revoke privileges that are granted by others.

YES
Separates DB2 security administrator duties from system administrator duties. Users with SYSADM authority cannot manage security objects (such as roles and trusted contexts), perform grants, or revoke privileges granted by others. Users with SYSCTRL authority cannot manage roles, perform grants, or revoke privileges granted by others. However, existing grants made by users with SYSADM or SYSCTRL authority are unchanged. SECADM or ACCESSCTRL authority is required for security administration.

Before setting SEPARATE SECURITY to YES, set at least one SECADM system parameter to an authorization ID, or create the necessary trusted contexts and roles. If you specify YES, system administrator authority can no longer be used to perform security tasks, and the SECADM authority is required to manage security objects such as trusted contexts and roles. If both SECADM system parameters are set to roles and those roles have not been created, no one will have the authority to manage security objects.

Note: This is a security-related parameter.

Related tasks:

"Installation step 9: Establish subsystem security (optional)" on page 520

UNKNOWN AUTHID field (DEFLTID subsystem parameter)

The DEFLTID subsystem parameter specifies the authorization ID that is to be used if RACF is not available for batch access and USER= is not specified in the JOB statement.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	IBMUSER
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DEFLTID

RESOURCE AUTHID field (RLFAUTH subsystem parameter)

The RLFAUTH subsystem parameter specifies the authorization ID that is to be used if you plan to use the resource limit facility (governor).

Acceptable values:	1 to 8 characters
Default:	SYSIBM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFAUTH

BIND NEW PACKAGE field (BINDNV subsystem parameter)

The BINDNV subsystem parameter controls whether BIND or BINDADD authority is to be required for a user to bind a new version of an existing package.

Acceptable values:	BINDADD, BIND
Default:	BINDADD
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM BINDNV
Security parameter:	Yes

If you accept the default, BINDADD, only users with the BINDADD system privilege are allowed to create a new package. If you specify BIND, users with the BIND privilege on a package or collection are allowed to create a new version of an existing package when they bind it. Users with PACKADM authority are also allowed to add a new package or a new version of a package to a collection.

Note: This is a security-related parameter. When this parameter is set to BIND, the BINDADD privilege check is bypassed, and a user with PACKADM authority can create new packages.

Related reference:

Privileges required for handling plans and packages (Managing Security)

DBADM CREATE AUTH field (DBACRVW subsystem parameter)

The DBACRVW subsystem parameter controls whether an authorization ID with DBADM authority on a database is to be allowed to complete certain tasks.

Acceptable values:	NO, YES
Default:	NO
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DBACRVW
Security parameter:	Yes

NO Does not allow authorization IDs with DBADM authority on a database to complete the following tasks:

- Create a view for another authorization ID on tables in that database.
- Create a materialized query table or alter a table to become a materialized query table for another authorization ID.
- Create an alias for itself or another authorization ID for a table in that database.

YES Allows authorization IDs with DBADM authority on a database to complete the following tasks:

- Create a view for another authorization ID on tables in that database.
- Create a materialized query table or alter a table to become a materialized query table for another authorization ID. This action requires that DBADM authority is held on the database in which the tables of the fullselect reside and that the authorization ID has DBADM authority on the database in which the materialized query table is to reside.
- Create an alias for itself or another authorization ID for a table in that database.

If you specify YES, an authorization ID with DBCTRL authority on a database can also create an alias for itself or for another authorization ID for a table or view in any database.

Specifying YES results in less need for SYSADM authority on a database. However, users that need full authority may still need to have SYSADM authority. Specifying YES does not allow an authorization ID with DBADM authority to grant authority on that view.

Note: This is a security-related parameter. It controls whether an authorization ID or role with DBADM authority on a database is to be allowed to create views, aliases, and materialized query tables for another authorization ID or role. If it is set to YES, authorization IDs and roles with DBADM authority on a database can give others access to user data.

REVOKE DEP PRIV field (REVOKE_DEP_PRIVILEGES subsystem parameter)

The REVOKE_DEP_PRIVILEGES subsystem parameter controls whether revoking a privilege from a user is to cause dependent privileges to be revoked. If dependent privileges are to be revoked, revoking a privilege from a user also revokes the privilege from anyone that the user has granted that privilege to.

Acceptable values:	NO, YES, SQLSTMT
Default:	SQLSTMT
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM REVOKE_DEP_PRIVILEGES
Security parameter:	Yes

NO REVOKE statements do not include dependent privileges. An error occurs if a REVOKE statement contains the INCLUDING DEPENDENT PRIVILEGES clause.

YES
REVOKE statements include dependent privileges, except when ACCESSCTRL, DATAACCESS, and system DBADM authorities are revoked. An error occurs if a REVOKE statement contains the NOT INCLUDING DEPENDENT PRIVILEGES clause, except when ACCESSCTRL, DATAACCESS, and system DBADM authorities are revoked.

SQLSTMT
Allows revoking of dependent privileges to be controlled at the SQL level, as specified in REVOKE statements. DB2 recognizes the dependent privileges clause (INCLUDING DEPENDENT PRIVILEGES or NOT INCLUDING DEPENDENT PRIVILEGES) of the REVOKE statement

Note: This is a security-related parameter. If it is set to NO, privileges that were granted by a user are retained even if that user loses the authority that allowed the user to perform the grant.

ROUTINES CREATOR field

The ROUTINES CREATOR field specifies the CURRENT SQLID setting that is to be used when creating, configuring, and validating most DB2-supplied routines. This field also specifies the default OWNER that is to be used when binding packages for these routines.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	none

The value that you enter in the ROUTINES CREATOR field is assigned by the installation CLIST as the setting of the AUTHID parameter for installation programs DSNTRIN in job DSNTIJRT and DSNTRVFY in job DSNTIJRV.

The AUTHID parameter is used by the DSNTRIN program as the CURRENT SQLID setting when creating and configuring DB2-supplied routines. The DSNTRIN program also uses the AUTHID parameter as the default OWNER when binding packages for the DB2-supplied routines.

The AUTHID parameter is used by the DSNTRVFY program as the CURRENT SQLID setting when validating DB2-supplied routines. The DSNTRVFY program also uses the AUTHID parameter as the default OWNER when binding packages for validation of these routines.

Recommendation: Specify an authorization ID that has installation system administrator authority. Routines that are created or configured with installation system administrator authority are marked as system-defined. See the information about the SYSTEM_DEFINED column in SYSIBM.SYSROUTINES table (DB2 SQL).

SEC DEF CREATOR field

The value of the SEC DEF CREATOR field specifies the CURRENT SQLID setting that is to be used when creating and configuring DB2-supplied routines that are defined with the SECURITY DEFINER option.

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	none

The value that you enter in the SEC DEF CREATOR field is assigned by the installation CLIST as the setting of the SECDEFID parameter for installation program DSNTRIN in job DSNTIJRT.

The SECDEFID parameter is used by the DSNTRIN program as the CURRENT SQLID setting when creating and configuring DB2-supplied routines that have the SECURITY DEFINER attribute. The default owner of the packages that are bound for these routines is the ID that is specified in the ROUTINES CREATOR field.

Requirement: Specify an ID that can be used as a logon ID because it will be used by WLM to execute routines that have the SECURITY DEFINER attribute.

MVS PARMLIB updates panel: DSNTIPM

The entries on the DSNTIPM panel produce the DSNTIJMV job, which defines DB2 to z/OS and updates several PARMLIB members.

DSNTIJMV updates the following PARMLIB members:

- IEFSSN*xx*, to define DB2 and IRLM as formal z/OS subsystems
- IEAAPF*xx*, to authorize the *prefix*.SDSNLOAD, *prefix*.SDSNLINK, and *prefix*.SDSNEXIT libraries
- LNKLST*xx*, to include the *prefix*.SDSNLINK library.

Updating the parameters: Different sites have different requirements for identifying DB2 to z/OS; as a result, the updates that DSNTIJMV makes to z/OS PARMLIB members might be incomplete. To ensure that the updates are complete, it is recommended that you edit the z/OS PARMLIB members directly when you install or migrate DB2. This is substantially easier than editing DSNTIJMV.

```
DSNTIPM         INSTALL DB2 - MVS PARMLIB UPDATES
===>

Check data and reenter to change:

 1  SUBSYSTEM NAME       ===> DSN1       Name for connecting to DB2
 2  COMMAND PREFIX       ===> -DSN1      DB2 subsystem command prefix
 3  SUBSYSTEM MEMBER     ===> 00         xx in IEFSSNxx
 4  SUBSYSTEM SEQUENCE   ===> 88888888   Sequence number for insertion
 5  AUTH MEMBER          ===> 00         xx in IEAAPFxx APF member name
 6  AUTH SEQUENCE        ===> 88888888   Sequence number for insertion
 7  LINK LIST ENTRY      ===> 00         xx in LNKLSTxx for DSNLINK
 8  LINK LIST SEQUENCE   ===> 88888888   Sequence number for insertion
 9  COMMAND SCOPE        ===> STARTED    SYSTEM, SYSPLEX, or STARTED
10  SUPPRESS SOFT ERRORS ===> YES        Suppress logrec recording. Yes or No




PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 42. MVS PARMLIB updates panel: DSNTIPM

SUBSYSTEM NAME field (SSID DECP value)

The SUBSYSTEM NAME field specifies the z/OS subsystem name for DB2.

Acceptable values:	1 to 4 characters, the first of which must be a letter (A - Z), #, $, or @. Others must be letters A - Z, integers 0 - 9, #, $, or @.
Default:	DSN1
Update:	see below
dsnhdecp:	SSID

The name is used in member IEFSSN*xx* of SYS1.PARMLIB.

If you specified a group attachment name in the GROUP ATTACH field on panel DSNTIPK, the SSID DECP value is set to that value.

Updating the parameters: Different sites have different requirements for identifying DB2 to z/OS; as a result, the updates that DSNTIJMV makes to z/OS PARMLIB

members might be incomplete. To ensure that the updates are complete, you should edit the z/OS PARMLIB members directly when you install or migrate DB2. This is substantially easier than editing DSNTIJMV.

COMMAND PREFIX field

The value of the COMMAND PREFIX field specifies the DB2 subsystem command prefix. When the prefix appears at the beginning of a command that is entered at a z/OS operator's console, z/OS passes the command to DB2 for processing. The command prefix is used in the DB2 entry of member IEFSSN*xx* of SYS1.PARMLIB.

Acceptable values:	1 to 8 characters; the first character must be a non-alphanumeric character.
Default:	-DSN1 (hyphen, concatenated with subsystem name)
Update:	see "MVS PARMLIB updates panel: DSNTIPM" on page 422
DSNZP*xxx*:	none

The first character of the command prefix must be a character from the table below. The remaining characters of the command prefix must be from the table below, letters A through Z, or numbers 0 through 9.

Table 75. Allowable special characters for the command prefix

Name	Character	Hexadecimal representation
cent sign	¢	X'4A'
period	.	X'4B'
less-than sign	<	X'4C'
plus sign	+	X'4E'
vertical bar	|	X'4F'
ampersand[1]	&	X'50'
exclamation point	!	X'5A'
dollar sign	$	X'5B'
asterisk	*	X'5C'
right parenthesis	)	X'5D'
semi-colon	;	X'5E'
hyphen	-	X'60'
slash	/	X'61'
percent sign	%	X'6C'
underscore	_	X'6D'
question mark	?	X'6F'
colon	:	X'7A'
number sign	#	X'7B'
at sign	@	X'7C'
apostrophe[2]	'	X'7D'
equal sign	=	X'7E'
quotation marks	"	X'7F'

1. To use the ampersand (&), accept the default in this field, and then edit job DSNTIJMV to specify the ampersand as the command prefix.

2. To use the apostrophe ('), you must code two consecutive apostrophes in your IEFSSNxx member. For example, the entry for subsystem DB2A with a command prefix of 'DB2A and a scope of started looks like this:

```
DB2A,DSN3INI,'DSN3EPX,''DB2A,'
```

Do not use the JES2 backspace character as a command prefix character. Do not assign a command prefix that is used by another subsystem or that can be interpreted as belonging to more than one subsystem or z/OS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix beginning from the first character. For example, you cannot assign "-" to one subsystem and "-DB2A" to another. Similarly, you cannot assign "?DB2" to one subsystem and "?DB2A" to another. However, you can assign "-DB2A" and "-DB2B" to different DB2 subsystems.

To use multiple-character command prefixes, have the system programmer update the IEFSSN*xx* subsystem definition statements in SYS1.PARMLIB.

Related concepts:

Advantages of database access threads in INACTIVE mode (DB2 Performance)

SUBSYSTEM MEMBER field

The value of the SUBSYSTEM MEMBER field specifies the last two characters (*xx*) of the name of member IEFSSN*xx* of SYS1.PARMLIB.

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see "MVS PARMLIB updates panel: DSNTIPM" on page 422
DSNZP*xxx*:	none

The subsystem member name indicates the available z/OS subsystems, including DB2 and IRLM.

SUBSYSTEM SEQUENCE field

The SUBSYSTEM SEQUENCE field specifies a sequence number that is to be used for insertion. Use any number that is greater than the highest sequence number that is already used in the IEFSSN*xx* PARMLIB member.

Acceptable values:	1 to 99999993
Default:	88888888
Update:	see "MVS PARMLIB updates panel: DSNTIPM" on page 422
DSNZP*xxx*:	none

AUTH MEMBER field

The value of the AUTH MEMBER field specifies the last two characters (*xx*) of the name of member IEAAPF*xx* of SYS1.PARMLIB. This member is to be used for authorized program facility (APF) authorization of the *prefix*.SDSNLOAD, *prefix*.SDSNLINK, and *prefix*.SDSNEXIT libraries.

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see "MVS PARMLIB updates panel: DSNTIPM" on page 422
DSNZP*xxx*:	none

The specified data set must be APF-authorized. The member name must currently exist for the z/OS update job DSNTIJMV to work correctly.

You can use the PROG*xx* member instead of the IEAAPF*xx* member. In this case, you must manually name the PROG*xx* member because job DSNTIJMV does not do it for you.

AUTH SEQUENCE field

The value of the AUTH SEQUENCE field is to be used as the sequence number for insertion in the IEAAPF*xx* PARMLIB member. Specify any number that is greater than the highest sequence number that is already used in the IEAAPF*xx* PARMLIB member.

Acceptable values:	1 to 99999995
Default:	88888888
Update:	see “MVS PARMLIB updates panel: DSNTIPM” on page 422
DSNZP*xxx*:	none

LINK LIST ENTRY field

The value of the LINK LIST ENTRY field specifies the last two characters of LNKLST*xx* as needed to include the *prefix*.SDSNLINK library.

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see “MVS PARMLIB updates panel: DSNTIPM” on page 422
DSNZP*xxx*:	none

LINK LIST SEQUENCE field

The LINK LIST SEQUENCE field specifies the sequence number that is to be used for insertion. Use any number that is greater than the highest sequence number that is already used in the LNKLST*xx* PARMLIB member.

Acceptable values:	1 to 99999999
Default:	88888888
Update:	see “MVS PARMLIB updates panel: DSNTIPM” on page 422
DSNZP*xxx*:	none

COMMAND SCOPE field

The value of the COMMAND SCOPE field controls whether the scope of the command prefix is to be one z/OS system or the entire Sysplex.

Acceptable values:	SYSTEM, SYSPLEX, STARTED
Default:	STARTED
Update:	see “MVS PARMLIB updates panel: DSNTIPM” on page 422
DSNZP*xxx*:	none

SYSTEM
Sets the scope of commands to one z/OS system. The command prefix is registered at z/OS IPL.

SYSPLEX
Sets the scope of commands to the entire Sysplex. The command prefix is registered at z/OS IPL.

STARTED
Sets the scope of commands to the entire Sysplex. The command prefix is registered at DB2 startup and unregistered when DB2 stops.

Although STARTED specifies a Sysplex scope, you can also use it for a DB2 subsystem in a non-data-sharing environment. Use STARTED if you intend to use the z/OS automatic restart manager, or if you might move this DB2 into a data sharing group.

SUPPRESS SOFT ERRORS field (SUPERRS subsystem parameter)

The SUPERRS subsystem parameter specifies whether DB2 is to record errors such as invalid decimal data and arithmetic exceptions and issue SQLCODEs for them. This option enables or disables the recording of these errors in the operating system data set, SYS1.LOGREC.

Acceptable values:	YES, NO
Default:	YES
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	SUPERRS

Active log data set parameters: DSNTIPL

The entries on the DSNTIPL panel define characteristics of active log data sets.

Performance note: Several fields on this panel affect the DB2 use of logging. Be careful when you determine the values that are associated with fields on this panel. These values can greatly affect the performance of your DB2 subsystem.

```
DSNTIPL         INSTALL DB2 - ACTIVE LOG DATA SET PARAMETERS
===> _

Enter data below:

 1  NUMBER OF LOGS     ===> 3        Number data sets per active log copy (2-31)
 2  OUTPUT BUFFER      ===> 4000K    Size in bytes (400K-400000K)
 3  ARCHIVE LOG FREQ   ===> 24       Hours per archive run
 4  UPDATE RATE        ===> 3600     Updates, inserts, and deletes per hour


PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 43. Log data sets panel: DSNTIPL

Related concepts:

How the initial DB2 logging environment is established (DB2 Administration Guide)

NUMBER OF LOGS field

The NUMBER OF LOGS field specifies the number of data sets that are to be established for each copy of the active log during installation.

Acceptable values:	2 to 93
Default:	3
Update:	cannot change during update or migration

DSNZP*xxx*:	none

You can later use the DSNJU003 utility to change the number of log data sets. If you use the DSNJU003 utility to modify the number of logs, your modified value is not reflected on this panel.

Related tasks:

"Updating other parameters" on page 505

OUTPUT BUFFER field (OUTBUFF subsystem parameter)

The OUTBUFF subsystem parameter specifies the size of the output buffer that is to be used for writing active log data sets.

Acceptable values:	400K to 400000K
Default:	4000K
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP OUTBUFF

You can enter the value in bytes (for example, 409600) or use the abbreviation "K" for kilobytes (for example, 400K). The larger the output buffer, the more likely that a requested RBA can be found without a read request. In Version 10, log buffers are permanently page-fixed. When you estimate real storage usage, you must use the entire size that you specify for the OUTBUFF parameter. To avoid page-fixing more storage than necessary, carefully choose the setting for OUTBUFF.

ARCHIVE LOG FREQ field

The value of the ARCHIVE LOG FREQ field determines, in hours, the interval at which the active log is to be offloaded to the archive log.

Acceptable values:	1 to 200
Default:	24
Update:	cannot change during update
DSNZP*xxx*:	none

If you accept the default value of 24, the active log is offloaded approximately once each day.

UPDATE RATE field

The value of the UPDATE RATE field specifies the estimated average number of inserts, updates, and deletes that are expected per hour in your subsystem.

Acceptable values:	1 to 16,777,216
Default:	3600
Update:	cannot change during update
DSNZP*xxx*:	none

You can use K (as in 32K) for multiples of 1024 bytes and M (as in 16M) for multiples of 1,048,576 bytes.

The size calculations in the DSNTINST CLIST assume that about 400 bytes of data are logged for each insert, update, and delete. The amount of data that is logged for these changes might be different at your site. Therefore, consider changing the

size of the log data sets after you gain some experience with DB2 and have a better idea of how many bytes of data are logged for each change. Generally, if you have a subsystem that is tuned for maximum efficiency, you can expect to log about 10 GB of data per hour while processing several millions of updates and inserts.

Together, the UPDATE RATE and the ARCHIVE LOG FREQ (field 3) determine the size of the active logs.

Checkpoint parameters: DSNTIPL1

The entries on the DSNTIPL1 panel define characteristics for log checkpoints.

```
DSNTIPL1        INSTALL DB2 - CHECKPOINT PARAMETERS
 ===>

 Enter log checkpoint options:
   1 CHECKPOINT TYPE     ===> MINUTES   LOGRECS only, MINUTES only, or BOTH
   2 RECORDS/CHECKPOINT  ===> NOTUSED    Log records per checkpoint
                                          - LOGRECS only: 1000 - 16000000
                                          - MINUTES only: NOTUSED
                                          - BOTH        : 1000 - 99999999
   3 MINUTES/CHECKPOINT  ===> 5          Minutes per checkpoint:
                                          - LOGRECS only: NOTUSED
                                          - MINUTES only: 1 - 60
                                          - BOTH        : 1 - 1439
   4 UR CHECK FREQ       ===> 5         Checkpoints to enable UR check. 0-255
   5 UR LOG WRITE CHECK  ===> 10K       Log Writes to enable UR check. 0-1000K
   6 LIMIT BACKOUT       ===> AUTO      Limit backout processing (AUTO, YES, NO,
                                        LIGHT, or LIGHTAUTO)
   7 BACKOUT DURATION    ===> 5         Checkpoints processed during backout if
                                        LIMIT BACKOUT is not NO (0-255)
   8 RO SWITCH CHKPTS    ===> 10        Checkpoints to read-only switch. 1-32767
   9 RO SWITCH TIME      ===> 10        Minutes to read-only switch. 1-32767
  10 LEVELID UPDATE FREQ ===> ON        Checkpoints between updates. ON or OFF

 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

CHECKPOINT TYPE field (CHKTYPE subsystem parameter)

The CHKTYPE subsystem parameter indicates whether the interval between log checkpoints is to be based on the number of written log records, the time between checkpoints, or both.

Acceptable values:	LOGRECS, MINUTES, BOTH
Default:	MINUTES
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CHKTYPE

The following values can be specified in the CHECKPOINT TYPE field on panel DSNTIPL1. If you run the installation or migration CLIST, the setting of the CHECKPOINT TYPE field determines the setting of the CHKTYPE subsystem parameter. The setting of the CHECKPOINT TYPE field also defines which settings are allowed in the RECORDS/CHECKPOINT and MINUTES/CHECKPOINT fields.

LOGRECS
Log checkpoints occur after a certain number of log records are created, as specified by the CHKFREQ subsystem parameter. The CHKFREQ

subsystem parameter is set through the RECORDS/CHECKPOINT field, and the CHKLOGR and CHKMINS subsystem parameters are both assigned a value of NOTUSED. When LOGRECS is specified in the CHECKPOINT TYPE field, the CHKTYPE subsystem parameter is set to SINGLE.

MINUTES
Log checkpoints occur at an interval of a certain number of minutes, as specified by the CHKFREQ subsystem parameter. The CHKFREQ subsystem parameter is set through the MINUTES/CHECKPOINT field, and the CHKLOGR and CHKMINS subsystem parameters are both assigned a value of NOTUSED. When MINUTES is specified in the CHECKPOINT TYPE field, the CHKTYPE subsystem parameter is set to SINGLE.

BOTH Log checkpoints occur after a specified number of log records are created or after a specified number of minutes, whichever occurs first. The number of log records is specified by the CHKLOGR subsystem parameter and the number of minutes is specified by the CHKMINS subsystem parameter. These subsystem parameters are set through the RECORDS/CHECKPOINT and MINUTES/CHECKPOINT fields, respectively. The CHKFREQ parameter is assigned a value of NOTUSED. When BOTH is specified in the CHECKPOINT TYPE field, the CHKTYPE subsystem parameter is set to BOTH.

The following settings are valid for the CHKTYPE subsystem parameter:

SINGLE
The DB2 subsystem uses the CHKFREQ subsystem parameter setting to determine the number of log records or the number of minutes between checkpoints. The CHKLOGR and CHKMINS subsystem parameters must both be set to NOTUSED.

BOTH Log checkpoints occur after a specified number of log records are created or after a specified number of minutes, whichever occurs first. The number of log records is specified by the CHKLOGR subsystem parameter and the number of minutes is specified by the CHKMINS subsystem parameter. The CHKFREQ subsystem parameter must be set to NOTUSED.

RECORDS/CHECKPOINT field (CHKFREQ and CHKLOGR subsystem parameters)

The RECORDS/CHECKPOINT field determines the number of log records that are to be created between log checkpoints.

Acceptable values:	1 000 - 16 000 000 (if CHECKPOINT TYPE is LOGRECS); NOTUSED (if CHECKPOINT TYPE is MINUTES); 1 000 - 99 999 999 (if CHECKPOINT TYPE is BOTH)
Default:	NOTUSED
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP.CHKFREQ and DSN6SYSP.CHKLOGR

- If the CHECKPOINT TYPE field is set to LOGRECS, then this field must contain an integer from 1 000 through 16 000 000. This integer is the number of log records that are created between checkpoints, and the value is assigned to the CHKFREQ parameter. The value NOTUSED is assigned to the CHKLOGR parameter.

- If the CHECKPOINT TYPE field is set to MINUTES, then this field must contain the value NOTUSED. The CHKFREQ parameter is assigned the value of the MINUTES/CHECKPOINT field, and the CHKLOGR parameter is assigned the value NOTUSED.
- If the CHECKPOINT TYPE field is set to BOTH, then this field must contain an integer from 1 000 through 99 999 999. This integer is the number of log records that are created between checkpoints, and the value is assigned to the CHKLOGR parameter. The value NOTUSED is assigned to the CHKFREQ parameter.

MINUTES/CHECKPOINT field (CHKFREQ and CHKMINS subsystem parameters)

The MINUTES/CHECKPOINT field determines the number of minutes that are to pass between log checkpoints.

Acceptable values:	NOTUSED (if CHECKPOINT TYPE is LOGRECS); 1 - 60 (if CHECKPOINT TYPE is MINUTES); 1- 1439 (if CHECKPOINT TYPE is BOTH)
Default:	5
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP.CHKFREQ and DSN6SYSP.CHKMINS

- If the CHECKPOINT TYPE field is set to LOGRECS, then this field must contain the value NOTUSED. The value of the RECORDS/CHECKPOINT field is assigned to the CHKFREQ parameter. The value NOTUSED is assigned to the CHKMINS parameter.
- If the CHECKPOINT TYPE field is set to MINUTES, then this field must contain an integer from 1 through 60. This integer is the number of minutes between log checkpoints, and the value is assigned to the CHKFREQ parameter. The value NOTUSED is assigned to the CHKMINS parameter.
- If the CHECKPOINT TYPE field is set to BOTH, then this field must contain an integer from 1 through 1439. This integer is the number of minutes between checkpoints, and the value is assigned to the CHKMINS parameter. The value NOTUSED is assigned to the CHKFREQ parameter.

UR CHECK FREQ field (URCHKTH subsystem parameter)

Unless the option is disabled, DB2 issues a warning message to the console for an uncommitted unit of recovery (UR) after a certain number of checkpoint cycles. The URCHKTH subsystem parameter specifies the number of checkpoint cycles that are to complete before DB2 issues the warning message.

Acceptable values:	0 to 255
Default:	5
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP URCHKTH

Accept the default to disable this option. This option does not affect performance. If you use this option, specify a value that is based on how often a checkpoint occurs in your system and how much time you can allow for a restart or shutdown. For example, if your site's checkpoint interval is 5 minutes and the standard limit for issuing commits with units of recovery is 20 minutes, divide 20 by 5 to determine the best value for your system.

UR LOG WRITE CHECK field (URLGWTH subsystem parameter)

Unless this option is disabled, DB2 issues a warning message to the console after log records are written by an uncommitted unit of recovery (UR). The URLGWTH subsystem parameter specifies the number of log records that are to be written before DB2 issues a warning message.

Acceptable values:	0 to 1000K
Default:	10K
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP URLGWTH

The purpose of this option is to provide notification of a long-running UR. Long-running URs might result in a lengthy DB2 restart or a lengthy recovery situation for critical tables. Specify the value in 1 K (1000 log records) increments. A value of 0 indicates that no write check is to be performed.

LIMIT BACKOUT field (LBACKOUT subsystem parameter)

The LBACKOUT subsystem parameter specifies whether DB2 is to postpone backward-log processing for some units of work.

Acceptable values:	AUTO, YES, LIGHT, LIGHTAUTO, NO
Default:	AUTO
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LBACKOUT

AUTO
DB2 postpones some backout processing, but automatically starts the backout processing when DB2 restarts and begins acceptance of new work. When DB2 is restarted with the LIGHT(YES) or LIGHT(NOINDOUBTS) option, DB2 ignores this setting and behaves as if LBACKOUT is set to NO.

YES
DB2 postpones backout processing for some units of work until the RECOVER POSTPONED command is issued. When DB2 is restarted with the LIGHT(YES) or LIGHT(NOINDOUBTS) option, DB2 ignores this setting and behaves as if LBACKOUT is set to NO.

LIGHT
DB2 postpones backout processing for some units of work until the RECOVER POSTPONED command is issued. When DB2 is restarted with the LIGHT(YES) or LIGHT(NOINDOUBTS) option, it creates but does not process postponed-abort units of recovery (URs). The URs are not processed until the RECOVER POSTPONED command is issued on a normal DB2 restart.

LIGHTAUTO
DB2 postpones some backout processing, but automatically starts the backout processing when DB2 restarts and begins acceptance of new work. When DB2 is restarted with the LIGHT(YES) or LIGHT(NOINDOUBTS) option, it creates but does not process postponed-abort URs. The postponed-abort URs are automatically processed after a normal DB2 restart.

NO
Specifies that DB2 backward-log processing should process all inflight and in-abort units of recovery (URs).

If you specify AUTO, YES, LIGHT, or LIGHTAUTO, backout processing runs concurrently with new work. Sets or partitions with pending backout work are unavailable until their backout work is complete.

Related concepts:

Restart light (DB2 Data Sharing Planning and Administration)

Related reference:

-START DB2 (DB2) (DB2 Commands)

"BACKOUT DURATION field (BACKODUR subsystem parameter)"

BACKOUT DURATION field (BACKODUR subsystem parameter)

The BACKODUR subsystem parameter is used to calculate the number of log records that are to be backed out during a restart. This value is applied when the LIMIT BACKOUT field is set to AUTO, YES, LIGHT, or LIGHTAUTO.

Acceptable values:	0 to 255
Default:	5
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP BACKODUR

During a restart, backward-log processing continues until the following events occur:

- All inflight and in-abort URs with update activity against the catalog or directory are backed out.
- All inflight and in-abort URs that only update temporary tables are backed out.
- The number of processed log records equals the number specified in the BACKOUT DURATION field multiplied by the number of log records per checkpoint. If the CHECKPOINT TYPE field is set to LOGRECS or BOTH, the value of the RECORDS/CHECKPOINT field is used as the number of log records per checkpoint. If the CHECKPOINT TYPE field is set to MINUTES, the default value of 500 000 is used as the number of log records per checkpoint.

Inflight and in-abort URs that are not completely backed out during restart are converted to postponed-abort status. Sets or partitions with postponed backout work are put into restart-pending (RESTP) state. This state blocks most access to the object. The exceptions are access by the RECOVER POSTPONED command state or by automatic backout processing that DB2 performs when the LIMIT BACKOUT field is set to AUTO or LIGHTAUTO.

A table space might be in restart-pending mode, without the associated index spaces also in restart-pending mode. This happens if a postponed-abort UR makes updates only to non-indexed fields of a table in a table space. In this case, the indexes are accessible to SQL (for index-only queries), even though the table space is inaccessible.

RO SWITCH CHKPTS field (PCLOSEN subsystem parameter)

The PCLOSEN subsystem parameter specifies the number of consecutive DB2 checkpoints that are to be allowed after a page set or partition is updated. After the specified number of checkpoints occur, DB2 converts the page set or partition from read-write to read-only.

Acceptable values:	1 to 32767
Default:	10 (checkpoints)
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP PCLOSEN

This value is used in conjunction with the value of the RO SWITCH TIME field. If the condition for RO SWITCH TIME or RO SWITCH CHKPTS is met, the page set or partition is converted from read-write to read-only.

For NOT LOGGED table spaces, DB2 converts the page set or partition from read-write to read-only after one checkpoint, regardless of the value of RO SWITCH CHKPTS.

Having DB2 switch an infrequently updated set from read-write to read-only can be a performance benefit for recovery, logging, and for data sharing processing.

Related tasks:

"Testing group buffer pool caching" on page 731

RO SWITCH TIME field (PCLOSET subsystem parameter)

The PCLOSET subsystem parameter specifies the number of minutes that can elapse after a page set or partition is updated. After the specified number of minutes, DB2 converts the set or partition from read-write to read-only.

Acceptable values:	1 to 32767
Default:	10 (minutes)
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP PCLOSET

This value is used in conjunction with the value of the RO SWITCH CHKPTS field. If the condition for RO SWITCH CHKPTS or RO SWITCH TIME is met, the page set or partition is converted from read-write to read-only.

For NOT LOGGED table spaces, DB2 converts the page set or partition from read-write to read-only after one minute, regardless of the value of RO SWITCH TIME.

Having DB2 switch an infrequently updated page set from read-write to read-only can be a performance benefit for recovery, logging, and for data sharing processing.

Related tasks:

"Testing group buffer pool caching" on page 731

LEVELID UPDATE FREQ field (DLDFREQ subsystem parameter)

The DLDFREQ subsystem parameter specifies whether the level ID of a page set or partition is to be updated at DB2-determined checkpoint intervals.

Acceptable values:	ON, OFF
Default:	ON
Update:	option 28 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DLDFREQ

ON Level ID updates for down-level detection are enabled.

OFF Level ID updates for down-level detection are disabled.

Archive log data set parameters panel: DSNTIPA

The entries on the DSNTIPA panel define the characteristics of archive log data sets.

Updating the parameters: You can update all the parameters on this panel by using their subsystem parameter name.

```
DSNTIPA         INSTALL DB2 - ARCHIVE LOG DATA SET PARAMETERS
===> _
Enter data below:

 1  ALLOCATION UNITS ===> BLK       Blk, Trk, or Cyl
 2  PRIMARY QUANTITY ===>           Primary space allocation
 3  SECONDARY QTY    ===>           Secondary space allocation
 4  CATALOG DATA     ===> NO        YES or NO to catalog archive data sets
 5  DEVICE TYPE 1    ===> TAPE      Unit name for COPY1 archive logs
 6  DEVICE TYPE 2    ===>           Unit name for COPY2 archive logs
 7  BLOCK SIZE       ===> 24576     Rounded up to 4096 multiple
 8  READ TAPE UNITS  ===> 2         Number of allocated read tape units
 9  DEALLOC PERIOD   ===> 0         Time interval to deallocate tape units
10  RECORDING MAX    ===> 10000     Number of data sets recorded in BSDS
11  WRITE TO OPER    ===> YES       Issue WTOR before mount for archive
12  WTOR ROUTE CODE  ===> 1,3,4
13  RETENTION PERIOD ===> 9999      Days to retain archive log data sets
14  QUIESCE PERIOD   ===> 5         Maximum quiesce interval (1-999)
15  COMPACT DATA     ===> NO        YES or NO for data compaction
16  SINGLE VOLUME    ===> NO        Single volume for disk archives. NO or YES

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 44. Archive log data sets panel: DSNTIPA

ALLOCATION UNITS field (ALCUNIT subsystem parameter)

The ALCUNIT subsystem parameter controls the units (blocks, tracks, or cylinders) in which primary and secondary space allocations are to be obtained.

Acceptable values:	BLK, TRK, or CYL
Default:	BLK
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ALCUNIT

PRIMARY QUANTITY field (PRIQTY subsystem parameter)

The PRIQTY subsystem parameter specifies the amount of primary space that is to be allocated for a disk data set. The units for this parameter are specified in the ALLOCATION UNITS field.

Acceptable values:	blank or 1 to 999999
Default:	blank
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP PRIQTY

If you use the default, a blank, the CLIST calculates this space using block size and size of the log. By default, DFSMS Direct Access Device Space Management

(DADSM) limits the space allocation on a single volume to less than 65535 tracks. Therefore, if the archive log data set size can be greater than or equal to 65535 tracks, you need to specify a primary space quantity of less than 65535 tracks. This action forces the archive log data set to extend to a second volume.

Alternatively, the archive log data sets can be allocated by a DFSMS data class that has a DSNMTYP (data set name type) setting of LARGE or EXT. In this case, you can specify a primary space quantity larger than 65535 tracks without extending to a second volume. LARGE indicates that data sets in the data class are to be allocated in large physical sequential format. EXT indicates that data sets in the data class are to be allocated in extended physical sequential format. A setting of EXT is recommended, and it is required for striping of data sets. If you specify EXT, also set the IFEXT (if extended) parameter to R (required) rather than P (preferred).

Related concepts:

"Archive log data sets storage requirements" on page 45

SECONDARY QTY field (SECQTY subsystem parameter)

The SECQTY subsystem parameter specifies the amount of secondary space that is to be allocated for a disk data set. The units for this parameter are specified in the ALLOCATION UNITS field.

Acceptable values:	blank or 1 to 999999
Default:	blank
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP SECQTY

If you use the default, a blank, the CLIST calculates this space using block size and size of the log.

CATALOG DATA field (CATALOG subsystem parameter)

The value of the CATALOG DATA field specifies whether archive log data sets on tape are to be cataloged.

Acceptable values:	YES, NO
Default:	NO
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP CATALOG

This option is meaningful only if you specify TAPE for the DEVICE TYPE 1 or DEVICE TYPE 2 field on panel DSNTIPA. The reason is that DB2 requires all archive log data sets that are allocated on disk to be cataloged.

If you choose to archive to disk, the CATALOG DATA field must be set to YES. If it is set to NO, you receive message DSNJ072E each time an archive log data set is allocated, although the DB2 subsystem still catalogs the data set.

Recommendation: If you archive to data sets on tape, set CATALOG DATA to YES.

Doing so allows the archive log data sets to be portable. DB2 accesses the archive logs through the catalog instead of the BSDS content that was written when the archive was created. One advantage to cataloging archived tape data sets is evident

when you perform operations like concurrent recovery. During such operations, for which the archive data sets must be shared and require faster devices, you can copy and recatalog the archive logs without modifying the BSDS.

Important: Setting CATALOG DATA to YES might prevent log read errors when the archive log data sets span multiple tape volumes. In this case, DB2 accesses the entire archive log data set through the catalog instead of using the individual volume information that is recorded in the BSDS.

DEVICE TYPE 1 field (UNIT subsystem parameter)

The UNIT subsystem parameter specifies the device type or unit name that is to be used for storing archive log data sets.

Acceptable values:	device type or unit name
Default:	TAPE
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP UNIT

The value can be any alphanumeric string. If you choose to archive to disk, you can specify a generic device type with a limited volume range. DB2 requires all archive log data sets that are allocated on disk to be cataloged. If the device type specifies disk, set the CATALOG DATA field on panel DSNTIPA to YES.

If the unit name specifies disk, the archive log data sets can extend to a maximum of 15 volumes. If the unit name specifies a tape device, DB2 can extend to a maximum of 20 volumes. If you chose to use disk, make the values of the PRIQTY and SECQTY subsystem parameters large enough to contain all of the data that comes from the active log data sets without extending beyond 15 volumes.

When archiving to disk, DB2 uses the number of online storage volumes for the specified UNIT name to determine a count of candidate volumes, up to a maximum of 15 volumes. If the archives are to be managed by SMS, do not use a storage class with the Guaranteed Space attribute. SMS attempts to allocate a primary extent on every candidate volume. This can result in allocation failures or unused space because the primary extent on each unused volume is not released when the archive data set is closed.

Related concepts:

Archive log data sets (DB2 Administration Guide)

Related tasks:

"Installation step 5: Define DB2 initialization parameters: DSNTIJUZ" on page 516

Related reference:

"PRIMARY QUANTITY field (PRIQTY subsystem parameter)" on page 434
"SECONDARY QTY field (SECQTY subsystem parameter)" on page 435

DEVICE TYPE 2 field (UNIT2 subsystem parameter)

The UNIT2 subsystem parameter specifies the device type or unit name that is to be used for storing the second copy of archive log data sets. These data sets are the COPY2 data sets.

Acceptable values:	device type or unit name
Default:	none
Update:	option 29 on panel DSNTIPB

DSNZP*xxx*: DSN6ARVP UNIT2

BLOCK SIZE field (BLKSIZE subsystem parameter)

The BLKSIZE subsystem parameter determines the block size that is to be used for the archive log data set.

Acceptable values: 8192 to 28672
Default: 24576
Update: option 29 on panel DSNTIPB
DSNZP*xxx*: DSN6ARVP BLKSIZE

The block size must be compatible with the device type that you use for archive logs. The value is rounded up to the next multiple of 4096 bytes. You can also enter the value with a K; for example, 28K.

If the archive log is written to tape, use the largest possible block size to improve the speed of reading the archive logs. Use the table below as a guide.

Table 76. Recommended block size values

Archive log device	Block size
Tape	28672
3390 geometry disk	24576

READ TAPE UNITS field (MAXRTU subsystem parameter)

The MAXRTU subsystem parameter specifies the maximum number of dedicated tape units that can be allocated to concurrently read archive log tape volumes.

Acceptable values: 1 to 99
Default: 2
Update: option 29 on panel DSNTIPB
DSNZP*xxx*: DSN6LOGP MAXRTU

This installation option, along with the DEALLOC PERIOD field, allows DB2 to optimize archive log reading from tape devices.

In a data sharing environment, the archive tape is not available to other members of the group until the deallocation period expires. You might not want to use this option in a data sharing environment unless all recover jobs are submitted from the same member.

Recommendation: Set the READ TAPE UNITS value to be at least one less than the number of tape units available to DB2. If you do otherwise, the OFFLOAD process could be delayed, which would affect the performance of your DB2 subsystem. For maximum throughput during archive log processing, specify the largest value possible for this option, remembering that you need at least one tape unit for offload processing. You can override this value by using the SET ARCHIVE command.

DEALLOC PERIOD field (DEALLCT subsystem parameter)

The DEALLCT subsystem parameter determines the length of time that an archive read tape unit is to be allowed to remain unused before it is deallocated.

Acceptable values:	*minutes*,*seconds*, 1440, NOLIMIT
Default:	0
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP DEALLCT

minutes,*seconds*
: Sets the amount of time before deallocation as the specified number of minutes (0 – 1440) and number of seconds (0 – 59). Separate the number of minutes and number of seconds with a comma and no spaces.

1440
: Sets the amount of time before deallocation to 1440 minutes.

NOLIMIT
: Sets no limit on the amount of time before deallocation. Specifying NOLIMIT allows maximum optimization opportunities.

Recommendation: If you are not using data sharing and your archive log data is on tape, set this value high enough to allow DB2 to optimize tape handling for multiple read applications. When all tape reading is complete, you can update this option with the SET ARCHIVE command. If you are using data sharing, tapes should be deallocated quickly to avoid deadlocks. If DB2 is running in data sharing mode, it is strongly recommended that you use disk archive logs, not tape archive logs.

Examples

Assume that you want the deallocation period to be 30 seconds. Enter `0,30`.

Assume that you want the deallocation period to be 23 minutes and 47 seconds. Enter `23,47`.

RECORDING MAX field (MAXARCH subsystem parameter)

The MAXARCH subsystem parameter controls the maximum number of archive log volumes that are to be recorded in the BSDS.

Acceptable values:	10 to 10000
Default:	10000
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP MAXARCH

When this number is exceeded, recording resumes at the beginning of the BSDS.

You must create image copies of all DB2 objects, probably several times, before the archive log data sets are discarded. If you fail to retain an adequate number of archive log data sets for all the image copies, you might need to cold start or reinstall DB2. In both cases, data is lost.

Related tasks:

Managing the log (DB2 Administration Guide)

WRITE TO OPER field (ARCWTOR subsystem parameter)

The ARCWTOR subsystem parameter controls whether DB2 is to send a message to the operator and wait for an answer before attempting to mount an archive log data set.

Acceptable values:	YES, NO
Default:	YES
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCWTOR

Other DB2 users can be forced to wait while the mount is pending. They are not affected while DB2 is waiting for a response to the message.

NO Specifies that the system will not send a message to the operator before attempting to mount an archive log data set. Specify this value if you use a device for storing archive log data sets, such as tape, that requires long delays for mounts.

YES Specifies that the system is to send a message to the operator and wait for an answer before attempting to mount an archive log data set. Use this value if you use a device that does not have long delays for mounts.

The DEVICE TYPE 1 field specifies the device type or unit name.

WTOR ROUTE CODE field (ARCWRTC subsystem parameter)

The ARCWRTC subsystem parameter identifies the list of route codes to be used in messages from the archive log data sets to the operator.

Acceptable values:	1 to 16 route codes. Specify at least one code. Separate multiple codes by commas only, with no blanks; for example: 1,3,5,7,9,10,11
Default:	1,3,4
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCWRTC

Related reference:

Routing Codes (MVS System Messages)

Routing the Message (z/OS MVS Assembler Services Guide

RETENTION PERIOD field (ARCRETN subsystem parameter)

The ARCRETN subsystem parameter specifies the retention period, which is the number of days that DB2 is to retain archive log data sets. The retention period is added to the current date to calculate the expiration date of the archive log data sets.

Acceptable values:	0 to 9999
Default:	9999
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCRETN

The retention period is often used in tape management systems to control the reuse and scratching of data sets and tapes. DB2 uses the ARCRETN value as the value for the dynamic allocation parameter DALRETPD when archive log data sets are created.

The retention period set by the DFSMSdfp storage management subsystem (SMS) can be overridden by this DB2 parameter. Typically, the retention period is set to

the smaller value that is specified by either DB2 or SMS. The storage administrator and database administrator should agree on a retention period value that is appropriate for DB2.

Important: A wide variety of tape management systems exists, along with the opportunity for external manual overrides of retention periods. Consequently, DB2 does not have an automated method to delete the archive log data sets from the BSDS inventory of archive log data sets. Therefore, the information about an archive log data set might be in the BSDS after its retention period has expired. The information might be held long after the tapes have been scratched by a tape management system. Conversely, the maximum number of archive log data sets might be exceeded (see READ TAPE UNITS field). In such a case, the data from the BSDS might be dropped long before the data set reaches its expiration date.

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

QUIESCE PERIOD field (QUIESCE subsystem parameter)

The QUIESCE subsystem parameter determines the maximum amount of time, in seconds, that DB2 is to be allowed to attempt a full system quiesce.

Acceptable values:	1 to 999
Default:	5
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP QUIESCE

This parameter requires some tuning. If you specify too short an interval, the quiesce period expires before a full quiesce is accomplished. If you specify too long an interval, the quiesce period might cause unnecessary DB2 lock contention and timeouts.

Related tasks:

Quiescing activity before offloading (DB2 Administration Guide)

COMPACT DATA field (COMPACT subsystem parameter)

The COMPACT subsystem parameter controls whether data that is written to archive logs is to be compacted.

Acceptable values:	YES or NO
Default:	NO
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP COMPACT

This option only applies to data that is written to a 3480 or 3490 device that has the improved data recording capability (IDRC) feature. When this feature is turned on, hardware in the IDRC-compliant tape control unit uses several techniques to increase the amount of data that can be written on a tape. Specify NO if you do not want your data to be compacted or you do not use a 3480 or 3490 device with the IDRC feature. Specify YES if you use a 3480 or 3490 device with the IDRC feature and you want the data to be compacted.

Recommendation: Archive to disk and then use another facility, such as DFSMShsm, to archive the archive log from disk to tape. Be aware that data that is

compressed to tape can only be read with a device that supports the IDRC feature. This could be a concern when you send archive tapes to another site for remote recovery.

SINGLE VOLUME field (SVOLARC subsystem parameter)

When archiving to disk, DB2 uses the number of online storage volumes for the specified UNIT name to determine a count of candidate volumes, up to 15. The SVOLARC subsystem parameter specifies whether DB2 is to use a single volume for disk archives.

Acceptable values:	YES, NO
Default:	NO
Update:	option 29 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP SVOLARC

Specify YES if you want DB2 to specify a unit and volume count of 1 when allocating a new archive log data set on disk. You may want to do this when SMS manages the archives.

Databases and spaces to start automatically panel: DSNTIPS

The entries on the DSNTIPS panel name the databases, table spaces, and index spaces to restart automatically when you start DB2.

Updating the parameters: You can update all parameters on this panel by using their subsystem parameter name.

```
DSNTIPS         INSTALL DB2 - DATABASES AND SPACES TO START AUTOMATICALLY
===> _

Enter data below:

 1  ===> RESTART       RESTART or DEFER the objects named below.
    The objects to restart or defer can be ALL in item 2, a database
    name, or database name.space name.

 2  ==> ALL               14  ==>                   26  ==>
 3  ==>                   15  ==>                   27  ==>
 4  ==>                   16  ==>                   28  ==>
 5  ==>                   17  ==>                   29  ==>
 6  ==>                   18  ==>                   30  ==>
 7  ==>                   19  ==>                   31  ==>
 8  ==>                   20  ==>                   32  ==>
 9  ==>                   21  ==>                   33  ==>
10  ==>                   22  ==>                   34  ==>
11  ==>                   23  ==>                   35  ==>
12  ==>                   24  ==>                   36  ==>
13  ==>                   25  ==>                   37  ==>

PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 45. Databases and spaces to start automatically panel: DSNTIPS

RESTART OR DEFER field (RESTART subsystem parameter)

The RESTART subsystem parameter determines whether DB2 is to restart or defer processing for the specified databases, table spaces, and index spaces when DB2 is started. The specified objects are those that are listed in fields 2 through 37 of panel DSNTIPS

Acceptable values:	RESTART, DEFER
Default:	RESTART
Update:	option 30 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RESTART

RESTART
Causes DB2 to perform restart processing for the listed objects.

DEFER
Causes DB2 not to perform restart processing for the objects.

START NAMES field (ALL subsystem parameter)

The ALL subsystem parameter specifies the databases, table spaces, and index spaces that are to be restarted or deferred when DB2 starts. The value of the RESTART OR DEFER field determines whether the object is to be restarted or if the restart is to be deferred.

Acceptable values:	ALL, space names
Default:	ALL
Update:	option 30 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ALL

ALL

The restart or defer restart applies to all DB2 databases, table spaces, and index spaces. If you use this value, leave the remaining fields on the DSNTIPS panel blank. Use ALL in this field and DEFER in the RESTART OR DEFER field to defer recovery of all objects, including DB2 catalog objects. Trusted connections are not available if ALL is used in this field and DEFER is used in the RESTART OR DEFER field.

ALL is the default value.

<*database name*>
The restart or defer restart applies to all spaces in the database.

<*table space or index space name*> **in the format `database-name.space-name`**
The restart or defer restart applies to the individual table space or index space.

You can specify up to 36 object names on this panel. If you want to control restart processing for more than 36 objects, edit job DSNTIJUZ after you run the CLIST, and add the object names as ending positional parameters to macro DSN6SPRM. You can add up to 2500 object names in DSNTIJUZ.

Related tasks:

Deferring restart processing (DB2 Administration Guide)

Distributed data facility panel 1: DSNTIPR

The entries on the DSNTIPR panel control the starting of the distributed data facility (DDF). The entries on the DSNTIPR panel also specify location names that are used to connect another DB2 subsystem.

```
DSNTIPR       INSTALL DB2 - DISTRIBUTED DATA FACILITY PANEL 1
===> _

DSNT512I WARNING: ENTER UNIQUE NAMES FOR LUNAME AND LOCATION NAME
Enter data below:

 1 DDF STARTUP OPTION   ===> NO       NO, AUTO, or COMMAND
 2 DB2 LOCATION NAME    ===> LOC1       The name other DB2s use to
                                        refer to this DB2
 3 DB2 NETWORK LUNAME   ===> LU1      The name VTAM uses to refer to this DB2
 4 DB2 NETWORK PASSWORD ===>          Password for DB2's VTAM application
 5 RLST ACCESS ERROR    ===> NOLIMIT  NOLIMIT, NORUN, or 1-5000000
 6 RESYNC INTERVAL      ===> 2        Minutes between resynchronization period
 7 DDF THREADS          ===> INACTIVE Status of a qualifying database access
                                      thread after commit. ACTIVE or INACTIVE.
 8 MAX INACTIVE DBATS   ===> 0        Max number of type 1 inactive threads.
 9 DB2 GENERIC LUNAME   ===>          Generic VTAM LU name for this DB2
                                      subsystem or data sharing group.
10 IDLE THREAD TIMEOUT  ===> 120      0 or seconds until dormant server ACTIVE
                                      thread will be terminated (0-9999)
11 EXTENDED SECURITY    ===> YES       Allow change password and descriptive
                                      security error codes. YES or NO.



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 46. Distributed data facility panel: DSNTIPR

Related concepts:

"Support for a communications network" on page 560

DDF STARTUP OPTION field (DDF subsystem parameter)

The DDF subsystem parameter controls whether DDF is to be loaded at startup and, if DDF is to be loaded, how it is to be started.

Acceptable values:	NO, AUTO, COMMAND
Default:	NO
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC DDF

NO Signifies that DDF is not to be loaded at DB2 startup and that it cannot be started with a command. If you specify NO, the stored procedures sample application and DDF sample jobs (DSNTEJ6, DSNTEJ6D, DSNTEJ6P, DSNTEJ6S, DSNTEJ6T, DSNTEJ6Z, DSNTEJ61, DSNTEJ62, DSNTEJ63, DSNTEJ64, DSNTEJ65, and DSNTEJ66) are not edited. Also, the remaining fields on the DSNTIPR panel are ignored. However, the remaining fields require entries, so you can accept the default values.

AUTO
Specifies that this facility is to be automatically initialized and started when the DB2 subsystem is started.

COMMAND
Specifies that the facility is to be initialized at DB2 startup and is prepared to receive the DSN1 START DDF command.

If AUTO or COMMAND is specified, the DDF address space is started as part of DDF initialization. The remaining fields on the DSNTIPR panel are mandatory. The repository for the field names (LOCATION, LUNAME, and PASSWORD) is the bootstrap data set (BSDS). The BSDS is updated by the change log inventory utility in step DSNTLOG of install job DSNTIJUZ.

DB2 LOCATION NAME field

The value of the DB2 LOCATION NAME field specifies the unique name that requesters are to use to connect to this DB2 subsystem. The value of the DB2 LOCATION NAME field must be the same on all members within a DB2 data sharing group.

Acceptable values:	1 to 16 alphanumeric characters
Default:	LOC1
Update:	see "Updating other parameters" on page 505
DSNZP*xxx*:	none

The location name must begin with a letter and cannot contain special characters. Acceptable characters are letters A through Z, numbers 0 through 9, and underscores.

You must specify a value for this field, even if you do not use DDF.

DB2 NETWORK LUNAME field

The value of the DB2 NETWORK LUNAME field specifies the LU name that is to uniquely identify this DB2 subsystem to VTAM. The name is also to be used to uniquely identify logical units of work within DB2 trace records.

Acceptable values:	1 to 8 alphanumeric characters
Default:	LU1
Update:	see "Updating other parameters" on page 505
DSNZP*xxx*:	none

The name must begin with a letter and must not contain special characters.

You must specify a value for this field.

DB2 NETWORK PASSWORD field

The optional DB2 NETWORK PASSWORD field specifies the password that VTAM is to use to recognize this DB2 subsystem.

Acceptable values:	1 to 8 alphanumeric characters
Default:	none
Update:	see Update on "Updating other parameters" on page 505
DSNZP*xxx*:	none

This password must also be supplied to VTAM on the VTAM APPL definition statement. The password must begin with a letter and must not contain special characters.

RLST ACCESS ERROR field (RLFERRD subsystem parameter)

The RLFERRD subsystem parameter specifies the action that DB2 is to take if the governor encounters a condition that prevents it from accessing the resource limit specification table. This setting also applies if DB2 cannot find an applicable row in the resource limit specification table.

An applicable row is one that applies to the authorization ID, plan or package name, and name of the logical unit of work of the query user. This parameter applies to the distributed data facility.

Acceptable values:	NOLIMIT, NORUN, 1 to 5000000
Default:	NOLIMIT
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC RLFERRD

NOLIMIT
Allows all dynamic SQL statements to run without limit.

NORUN
Terminates all dynamic SQL statements immediately with an SQL error code.

1 to 5000000
Specifies the default limit for the number of resource limit specification table access errors. If the limit is exceeded, the SQL statement is terminated.

Related concepts:

Controlling the resource limit facility (governor) (DB2 Administration Guide)

Related reference:

DSNRLSTxx (DB2 Performance)

RESYNC INTERVAL field (RESYNC subsystem parameter)

A *resynchronization period* is the time during which indoubt logical units of work that involve this DB2 subsystem and partner logical units are processed. The RESYNC subsystem parameter specifies the time interval, in minutes, that is to occur between resynchronization periods.

Acceptable values:	1 to 99
Default:	2
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC RESYNC

DDF THREADS field (CMTSTAT subsystem parameter)

The CMTSTAT subsystem parameter controls whether threads are to be made active or inactive after they successfully commit or roll back and hold no cursors. A thread can become inactive only if it holds no cursors, has no temporary tables defined, and executes no statements from the dynamic statement cache.

Acceptable values:	ACTIVE, INACTIVE
Default:	INACTIVE
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC CMTSTAT

ACTIVE
Threads remain active after they successfully commit or roll back and hold no cursors. This setting provides the best performance but consumes system resources. If your installation must support a large number of connections, specify INACTIVE instead.

INACTIVE
DB2 supports two different types of inactive concepts:

- An inactive DBAT (previously called a type 1 inactive thread), which has the same characteristics as inactive threads that were available in releases prior to DB2 Version 10. In this case, the thread remains associated with the connections, but the thread's storage utilization is reduced as much as possible. However, this still potentially requires a large number of threads to support a large number of connections.
- An inactive connection (previously called a type 2 inactive thread), which uses less storage than an inactive DBAT. In this case, the connections are disassociated from the thread. The thread is allowed to be pooled and reused for other connections, new or inactive. This provides better resource utilization because there are typically a small number of threads that can be used to service a large number of connections.

Because they use less storage, inactive connections are preferable. However, not all threads can become inactive connections. The following table summarizes the conditions under which you can have an inactive DBAT or an inactive connection. If a thread is to become inactive, DB2 tries to make it an inactive connection. If DB2 cannot make it an inactive connection, it tries to make it a inactive DBAT. If neither attempt is successful, the thread remains active.

Table 77. Requirements for inactive DBATs and inactive connections

If the event is...	Inactive connection can be created	Inactive DBAT can be created
A hop to another location	Yes	Yes
A package bound with RELEASE(COMMIT)	Yes	Yes
A package bound with RELEASE(DEALLOCATE)	Yes	No
An outstanding open and held cursor[1], a held LOB locator, or a package bound with KEEPDYNAMIC(YES)	No	No

[1]A cursor can be closed with fast implicit close. For more information, see Fast implicit close (DB2 Performance).

IBM Systems Virtualization Engine Enterprise Workload Manager™ (EWLM) is a robust performance management tool that allows you to monitor and manage work that runs within your environment. If you use EWLM with DB2, specify INACTIVE for DDF THREADS. Otherwise, DDF ignores all EWLM correlators that are passed to it.

MAX INACTIVE DBATS field (MAXTYPE1 subsystem parameter)

The MAXTYPE1 subsystem parameter determines the number of inactive DBATs that DB2 is to allow.

Acceptable values:	0 to the value of the MAX REMOTE CONNECTED field
Default:	0
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC MAXTYPE1

This limit is defined because a large number of inactive DBATs might adversely affect system performance. Inactive DBATs were traditionally used for private protocol, while DRDA uses inactive connections. However, inactive DBATS can occur in DRDA when DB2 acts as an intermediate server, receiving a connection from a client, and the thread then hops to another location while a held cursor spans the commit scope.

A value of 0 indicates that inactive DBATs are not allowed. If a thread meets the requirement of an inactive DBATs, and MAX INACTIVE DBATS is 0, the thread remains active.

A value of greater than 0 indicates that inactive DBATs are allowed, but they are limited to the specified number. When a thread meets the requirement of an inactive DBAT, and MAX INACTIVE DBATS is reached, the remote connection is terminated.

If you want to allow inactive DBATs, set this value to the maximum number of concurrent connections that you want to allow to go inactive that access another remote location with three-part names.

A value that is equal to the value in the MAX REMOTE CONNECTED field from panel DSNTIPE allows all remote threads to become type 1 inactive threads.

DB2 GENERIC LUNAME field

The value of the DB2 GENERIC LUNAME field specifies a generic LU name that is to identify this DB2 subsystem or data sharing group in a network.

Acceptable values:	1 to 8 alphanumeric characters
Default:	none
DSNZP*xxx*:	none

You can use a generic LU name only if DB2 is running as part of a z/OS Sysplex. Using a generic LU name helps you control the distributed workload among the servers in a data sharing group. Previously, you could associate only one LU name with a LOCATION name. Now, you can associate multiple NETID.LUNAME values with a single LOCATION name. When an application requests access to a particular location, DB2 uses the SYSIBM.LOCATIONS and SYSIBM.LULIST tables to find the available network destinations (LU names) for that location.

Related tasks:

"Connecting distributed database systems" on page 877

Configuring data sharing groups for group-generic access (DB2 Data Sharing Planning and Administration)

IDLE THREAD TIMEOUT field (IDTHTOIN subsystem parameter)

The IDTHTOIN subsystem parameter controls the amount of time, in seconds, that an active server thread is to be allowed to remain idle. This amount of time is the timeout value. The thread is canceled after the timeout value expires, and its locks and cursors are released.

Acceptable values:	0 to 9999
Default:	120
Update:	option 31 on panel DSNTIPB

DSNZP*xxx*: DSN6FAC IDTHTOIN

Inactive and indoubt threads are not subject to timeout. The value that you specify for the DDF THREADS field determines whether a thread can become inactive, and thus not subject to timeout.

Threads are checked every two minutes to see if they have exceeded the timeout value. If the timeout value is less than two minutes, the thread might not be canceled if it has been inactive for more than the timeout value but less than two minutes.

Specifying 0 disables timeout processing. If timeout processing is disabled, idle server threads remain in the system and continue to hold any resources .

EXTENDED SECURITY field (EXTSEC subsystem parameter)

The EXTSEC subsystem parameter specifies how two related security options are to be set. These settings control what happens when a DDF connection has security errors and whether RACF users can change their passwords through the DRDA change password function.

Acceptable values:	YES, NO
Default:	YES
Update:	option 31 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP EXTSEC
Subsystem parameter:	Yes

YES Detailed reason codes are returned to a DRDA level 3 client when a DDF connection request fails because of security errors. When using SNA protocols, the requester must have included a product that supports the extended security sense codes. One such product is DB2 Connect.

RACF users can change their passwords by using the DRDA change password function. This support is only for DRDA requesters that have implemented support for changing passwords.

NO Generic error codes are returned to the clients and RACF users are prevented from changing their passwords.

Recommendation: Specify a value of YES. This setting allows properly enabled DRDA clients to determine the cause of security failures without requiring DB2 operator support. A value of YES also allows RACF users on properly enabled DB2 clients to change their passwords.

Note: This is a security-related parameter. When this parameter is set to YES, detailed reason codes are returned to the client when a DDF connection request fails because of security errors that might enable more malicious attacks. If this parameter is set to YES, RACF users can change their passwords by using the DRDA change password function.

Distributed data facility panel 2: DSNTIP5

The entries on the DSNTIP5 panel are used to configure the distributed data facility (DDF).

```
DSNTIP5       INSTALL DB2 - DISTRIBUTED DATA FACILITY PANEL 2
===>_

Enter data below:

 1  DRDA PORT           ===>          TCP/IP port number for DRDA clients.
                                      1-65534 (446 is reserved for DRDA)
 2  SECURE PORT         ===>          TCP/IP port number for secure DRDA
                                      clients. 1-65534 (448 is reserved
                                      for DRDA using SSL)
 3  RESYNC PORT         ===>          TCP/IP port for 2-phase commit. 1-65534
 4  TCP/IP ALREADY VERIFIED  ===> NO  Accept requests containing only a
                                      userid (no password)?  YES, CLIENT,
                                      NO, SERVER, or SERVER_ENCRYPT.
 5  EXTRA BLOCKS REQ    ===> 100      Maximum extra query blocks when DB2 acts
                                      as a requester. 0-100
 6  EXTRA BLOCKS SRV    ===> 100      Maximum extra query blocks when DB2 acts
                                      as a server. 0-100
 7  TCP/IP KEEPALIVE    ===> 120      ENABLE, DISABLE, or 1-65534
 8  POOL THREAD TIMEOUT ===> 120      0-9999 seconds



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 47. Distributed data facility panel: DSNTIP5

DRDA PORT field

The value of the DRDA PORT field determines the TCP/IP port number that is to be used for accepting TCP/IP connection requests from remote DRDA clients.

Acceptable values:	1 to 65534
Default:	none
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	none

If you are enabling data sharing, each member must have the same DRDA port number. You must specify a value for the DRDA PORT field if you plan to use TCP/IP. Leaving this field blank means that you are not using TCP/IP. A blank field is equivalent to using 0 in the change log inventory (DSNJU003) utility.

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

SECURE PORT field

The value of the SECURE PORT field specifies the TCP/IP port number that is to be used for accepting secure TCP/IP connection requests from remote DRDA clients.

Acceptable values:	1 to 65534
Default:	none
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	none

If you are enabling data sharing, each member must have the same secure DRDA port number. You must specify a value if you plan to use TCP/IP with Secure

Socket Layer (SSL). Leaving this field blank means that you are not using TCP/IP with SSL. A blank field is equivalent to using 0 in the change log inventory (DSNJU003) utility.

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

RESYNC PORT field

The value of the RESYNC PORT field specifies the TCP/IP port number that is to be used to process requests for two-phase commit resynchronization.

Acceptable values:	1 to 65534
Default:	none
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	none

This value must be different than the value that is specified for DRDA PORT. If you are enabling data sharing, each member must have a unique resynchronization port. Leaving this field blank means that you are not using TCP/IP. A blank field is equivalent to using 0 in the change log inventory (DSNJU003) utility.

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

TCP/IP ALREADY VERIFIED field (TCPALVER subsystem parameter)

The TCPALVER subsystem parameter specifies whether DB2 is to accept TCP/IP connection requests that contain only a user ID (no password, RACF PassTicket, or Kerberos ticket). Or, this parameter specifies if a stronger form of security is required. This parameter is not relevant to trusted context users that have been switched.

Acceptable values:	YES, CLIENT, NO, SERVER, SERVER_ENCRYPT
Default:	NO (SERVER)
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC TCPALVER
Security parameter:	Yes

YES A new connection is accepted with a user ID only.

CLIENT
This value can be used as an alternative to YES.

NO A user ID and password are required for connection requests, or the connection must be authenticated by a RACF PassTicket or Kerberos ticket. The user ID and password can be encrypted or non-encrypted.

SERVER
This value can be used as an alternative to NO.

SERVER_ENCRYPT
A user ID and password are required for connection requests. Kerberos tickets are also accepted. In addition, one of the following must be true:

- The user ID and password is AES (Advanced Encryption Standard)-encrypted.

- The connection is accepted on a port that ensures AT-TLS (Application Transparent - Transport Layer Security) policy protection, such as a DB2 security port (SECPORT).

Non-encrypted security credentials or RACF PassTickets are not accepted unless the connection is secured by the TCP/IP network. RACF PassTickets are encoded, which is considered to be a form of security that is weaker than encryption. DES (Data Encryption Standard)-based encryption is also considered insecure.

This value must be the same for all members of a data sharing group. This option applies to all incoming requests that use TCP/IP, regardless of the requesting location.

Note: This is a security-related parameter. If the parameter is set to YES or CLIENT, connections are accepted with a user ID only. Security credentials such as a password are not required to authenticate the user ID that is associated with the connection.

Recommendation: Setting the parameter to SERVER_ENCRYPT provides the best security. Connections are accepted only if user credentials are provided to authenticate the user ID, and strong encryption is used to protect the user ID and credentials.

Related tasks:

Managing inbound TCP/IP-based connection requests (Managing Security)

EXTRA BLOCKS REQ field (EXTRAREQ subsystem parameter)

The EXTRAREQ subsystem parameter limits the number of extra DRDA query blocks that DB2 is to request from a remote DRDA server. This parameter does not limit the size of the SQL query answer set. It simply controls the total amount of data that can be transmitted on a network exchange.

Acceptable values:	0 to 100
Default:	100
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP EXTRAREQ

EXTRA BLOCKS SRV field (EXTRASRV subsystem parameter)

The EXTRASRV subsystem parameter limits the number of extra DRDA query blocks that DB2 is to return to a DRDA client. This parameter not limit the size of the SQL query answer set. It simply controls the total amount of data that can be transmitted on a network exchange.

Acceptable values:	0 to 100
Default:	100
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP EXTRASRV

TCP/IP KEEPALIVE field (TCPKPALV subsystem parameter)

The TCPKPALV subsystem parameter determines whether the TCP/IP KeepAlive value in the TCP/IP configuration is to be overridden with a different value. This option can be used in cases where the TCP/IP KeepAlive value in the TCP/IP configuration is not appropriate for the DB2 subsystem.

Acceptable values:	ENABLE, DISABLE, or 1 to 65534
Default:	120
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC TCPKPALV

ENABLE
The TCP/IP KeepAlive configuration value is not overridden.

DISABLE
KeepAlive probing is disabled for this subsystem.

1 to 65534
The TCP/IP KeepAlive configuration value is overridden with the value of the TCP/IP KEEPALIVE field. This value is specified in seconds. Consider setting this value close to the IDLE THREAD TIMEOUT value on installation panel DSNTIPR or the RESOURCE TIMEOUT value on installation panel DSNTIPI.

Avoid using very small values. KeepAlive detection is accomplished by probing the network based on the time that is entered in the KeepAlive parameter. A small KeepAlive value can cause excessive network traffic and system resource consumption. Maintain a proper balance that allows network failures to be detected on a timely basis without a severe impact on system and network performance.

Related tasks:

Timing out idle active threads (DB2 Performance)

POOL THREAD TIMEOUT field (POOLINAC subsystem parameter)

The POOLINAC subsystem parameter specifies the approximate time, in seconds, that a database access thread (DBAT) can remain idle in the pool before it is terminated.

Acceptable values:	0 to 9999
Default:	120
Update:	option 32 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC POOLINAC

A database access thread in the pool counts as an active thread against MAX REMOTE ACTIVE and can hold locks, but does not have any cursors.

Threads are checked every three minutes to see if they have exceeded the timeout value. If the timeout value is less than three minutes, the thread might not be canceled if it has been inactive for more than the timeout value but less than three minutes.

Specifying 0 causes a DBAT to terminate rather than go into the pool if the pool has a sufficient number of threads to process the number of inactive DBATs (type 2 inactive threads) that currently exist.

Routine parameters panel: DSNTIPX

The entries on this panel are used to start the stored procedures address space so that it can run stored procedures or user-defined functions.

```
DSNTIPX         INSTALL DB2 - ROUTINE PARAMETERS
 ===>_

 Enter data below:

  1 WLM PROC NAME    ===> DSN1WLM      WLM-established stored procedure JCL PROC
  2 NUMBER OF TCBS   ===> 8            Number of concurrent TCBs (1-100)
  3 MAX ABEND COUNT  ===> 0            Allowable ABENDs for a routine (0-255)
  4 TIMEOUT VALUE    ===> 180          Seconds to wait before SQL CALL or
                                       function invocation fails (5-1800,NOLIMIT)
  5 WLM ENVIRONMENT  ===>              Default WLM environment name
  6 MAX OPEN CURSORS ===> 500          Maximum open cursors per thread
  7 MAX STORED PROCS ===> 2000         Maximum active stored procs per thread
  8 MAXIMUM LE TOKENS===> 20           Maximum tokens at any time. 0-50
  9 BIF COMPATIBILITY===> CURRENT            Compatibility level for DB2 built-in
                                       functions (V9, V9_DECIMAL_VARCHAR, or
                                       CURRENT)
 10 ADMIN SCHEDULER  ===> DSNADMT      Administrative scheduler task proc name




 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 48. Routine parameters panel: DSNTIPX

Related concepts:

"Migration step 25: Configure DB2 for running stored procedures and user-defined functions" on page 610

"WLM management of stored procedures" on page 839

WLM PROC NAME field

The value of the WLM PROC NAME field specifies a name for the stored procedures JCL procedure that is to be generated during installation. This JCL procedure is generated for the default WLM environment that is specified in the WLM ENVIRONMENT field.

Acceptable values:	1 to 8 alphanumeric characters
Default:	*ssnm*WLM
DSNZP*xxx*:	none

This procedure is used for a WLM-established stored procedures address space.

If this field has a blank, the JCL procedure is still generated. In this case, the JCL procedure will be named by appending the string `WLM` to the DB2 subsystem name (specified on panel DSNTIPM in the field SUBSYSTEM NAME).

NUMBER OF TCBS field

The NUMBER OF TCBS field determines the number of SQL CALL statements and invocations of user-defined functions that can be processed concurrently in the address space of the default WLM environment. The default WLM environment is specified in the WLM ENVIRONMENT field.

Acceptable values:	1 to 100
Default:	8
DSNZP*xxx*:	none

The larger the value, the more stored procedures and user-defined functions you can run concurrently in one address space. This value is dependent on the z/OS UNIX System Services MAXPROCUSER value. If this value is set above the z/OS UNIX System Services MAXPROCUSER value, you may exceed the maximum number of processes for the user.

Related tasks:

Maximizing the number of procedures or functions that run in an address space (DB2 Performance)

Related reference:

"WLM ENVIRONMENT field (WLMENV subsystem parameter)" on page 455

MAX ABEND COUNT field (STORMXAB subsystem parameter)

The STORMXAB subsystem parameter specifies the number of times a stored procedure or an invocation of a user-defined function is to be allowed to terminate abnormally. If this threshold is reached, SQL CALL statements for the stored procedure or user-defined function are rejected.

Acceptable values:	0 to 255
Default:	0
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STORMXAB

The default of 0 means that the first abend of a stored procedure or user defined function causes SQL CALL statements to that procedure or function to be rejected. For production systems, you should accept the default.

TIMEOUT VALUE field (STORTIME subsystem parameter)

The STORTIME subsystem parameter specifies the number of seconds before DB2 is to stop waiting for an SQL CALL statement or invocation of a user-defined function. This parameter applies to user-defined functions that are to be assigned to one of the task control blocks (TCBs) in a DB2 stored procedures address space.

Acceptable values:	5 to 1800, NOLIMIT
Default:	180
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STORTIME

If the time interval expires, the SQL statement fails. The default is a reasonable waiting time for most sites. You might want to choose a higher value if your system has long queues. You might want to choose a lower value if you want to minimize the waiting time for end-user requests. The NOLIMIT value means that DB2 waits indefinitely for the SQL request to complete, while the thread is active.

Recommendation: Do not select the NOLIMIT value. If the stored procedure address space is down for some reason or the user-defined function does not complete, your SQL request hangs until the request is satisfied or the thread is canceled.

WLM ENVIRONMENT field (WLMENV subsystem parameter)

The WLMENV subsystem parameter specifies the name of the default WLM_ENVIRONMENT that is to be used for a user-defined function or stored procedure. This environment is used when a value is not specified for the WLM_ENVIRONMENT option on the CREATE FUNCTION or CREATE PROCEDURE statements.

Acceptable values:	Any valid name from 1 to 32 alphanumeric characters, including underscores
Default:	blank
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP WLMENV

Changing this value does not change existing routines because the value is stored in the catalog when the function or procedure is created.

MAX OPEN CURSORS field (MAX_NUM_CUR subsystem parameter)

The MAX_NUM_CUR subsystem parameter specifies the maximum number of cursors, including allocated cursors, that can be open at a given DB2 site for each thread.

Acceptable values:	0 to 99999
Default:	500
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAX_NUM_CUR

DB2 keeps a count of currently open cursors. If an application attempts to open a thread after the maximum is reached, the statement will fail.

MAX STORED PROCS field (MAX_ST_PROC subsystem parameter)

The MAX_ST_PROC subsystem parameter determines the maximum number of stored procedures that are to be allowed for each thread.

Acceptable values:	0 to 99999
Default:	2000
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAX_ST_PROC

If an application attempts to call a stored procedure after the maximum is reached, the statement will fail.

MAXIMUM LE TOKENS field (LEMAX subsystem parameter)

The LEMAX subsystem parameter specifies the maximum number of Language Environment tokens that are to be active at any time.

Acceptable values:	0 to 50
Default:	20
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM LEMAX

If the value is 0, no tokens are available. A token is used each time one of the following functions is used:

- Trigonometry functions (SIN, SINH, ASIN, COS, COSH, ACOS, TAN, TANH, ATANH, ATAN, and ATAN2)
- DEGREES
- RADIANS
- RAND
- EXP
- POWER®
- Log functions (LOG, and LOG10)
- UPPER
- LOWER
- TRANSLATE
- ROUND_TIMESTAMP
- TRUNC_TIMESTAMP
- LAST_DAY
- NEXT_DAY
- ADD_MONTHS
- Decimal float functions

Related reference:

Scalar functions (DB2 SQL)

BIF COMPATIBILITY field (BIF_COMPATIBILITY subsystem parameter)

DB2 Version 10 introduces a change in the formatting of decimal data by built-in functions CHAR and VARCHAR and CAST(*decimal-expression* AS CHAR) and CAST(*decimal-expression* AS VARCHAR) specifications. The BIF_COMPATIBILITY subsystem parameter specifies whether these built-in functions and specifications are to return results in the Version 10 format or revert to the pre-Version 10 format.

Acceptable values:	V9, V9_DECIMAL_VARCHAR, CURRENT
Default:	for installation: CURRENT; for migration: V9_DECIMAL_VARCHAR
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM.BIF_COMPATIBILITY

In Version 10, when the input data is in decimal format, the CHAR, VARCHAR, CAST(*decimal-expression* AS CHAR), and CAST(*decimal-expression* AS VARCHAR) functions no longer return:

- Leading zeros.
- The trailing decimal point character.
- Leading blanks for positive decimal values. This change applies to the CHAR built-in function only.

If you are migrating from Version 8 or Version 9.1, applications that use these functions or cast specifications can produce unexpected output. You might need to modify some of your applications to handle this change. While you prepare for this change, you can use the following options of the BIF_COMPATIBILITY

subsystem parameter to temporarily revert to the pre-Version 10 behavior. Before using these options, apply the PTFs for APARs PM29124 and PM66095.

V9
For decimal input, the CHAR built-in function returns data in the DB2 Version 9 format. The VARCHAR function and the CAST(*decimal-expression* AS CHAR) and CAST(*decimal-expression* AS VARCHAR) specifications return data in the Version 10 format.

V9_DECIMAL_VARCHAR
For decimal input, the CHAR and VARCHAR functions and the CAST(*decimal-expression* AS CHAR) and CAST(*decimal-expression* AS VARCHAR) specifications return data in the DB2 Version 9 format. When you run the installation CLIST in MIGRATE mode with a DSNTIDXA or DSNTIDXB input member, V9_DECIMAL_VARCHAR is the default setting for this parameter.

CURRENT
For decimal input, the CHAR and VARCHAR functions and the CAST(*decimal-expression* AS CHAR) and CAST(*decimal-expression* AS VARCHAR) specifications return data in the DB2 Version 10 format. When you run the installation CLIST in INSTALL mode, CURRENT is the default setting for this parameter.

The following examples show the effect of the BIF_COMPATIBILITY subsystem parameter on the output of these built-in functions and cast specifications.

Table 78. Example output for CHAR based on BIF_COMPATIBILITY settings

BIF_COMPATIBILITY setting	OUTPUT for CHAR(000.1)	OUTPUT for CHAR(1000.)	OUTPUT for CHAR(1.1)
V9	' 000.1'	' 1000.'	' 1.1'
V9_DECIMAL_VARCHAR	' 000.1'	' 1000.'	' 1.1'
CURRENT	'.1'	'1000'	'1.1'

Table 79. Example output for VARCHAR, CAST(decimal-expression AS CHAR), and CAST(decimal-expression AS VARCHAR) based on BIF_COMPATIBILITY settings

BIF_COMPATIBILITY setting	OUTPUT for VARCHAR(000.1), CAST('000.1' AS CHAR), or CAST('000.1' AS VARCHAR)	OUTPUT for VARCHAR(1000.), CAST('1000.' AS CHAR), or CAST('1000.' AS VARCHAR)	OUTPUT for VARCHAR(1.1), CAST('1.1' AS CHAR), or CAST('1.1' AS VARCHAR)
V9	'.1'	'1000'	'1.1'
V9_DECIMAL_VARCHAR	'0.1'	'1000.'	'1.1'
CURRENT	'.1'	'1000'	'1.1'

Related concepts:

From V8 "Application and SQL release incompatibilities for migration from Version 8" on page 66 From V8

From V9.1 "Application and SQL release incompatibilities for migration from Version 9.1" on page 114 From V9.1

ADMIN SCHEDULER field (ADMTPROC subsystem parameter)

The ADMIN SCHEDULER field identifies a name for the JCL procedure that is used to start the DB2 administrative task scheduler that is associated with this DB2 subsystem.

Acceptable values:	1 to 8 alphanumeric characters
Default:	*ssn*ADMT
Update:	option 33 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ADMTPROC

`1 to 8 alphanumeric characters`
When DB2 starts, it starts the administrative task scheduler that is specified in the ADMIN SCHEDULER field, if that scheduler is not yet started. Also, every time DB2 starts or stops, it posts an event to the administrative task scheduler so that the administrative task scheduler can execute tasks that depend on those events.

`blank`
To disable the administrative task scheduler, enter blanks in the ADMIN SCHEDULER field. If you replace the default value with blanks, DB2 will not start the administrative task scheduler until you update the subsystem parameter.

This parameter cannot be updated online.

Parameters for DB2-supplied routines panel: DSNTIPR1

The DSNTIPR1 panel enables you to establish default WLM environment names for DB2-supplied routines and to access panels for advanced configuration of these routines.

```
DSNTIPR1     INSTALL DB2 PARAMETERS FOR DB2-SUPPLIED ROUTINES
===>
Enter the default WLM environment names for DB2-supplied routines:
  1 FOR MOST DB2 ROUTINES          ===> DSNWLM_GENERAL
  2 FOR ROUTINES IDENTIFIED
    TO PROGRAM CONTROL             ===> DSNWLM_PGM_CONTROL
  3 FOR SERIALIZED ROUTINES        ===> DSNWLM_NUMTCB1

  4 FOR DB2 UTILITIES ROUTINES     ===> DSNWLM_UTILS
  5 FOR DB2 XML PROCESSING ROUTINES===> DSNWLM_XML
  6 FOR DB2 JAVA ROUTINES          ===> DSNWLM_JAVA
  7 FOR DB2 REXX ROUTINES          ===> DSNWLM_REXX
  8 FOR UNIFIED DEBUGGER ROUTINES  ===> DSNWLM_DEBUGGER

  9 FOR THE DB2 CICS ROUTINE       ===> DSNWLM_DSNACICS
 10 FOR DB2 MQSERIES ROUTINES      ===> DSNWLM_MQSERIES
 11 FOR DB2 WEB SERVICES ROUTINES  ===> DSNWLM_WEBSERVICES

 Specify whether to enter advanced configuration of DB2-supplied routines:
 12 ADVANCED CONFIGURATION OPTIONS ===> NO   Enter advanced options (NO or YES)

 PRESS: ENTER to continue    RETURN to exit   HELP for more information
```

Figure 49. Parameters for DB2-supplied routines panel: DSNTIPR1

Values on the DSNTIPR1 panel are used to customize configuration statements in installation job DSNTIJRT, which creates and grants access to the DB2-supplied routines and any related databases, table spaces, tables, indexes, and created global temporary tables. The values are also used to customize installation job DSNTIJRW, which defines core WLM environments for DB2-supplied stored procedures and user-defined functions. If you change an environment name on the DSNTIPR1 panel, the change is applied in the advanced configuration panel settings to all routines that use the original name.

If you change a WLM environment name on this panel, the WLM ENV NAME field of each corresponding routine in the advanced configuration options panels is also changed. For example, if you change the setting of the FOR DB2 XML PROCESSING ROUTINES field from DSNWLM_XML to DSNWLM_MY_XML, then the DSNTIPRJ panel will contain DSNWLM_MY_XML in the WLM ENV NAME fields for all routines.

Changes on the DSNTIPR1 panel do not affect a corresponding routine's WLM ENV NAME field if the WLM ENV NAME value has been previously modified on the advanced configuration options panel. For example, if you change the WLM ENV NAME setting for XSR_ADDSCHEMADOC from DSNWLM_XML to MY_WLM_ENVIRONMENT on the DSNTIPRJ panel, it will remain MY_WLM_ENVIRONMENT even if you subsequently change the setting of the FOR DB2 XML PROCESSING ROUTINES field on the DSNTIPR1 panel from DSNWLM_XML to DSNWLM_MY_XML.

FOR MOST DB2 ROUTINES field

The value of the FOR MOST DB2 ROUTINES field identifies the WLM environment that is to be used for running most DB2-supplied routines.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL
DSNZP*xxx*:	none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR ROUTINES IDENTIFIED TO PROGRAM CONTROL field

The FOR ROUTINES IDENTIFIED TO PROGRAM CONTROL field identifies the WLM environment that is to be used for running DB2-supplied routines that require program control. These routines can be registered to RACF or an equivalent product.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_PGM_CONTROL
DSNZP*xxx*:	none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

Related information:

Defining modules to program control (UNIX System Services Planning)

FOR SERIALIZED ROUTINES field

The value of the FOR SERIALIZED ROUTINES field identifies the WLM environment that is to be used for DB2-supplied routines that need to run serially. A routine that must run serially cannot share a task control block with another routine.

These routines require an environment that has a NUMTCB setting of 1. These routines do not include LANGUAGE REXX routines or SYSPROC.DSNUTILS and SYSPROC.DSNUTILU.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_NUMTCB1
DSNZP*xxx*:	none

FOR DB2 UTILITIES ROUTINES field

The value of the FOR DB2 UTILITIES ROUTINES field identifies the name of the WLM environment that is to be used for running the DB2-supplied routines for utilities. This WLM environment should be used only by the SYSPROC.DSNUTILS and SYSPROC.DSNUTILU stored procedures.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_UTILS
DSNZP*xxx*:	none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR DB2 XML PROCESSING ROUTINES field

The value of the FOR DB2 XML PROCESSING ROUTINES field identifies the WLM environment that is to be used for running most DB2-supplied XML processing routines. This applies to the XSR_ADDSCHEMADOC, XSR_REGISTER, and XSR_REMOVE routines.

Acceptable values:	a valid WLM environment name

Default: DSNWLM_XML
DSNZP*xxx*: none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR MOST DB2 JAVA ROUTINES field

The value of the FOR MOST DB2 JAVA ROUTINES field identifies the WLM environment that is to be used for running most DB2-supplied Java routines.

Acceptable values: a valid WLM environment name
Default: DSNWLM_JAVA
DSNZP*xxx*: none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR DB2 REXX ROUTINES field

The value of the FOR DB2 REXX ROUTINES field identifies the name of the WLM environment that is to be used for DB2-supplied REXX routines.

Acceptable values: a valid WLM environment name
Default: DSNWLM_REXX
DSNZP*xxx*: none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR UNIFIED DEBUGGER ROUTINES field

The FOR UNIFIED DEBUGGER ROUTINES field identifies the WLM environment that is to be used for DB2-supplied routines that are for the unified debugger tool.

Acceptable values: a valid WLM environment name
Default: DSNWLM_DEBUGGER
DSNZP*xxx*: none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR THE DB2 CICS ROUTINE field

The value of the FOR THE DB2 CICS ROUTINE field identifies the WLM environment that is to be used for the DB2 CICS transaction processor routine SYSPROC.DSNACICS.

Acceptable values: a valid WLM environment name
Default: DSNWLM_CICS
DSNZP*xxx*: none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR DB2 MQSERIES ROUTINES field

The value of the FOR DB2 MQSERIES ROUTINES field specifies the name of the WLM environment that is to be used for DB2 MQSeries functions.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_MQSERIES
DSNZP*xxx*:	none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

FOR DB2 WEB SERVICES ROUTINES field

The value of the FOR DB2 WEB SERVICES ROUTINES field identifies the WLM environment that is to be used for DB2 web services functions.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_WEBSERVICES
DSNZP*xxx*:	none

Related reference:

"DB2-supplied stored procedures and user-defined functions" on page 833

ADVANCED CONFIGURATION OPTIONS field

The value of the ADVANCED CONFIGURATION OPTIONS field determines whether optional panels that are for advanced configuration of DB2-supplied stored procedures and functions are to be displayed.

Acceptable values:	NO, YES
Default:	NO
DSNZP*xxx*:	none

NO Skips the panels for advanced configuration of stored procedures and displays the DSNTIPZ panel.

YES
Displays panels for advanced configuration of DB2-supplied stored procedures, starting with panel DSNTIPRA.

DB2 routine selection panel: DSNTIPRA

The DSNTIPRA panel provides access to additional panels for advanced configuration options of DB2-supplied routines. This panel is displayed if you specify YES in the ADVANCED CONFIGURATION OPTIONS field on the DSNTIPR1 panel.

If you specify NO in the ADVANCED CONFIGURATION OPTIONS field, the DSNTIPZ panel is displayed instead.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a

scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+--------------- DB2 ROUTINES: ADVANCED CONFIGURATION  ---------------+
| DSNTIPRA           DB2 Routine Selection Panel                      |
| Option ===>                                                         |
|                                                                     |
| 1 Trace formatting routines                                         |
| 2 Utility invocation routines                                       |
| 3 Runtime statistics processing routines                            |
| 4 DB2 security routines                                             |
| 5 Java and SQL procedure processing routines                        |
| 6 Unified debugger for Java and SQL routines                        |
| 7 XML schema processing routines                                    |
| 8 IBM Text Search routines                                          |
| 9 Database metadata routines                                        |
| 10 DB2 administration routines                                      |
| 11 DB2 query optimization routines                                  |
| 12 CICS and IMS transaction invocation routines                     |
| 13 MQSeries routines                                                |
| 14 Web services/SOAP routines                                       |
|                                                                     |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit            |
+---------------------------------------------------------------------+
```

Figure 50. DB2 routine selection panel: DSNTIPRA

To view and modify the advanced options for a routine type:
In the Option field, enter the number of the routine type, and then press the Enter key.

To exit advanced configuration and display the next standard installation panel, DSNTIPZ:
Leave the Option field blank and press the Enter key.

To return to the DSNTIPRA panel:
Press the Return key.

Trace formatting routines panel: DSNTIPRB

The DSNTIPRB panel allows you to modify configuration options for trace formatting routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRB           Trace Formatting routines                      |
| ===>                                                              |
| DSNWSPM                                                           |
|   WLM ENV NAME  ===> DSNWLM_NUMTCB1                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DSNWZP                                                            |
|   WLM ENV NAME  ===> DSNWLM_NUMTCB1                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 51. Trace formatting routines panel: DSNTIPRB

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_UTILS
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

Utility invocation routines panel: DSNTIPRC

The DSNTIPRC panel allows you to modify configuration options for utility invocation routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN

and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRC          Utility invocation routines                     |
| ===>                                                              |
| DSNUTILS                                                          |
|   WLM ENV NAME  ===> DSNWLM_UTILS                                 |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DSNUTILU                                                          |
|   WLM ENV NAME  ===> DSNWLM_UTILS                                 |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 52. Utility invocation routines panel: DSNTIPRC

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_UTILS
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

Runtime statistics processing routines panel: DSNTIPRD

The DSNTIPRD panel allows you to modify configuration options for runtime statistics processing routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel

can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRD    Runtime statistics processing routines                |
| ===>                                                              |
|                                                                   |
| DSNACCOR                                                          |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| STATS_MONITOR                                                     |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 53. Runtime statistics processing routines panel: DSNTIPRD

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

DB2 security routines panel: DSNTIPRE

The DSNTIPRE panel allows you to modify configuration options for the SYSIBM.USERNAMES encryption routine.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a

scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRE                 DB2 security routines                   |
| ===>                                                             |
|                                                                  |
| DSNLEUSR                                                         |
|   WLM ENV NAME  ===> DSNWLM_NUMTCB1                              |
|   GRANT EXECUTE ===> PUBLIC                                      |
|   PACKAGE OWNER ===>                                             |
|                                                                  |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit         |
+------------------------------------------------------------------+
```

Figure 54. DB2 security routines panel: DSNTIPRE

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_NUMTCB1
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

Java and SQL procedure processing routines panel: DSNTIPRF

The DSNTIPRF panel allows you to modify configuration options for SQL procedure processing routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split

screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRF   Java and SQL procedure processing routines             |
| ===>                                                              |
|                                                                   |
|ALTER_JAVA_PATH                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DB2_INSTALL_JAR                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DB2_REMOVE_JAR                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DB2_REPLACE_JAR                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DB2_UPDATEJARINFO                                                 |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| INSTALL_JAR                                                       |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| REMOVE_JAR                                                        |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| REPLACE_JAR                                                       |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DSNTBIND                                                          |
|   WLM ENV NAME  ===> DSNWLM_REXX                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DSNTPSMP                                                          |
|   WLM ENV NAME  ===> DSNWLM_REXX                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
| WLM_REFRESH                                                       |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 55. Java and SQL procedure processing routines panel: DSNTIPRF

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL for most routines, DSNWLM_REXX for DSNTBIND and DSNTPSMP

DSNZP*xxx*: none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values: at least one valid authorization ID
Default: PUBLIC
DSNZP*xxx*: none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values: blank, or a valid authorization ID
Default: blank
DSNZP*xxx*: none

Unified debugger for Java and SQL routines panel: DSNTIPRH

The DSNTIPRH panel allows you to modify configuration options for unified debugger for SQL and Java routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRH   Unified debugger for Java and SQL routines             |
| ===>                                                              |
| CREATE_SESSION                                                    |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DEBUGGERLEVEL                                                     |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DESTROY_SESSION                                                   |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| GET_REPORT                                                        |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| LIST_SESSION                                                      |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| PUT_COMMAND                                                       |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| QUERY_SESSION                                                     |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_ENDSESSIONMANAGER                                             |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_INITIALIZECLIENT                                              |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_LOOKUPSESSIONMANAGER                                          |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_PINGSESSIONMANAGER                                            |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_RECVCLIENTREPORTS                                             |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_RUNSESSIONMANAGER                                             |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_SENDCLIENTCOMMANDS                                            |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_SENDCLIENTREQUESTS                                            |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
| DBG_TERMINATECLIENT                                               |
|   WLM ENV NAME  ===> DSNWLM_DEBUGGER                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 56. Unified debugger for Java and SQL routines panel: DSNTIPRH

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_DEBUGGER for most routines, DSNWLM_GENERAL for DBG_RUNSESSIONMANAGER
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

XML schema processing routines panel: DSNTIPRI

The DSNTIPRI panel allows you to modify configuration options for XML schema processing routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRI         XML schema processing routines                   |
| ===>                                                              |
|                                                                   |
| XSR_ADDSCHEMADOC                                                  |
|   WLM ENV NAME  ===> DSNWLM_XML                                   |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| XSR_COMPLETE                                                      |
|   WLM ENV NAME  ===> DSNWLM_JAVA                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| XSR_REGISTER                                                      |
|   WLM ENV NAME  ===> DSNWLM_XML                                   |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| XSR_REMOVE                                                        |
|   WLM ENV NAME  ===> DSNWLM_XML                                   |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 57. XML schema processing routines panel: DSNTIPRI

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_XML for most routines, DSNWLM_JAVA for XSR_COMPLETE
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

IBM Text Search routines panel: DSNTIPRJ

The DSNTIPRJ panel allows you to modify configuration options for Text Search routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRJ            IBM Text Search routines                      |
| ===>                                                              |
| SYSTS_ALTER                                                       |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_CREATE                                                      |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_DROP                                                        |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_ENCRYPT                                                     |
|   WLM ENV NAME  ===> DSNWLM_JAVA                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_RESTORE                                                     |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_START                                                       |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_STOP                                                        |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_TAKEOVER                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SYSTS_UPDATE                                                      |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 58. IBM Text Search routines panel: DSNTIPRJ

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL (DSNWLM_JAVA for SYSTS_ENCRYPT)
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

Database metadata routines panel: DSNTIPRK

The DSNTIPRK panel allows you to modify configuration options for database metadata routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRK           Database metadata routines                     |
| ===>                                                              |
|                                                                   |
| SQLCAMESSAGE                                                      |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
| SQLCOLPRIVILEGES                                                  |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLCOLUMNS                                                        |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLFOREIGNKEYS                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLFUNCTIONCOLS                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLFUNCTIONS                                                      |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLGETTYPEINFO                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLPRIMARYKEYS                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLPROCEDURECOLS                                                  |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLPSEUDOCOLUMNS                                                  |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLPROCEDURES                                                     |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLSPECIALCOLUMNS                                                 |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLSTATISTICS                                                     |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLTABLEPRIVILEGES                                                |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLTABLES                                                         |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| SQLUDTS                                                           |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 59. Database metadata routines panel: DSNTIPRK

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

DB2 administration routines panel: DSNTIPRL

The DSNTIPRL panel allows you to modify configuration options for routines that are to be used for administration.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRL          DB2 administration routines                     |
| ===>                                                              |
| ADMIN_COMMAND_DB2                                                 |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_COMMAND_DSN                                                 |
|   WLM ENV NAME  ===> DSNWLM_REXX                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_COMMAND_UNIX                                                |
|   WLM ENV NAME  ===> DSNWLM_PGM_CONTROL                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_BROWSE                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_DELETE                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_LIST                                                     |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_RENAME                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_SEARCH                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_DS_WRITE                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_INFO_HOST                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_INFO_SMS                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_INFO_SQL                                                    |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_INFO_SSID                                                   |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_INFO_SYSLOG                                                 |
|   WLM ENV NAME  ===> DSNWLM_REXX                                  |
|   GRANT EXECUTE ===> PUBLIC                                       |
| ADMIN_INFO_SYSPARM                                                |
|   WLM ENV NAME  ===> DSNWLM_NUMTCB1                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| ADMIN_JOB_CANCEL                                                  |
|   WLM ENV NAME  ===> DSNWLM_PGM_CONTROL                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
```

Figure 60. DB2 administration routines panel: DSNTIPRL (part 1 of 3)

```
ADMIN_JOB_FETCH
  WLM ENV NAME  ===> DSNWLM_PGM_CONTROL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_JOB_QUERY
  WLM ENV NAME  ===> DSNWLM_PGM_CONTROL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_JOB_SUBMIT
  WLM ENV NAME  ===> DSNWLM_PGM_CONTROL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_ADD
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_CANCEL
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_LIST
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_OUTPUT
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_REMOVE
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_STATUS
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_TASK_UPDATE
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_UTL_EXECUTE
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_UTL_MODIFY
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_UTL_MONITOR
  WLM ENV NAME  ===> DSNWLM_PGM_CONTROL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_UTL_SCHEDULE
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
ADMIN_UTL_SORT
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
DSN_WLM_APPLENV
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
DSNACCOX
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
```

Figure 61. DB2 administration routines panel: DSNTIPRL (part 2 of 3)

```
GET_CONFIG
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
GET_MESSAGE
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
GET_SYSTEM_INFO
  WLM ENV NAME  ===> DSNWLM_NUMTCB1
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>
WLM_SET_CLIENT_INFO
  WLM ENV NAME  ===> DSNWLM_GENERAL
  GRANT EXECUTE ===> PUBLIC
  PACKAGE OWNER ===>

PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit
```

Figure 62. DB2 administration routines panel: DSNTIPRL (part 3 of 3)

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	see the panel illustration for the default environment name of each routine
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

DB2 optimization routines panel: DSNTIPRM

The DSNTIPRM panel allows you to modify configuration options for routines that are to be used for optimization.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a

scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRM            DB2 optimization routines                      |
| ===>                                                               |
| DSNAEXP                                                            |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                                |
|   GRANT EXECUTE ===> user-id                                       |
|   PACKAGE OWNER ===> user-id                                       |
| DSNAHVPM                                                           |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                                |
|   GRANT EXECUTE ===> user-id                                       |
|   PACKAGE OWNER ===> user-id                                       |
|                                                                    |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit           |
+--------------------------------------------------------------------+
```

Figure 63. DB2 query optimization routines panel: DSNTIPRM

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

The setting *user-id* is the TSO user ID used to run the installation CLIST.

Acceptable values:	at least one valid authorization ID
Default:	user-id
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

The setting *user-id* is the TSO user ID used to run the installation CLIST.

Acceptable values:	blank, or a valid authorization ID
Default:	user-id
DSNZP*xxx*:	none

CICS and IMS transaction invocation routines panel: DSNTIPRN

The DSNTIPRN panel allows you to modify configuration options for CICS and IMS transaction invocation routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRN CICS and IMS transaction invocation routines             |
| ===>                                                              |
| DSNACICS                                                          |
|   WLM ENV NAME  ===> DSNWLM_DSNACICS                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DSNAIMS                                                           |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| DSNAIMS2                                                          |
|   WLM ENV NAME  ===> DSNWLM_GENERAL                               |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 64. CICS and IMS transaction invocation routines panel: DSNTIPRN

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_GENERAL (DSNWLM_DSNACICS for DSNACICS)
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for

most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

MQSeries routines panel: DSNTIPRO

The DSNTIPRO panel allows you to modify configuration options for MQSeries routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRO              MQSeries routines                           |
| ===>                                                              |
| MQREAD                                                            |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQREADALL                                                         |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQREADALLCLOB                                                     |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQREADCLOB                                                        |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQRECEIVE                                                         |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQRECEIVEALL                                                      |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQRECEIVEALLCLOB                                                  |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQRECEIVECLOB                                                     |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
| MQSEND                                                            |
|   WLM ENV NAME  ===> DSNWLM_MQSERIES                              |
|   GRANT EXECUTE ===> PUBLIC                                       |
|   PACKAGE OWNER ===>                                              |
|                                                                   |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 65. MQSeries routines panel: DSNTIPRO

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_MQSERIES
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID
Default:	PUBLIC
DSNZP*xxx*:	none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for

most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values:	blank, or a valid authorization ID
Default:	blank
DSNZP*xxx*:	none

Web services/SOAP routines panel: DSNTIPRP

The DSNTIPRP panel allows you to modify configuration options for web services and SOAP routines.

Because space on this panel is limited, all fields might not be displayed at one time. The number of fields that are displayed depends on the maximum number of rows your ISPF display device can display at one time, whether you use split screen mode, and where you choose to split the screen. This panel is defined as a scrollable panel, which enables you to view all the fields if they are not displayed at one time. The panel can be scrolled when the label "More" displays near the upper right corner of the panel. A plus symbol (+) following "More" indicates that the panel can be scrolled down, and a minus symbol (-) indicates that the panel can be scrolled up. You can use the ISPF scroll commands DOWN and UP to scroll through the fields. (Alternatively, you can assign your PF keys to be the DOWN and UP commands.)

```
+---------- DB2 ROUTINES: ADVANCED CONFIGURATION OPTIONS -----------+
| DSNTIPRP          Web services/SOAP routines                      |
| ===>                                                              |
|                                                                   |
| SOAPHTTPC                                                         |
|   WLM ENV NAME  ===> DSNWLM_WEBSERVICES                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
| SOAPHTTPNC                                                        |
|   WLM ENV NAME  ===> DSNWLM_WEBSERVICES                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
| SOAPHTTPNV                                                        |
|   WLM ENV NAME  ===> DSNWLM_WEBSERVICES                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
| SOAPHTTPV                                                         |
|   WLM ENV NAME  ===> DSNWLM_WEBSERVICES                           |
|   GRANT EXECUTE ===> PUBLIC                                       |
|PRESS: ENTER to continue UP/DOWN to scroll RETURN to exit          |
+-------------------------------------------------------------------+
```

Figure 66. Web services/SOAP routines panel: DSNTIPRP

WLM ENV NAME

Specify the WLM environment name for the routine.

Acceptable values:	a valid WLM environment name
Default:	DSNWLM_WEBSERVICES
DSNZP*xxx*:	none

GRANT EXECUTE

Specify one or more authorization IDs that are to be granted EXECUTE access on the routine. Use commas to separate individual IDs. Do not use embedded blanks. You can enter up to 44 characters, including commas.

Acceptable values:	at least one valid authorization ID

Default: PUBLIC
DSNZP*xxx*: none

PACKAGE OWNER

Specify the ID that should own the package when it is bound (the value for the OWNER parameter of the BIND PACKAGE command). The default for most routines is blank, meaning that no OWNER parameter is included in the BIND PACKAGE command. This field does not display for routines that do not have package.

Acceptable values: blank, or a valid authorization ID
Default: blank
DSNZP*xxx*: none

Data definition control support panel: DSNTIPZ

The entries on the DSNTIPZ panel allow you to install and tailor data definition control support.

Two SQL tables (application registration and object registration) are identified and created even if data definition control support is not installed. This simplifies future activation of the facility. Specified application identifiers (DB2 plans or collections of packages) can be registered in the application registration table, and, optionally, their associated DB2 object names can be registered in the object registration table. DB2 consults these two tables prior to accepting a given DDL statement to make sure that a particular application identifier and object name are registered.

```
DSNTIPZ      INSTALL DB2 - DATA DEFINITION CONTROL SUPPORT
===>

Enter data below:

 1 INSTALL DD CONTROL SUPT  ===> NO        YES - activate the support
                                           NO  - omit DD control support
 2 CONTROL ALL APPLICATIONS ===> NO        YES or NO
 3 REQUIRE FULL NAMES       ===> YES       YES or NO
 4 UNREGISTERED DDL DEFAULT ===> ACCEPT    Action for unregistered DDL:
                                           ACCEPT - allow it
                                           REJECT - prohibit it
                                           APPL   - consult ART
 5 ART/ORT ESCAPE CHARACTER ===>           Used in ART/ORT Searches
 6 REGISTRATION OWNER       ===> DSNRGCOL          Qualifier for ART and ORT
 7 REGISTRATION DATABASE    ===> DSNRGFDB          Database name
 8 APPL REGISTRATION TABLE  ===> DSN_REGISTER_APPL Table name
 9 OBJT REGISTRATION TABLE  ===> DSN_REGISTER_OBJT Table name

Note: ART = Application Registration Table
      ORT = Object      Registration Table

PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 67. Data definition control support panel: DSNTIPZ

Related information:

Managing access through data definition control (Managing Security)

INSTALL DD CONTROL SUPT field (RGFINSTL subsystem parameter)

The RGFINSTL subsystem parameter determines whether data definition control support is to be installed.

Acceptable values:	YES, NO
Default:	NO
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFINSTL

YES
Data definition control support is installed.

NO Data definition control support is not installed. DDL statements are not validated by this support. The application registration table and object registration table are still created according to values that are entered in fields 5 through 8 on panel DSNTIPZ.

CONTROL ALL APPLICATIONS field (RGFDEDPL subsystem parameter)

The RGFDEDPL subsystem parameter specifies whether the DB2 subsystem is to be completely controlled by a set of closed applications that are identified in the application registration table.

Acceptable values:	YES, NO
Default:	NO
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFDEDPL

Closed applications require their DB2 objects to be managed solely through the plans or packages of the closed application that is registered in the application registration table.

REQUIRE FULL NAMES field (RGFFULLQ subsystem parameter)

The RGFFULLQ subsystem parameter specifies whether registered objects are to require fully qualified names.

Acceptable values:	YES, NO
Default:	YES
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFFULLQ

UNREGISTERED DDL DEFAULT field (RGFDEFLT subsystem parameter)

The RGFDEFLT subsystem parameter specifies the action that is to be taken for DDL statements that name unregistered objects.

Acceptable values:	ACCEPT, REJECT, APPL
Default:	ACCEPT
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFDEFLT

ACCEPT
DDL statements that name unregistered objects are accepted.

REJECT
DDL statements that name unregistered objects are rejected.

APPL
DDL statements that name unregistered objects are rejected if the current application is not registered.

ART/ORT ESCAPE CHARACTER field (RGFESCP subsystem parameter)

The RGFESCP subsystem parameter specifies the escape character that is to be used in the application registration table (ART) or object registration table (ORT).

Acceptable values:	any non-alphanumeric character
Default:	none
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFESCP

Sets of names in the ART and ORT can be represented by patterns that use the underscore (_) and percent sign (%) characters in the same way as in an SQL LIKE predicate.

If you enter a character in this field, it can be used in those patterns in the same way as an escape character is used in an SQL LIKE predicate.

Related tasks:

Installing data definition control support (Managing Security)

REGISTRATION OWNER field (RGFCOLID subsystem parameter)

The RGFCOLID subsystem parameter specifies the ID that is to own both the application registration table and the object registration table.

Acceptable values:	1 to 8 characters
Default:	DSNRGCOL
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFCOLID

REGISTRATION DATABASE field (RGFDBNAM subsystem parameter)

The RGFDBNAM subsystem parameter specifies the name of the database that is to contain the registration tables that are used for data definition control support.

Acceptable values:	1 to 8 characters
Default:	DSNRGFDB
Update:	option 35 on panel DSNTIPB

DSNZP*xxx*: DSN6SPRM RGFDBNAM

APPL REGISTRATION TABLE field (RGFNMPRT subsystem parameter)

The RGFNMPRT subsystem parameter specifies the name that is to be used for the application registration table.

Acceptable values:	1 to 17 characters
Default:	DSN_REGISTER_APPL
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFNMPRT

OBJT REGISTRATION TABLE field (RGFNMORT subsystem parameter)

The RGFNMORT subsystem parameter specifies the name that is to be used for the object registration table (ORT).

Acceptable values:	1 to 17 characters
Default:	DSN_REGISTER_OBJT
Update:	option 35 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFNMORT

Job editing panel: DSNTIPY

The entries on the DSNTIPY panel specify values and information about job statements for the installation and sample application jobs.

```
DSNTIPY         INSTALL DB2 - JOB EDITING
===>

Enter data below:

 1  REMOTE LOCATION   ===>                  Remote location for COBOL
                                            organization application


Enter job card information for install and sample jobs:

 2 ===>
 3 ===>
 4 ===>
 5 ===>
 6 ===>
 7 ===>




PRESS:  ENTER to continue  RETURN to exit   HELP for more information
```

Figure 68. Job editing panel: DSNTIPY

Related concepts:

"System affinity for installation jobs" on page 224

"Multiple procedure libraries" on page 223

Related reference:

JCLLIB statement (MVS JCL Reference)

JES2 Control Statements (MVS JCL Reference)

JES3 Control Statements (MVS JCL Reference)

REMOTE LOCATION field

The REMOTE LOCATION field specifies the location of another DB2 subsystem. The remote subsystem is used by the COBOL preparation sample job DSNTEJ3C, the DDF remote location update sample job DSNTEJ6, and the stored procedures sample jobs. The stored procedures sample jobs that require a remote location name are DSNTEJ6D, DSNTEJ6P, DSNTEJ6S, DSNTEJ6T, DSNTEJ61, and DSNTEJ62.

Acceptable values:	1 to 16 characters
Default:	none
Update:	See "The update process" on page 501
DSNZP*xxx*:	none

The location name must begin with a letter and must not contain special characters. A remote location name is accepted only if you have also entered a DB2 location name for the DB2 LOCATION NAME field on installation panel DSNTIPR.

Related reference:

"DB2 LOCATION NAME field" on page 444

Job card information field

The job card information fields specify the job statements that are to be used in all the installation and sample application jobs.

Acceptable values:	see below
Default:	none
Update:	option 36 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the job name in one of two ways:

- If the job name is *member*, the job name for each job is the same as its member name.
- If the job name is any value other than *member*, the name is truncated to seven characters, and one character is added to the end of the name to identify the run order for that job.

An example of job card information follows:

```
3====> //MEMBER JOB,
4====> // MSGLEVEL=(1,1),MSGCLASS=H,REGION=2M,CLASS=A
5====> // USER=SYSADM,PASSWORD=SYSADM,NOTIFY=SYSADM
6====>
```

CLIST calculations panel 1: DSNTIPC

The DSNTIPC panel displays the messages that the installation CLIST issues to indicate calculated storage sizes.

Space estimates from these messages do not account for cylinder rounding. Base requirements can be 10% - 20% higher than the message indicates, depending on the disk type.

The messages show that most of the needed virtual storage is in extended private storage and storage above the 2 GB bar, including the buffer pool, the EDM pool, and almost all of the code and working storage.

During the tailoring session, message DSNT438I is issued to the tailoring terminal. This message, which shows the IRLM LOCK MAXIMUM SPACE and MAX STORAGE FOR LOCKS values, is always issued if you accept the default values.

This message indicates that the IRLM could request a total amount of space that is larger than the available space, causing an abend. The message is based on the following variables:

- The maximum number of data or row locks per user that is specified on installation panel DSNTIPJ (LOCKS PER USER)
- The number of users that is specified on installation panel DSNTIPE for MAX USERS and MAX REMOTE ACTIVE

The CLIST assumes that the private region that is available for IRLM locks is estimated as 60,000 KB.

```
DSNTIPC         INSTALL DB2 - CLIST CALCULATIONS - PANEL 1
===>
 You can update the DSMAX, EDMPOOL STATEMENT CACHE (if CACHE DYNAMIC is YES),
 EDM DBD CACHE, SORT POOL, and RID POOL sizes if necessary.
                                      Calculated   Override
 1  DSMAX - MAXIMUM OPEN DATA SETS    =    20000                (1-100000)
 2  DSNT485I  EDM STATEMENT CACHE     =   113386  K            K
 3  DSNT485I  EDM DBD CACHE           =    23400  K            K
 4  DSNT485I  EDM SKELETON POOL SIZE  =    10240  K            K
 5  DSNT485I  EDM LIMIT BELOW THE BAR =        0  K            K
 6  DSNT485I  BUFFER POOL SIZE        =      101  M
 7  DSNT485I  SORT POOL SIZE          =    10000  K            K
 8  DSNT485I  RID POOL SIZE           =   400000  K            K
 9  DSNT485I  DATA SET STORAGE SIZE   =    17928  K
10  DSNT485I  CODE STORAGE SIZE       =    38200  K
11  DSNT485I  WORKING STORAGE SIZE    =    55800  K
12  DSNT486I  TOTAL MAIN STORAGE      =      238  M
13  DSNT487I  TOTAL STORAGE BELOW 16M =     1159  K WITH SWA ABOVE 16M LINE
14  DSNT438I  IRLM LOCK MAXIMUM SPACE =     2160  M, AVAILABLE =   2160 M


PRESS:  ENTER to continue RETURN to exit   HELP for more information
```

Figure 69. CLIST calculations panel 1: DSNTIPC

Related tasks:

Estimating the storage needed for locks (DB2 Performance)

Related information:

DSNT438I (DB2 Messages)

DSMAX field (DSMAX subsystem parameter)

The DSMAX subsystem parameter determines the maximum number of data sets that is to be allowed open at one time.

Acceptable values:	1 to 100000
Default:	based on calculations
Update:	enter a value in the override column
DSNZP*xxx*:	DSN6SPRM DSMAX

Although the maximum number of data sets is 100,000, the practical limit can be less than 30,000, depending on available storage below the line. The value that you enter can substantially influence the performance of DB2.

If you use z/OS Version 1 Release 12 or later, you should complete one of the following actions to improve the performance of opening and closing data sets:

- Update the ALLOC*xx* parmlib member to set the SYSTEM MEMDSENQMGMT value to ENABLE.
- Issue system command SETALLOC SYSTEM,MEMDSENQMGMT=ENABLE.

Updating the ALLOC*xx* parmlib is strongly recommended because the change remains effective across IPLs. If the SETALLOC command is used to enable SYSTEM MEMDSENQMGM, a DB2 restart is required to make the change effective.

When a secondary index is nonpartitioned, the number of data sets that are required for the index is dependent on the total required space to contain the tree structure and on the size limit of each data set. When a secondary index is data-partitioned, the number of data sets that are required for the index is equal to the number of data partitions in the table space that contains the table. Unless a small piece size is used for nonpartitioned secondary indexes, partitioning generally results in an increase in the number of data sets for the index.

If the partitioning of secondary indexes causes the number of data sets to increase or decrease appreciably, you can modify the value of DSMAX. The default value for DSMAX is calculated by DB2 and does not count partitioned objects. Choose the value for DSMAX according to the impact that partitioning secondary indexes has on the number of data sets for those objects. DB2 defers closing and deallocating table spaces or indexes until the number of open data sets reaches the operating system limit or approaches the value that is specified by DSMAX.

Related concepts:

How DB2 determines the initial value of DSMAX (DB2 Performance)

Related tasks:

Managing the opening and closing of data sets (DB2 Performance)

Evaluating the value of DSMAX (DB2 Performance)

EDM STATEMENT CACHE field (EDMSTMTC subsystem parameter)

The EDMSTMTC subsystem parameter determines the size (in KB) of the statement cache that is to be used by the EDM.

Acceptable values:	5000 to 1048576
Default:	113386
Update:	press ENTER on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMSTMTC

This value is used at DB2 startup time as the minimum value. It can be increased and subsequently decreased with the SET SYSPARM command. This value cannot be decreased below the value that is specified at DB2 startup. The CLIST calculates a statement cache size. This storage pool is located above the 2 GB bar. You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC; the CLIST calculates this value based on input from previous panels. If a value is in the Override column, you must erase the override value in order to accept the calculated value.
- Entering your own value in the Override column of panel DSNTIPC.

Related tasks:

"Calculating EDM pool size" on page 56

Related reference:

-SET SYSPARM (DB2) (DB2 Commands)

EDM DBD CACHE field (EDMDBDC subsystem parameter)

The EDMDBDC subsystem parameter determines the minimum size (in KB) of the DBD cache that is to be used by the environmental descriptor manager (EDM).

Acceptable values:	5000 to 2097152
Default:	23400
Update:	press ENTER on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMDBDC

The default value for EDM DBD CACHE is the larger of the EDM calculated value or 5000 K.

This value is used at DB2 startup time as the minimum value. It can be increased and subsequently decreased with the SET SYSPARM command. This value cannot be decreased below the value that is specified at DB2 startup. This storage pool is located above the 2 GB bar. The CLIST calculates the DBD cache size.

You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the calculated value.
- Entering your own value in the Override column of panel DSNTIPC.

EDM SKELETON POOL SIZE field (EDM_SKELETON_POOL subsystem parameter)

The EDM_SKELETON_POOL subsystem parameter determines the minimum size of the EDM skeleton pool in KB.

Acceptable values:	5120 to 2097152
Default:	10240
Update:	press ENTER on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDM_SKELETON_POOL

You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC; the CLIST calculates this value based on input from previous panels. If there is a value in the Override column, you must erase the override value in order to accept the calculated value.
- Entering your own value in the Override column of panel DSNTIPC.

The setting is used at DB2 start time and it can be increased or decreased to the startup value by using the SET SYSPARM command. The skeleton pool, which is located above the bar, is a component of EDM for storing skeleton package tables (SKPTs), where each SKPT represents a package.

Related tasks:

"Calculating EDM pool size" on page 56

Related reference:

-SET SYSPARM (DB2) (DB2 Commands)

EDM LIMIT BELOW THE BAR (EDMPOOL subsystem parameter)

The EDMPOOL subsystem parameter determines the maximum KB of storage below the 2 GB bar that can be consumed by the environmental descriptor manager (EDM). A value of 0 means that 31-bit EDM consumption is unbounded.

Acceptable values:	0 to 2097152
Default:	0
Update:	press ENTER twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMPOOL

EDM does not allocate storage below the bar except to process a package or plan that was last bound or rebound prior to Version 10. If you are migrating to Version 10, override the default setting with the EDMPOOL value that you currently use in the previous release.

Recommendation: To eliminate use of storage below the bar, rebind all packages and plans after DB2 enters new-function mode.

If the specified limit is reached or a new request would otherwise exceed the limit, DB2 issues a resource unavailable message, and the requested operation is not performed.

The EDMPOOL parameter can be changed online. When you dynamically update EDMPOOL by using the SET SYSPARM command, you can set EDMPOOL to the maximum allowable value of 2097152 KB. The new limit will be used for future requests.

Related tasks:

"Calculating EDM pool size" on page 56

BUFFER POOL SIZE field

The value of the BUFFER POOL SIZE field is the buffer pool size that is calculated by the CLIST.

Acceptable values:	none
Default:	101 M
Update:	run CLIST again
DSNZP*xxx*:	none

This field is protected and cannot be changed during update processing. If you want to change the size of a buffer pool, you must use the command ALTER BUFFERPOOL.

Related reference:

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

SORT POOL SIZE field (SRTPOOL subsystem parameter)

The SRTPOOL subsystem parameter specifies the amount of storage in kilobytes that is needed for the sort pool.

Acceptable values:	240 to 128000
Default:	10000
Update:	press Enter twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SRTPOOL

You have a choice of:

- Accepting the default value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the default value.
- Typing your own value in the Override column of panel DSNTIPC.

If you decide to change this field, estimate the sort pool value by using the following formula:

```
32000 * (16 + sort key length + sort data length)
```

Related tasks:

Choosing buffer pool sizes (DB2 Performance)

RID POOL SIZE field (MAXRBLK subsystem parameter)

The MAXRBLK subsystem parameter specifies the amount of storage in kilobytes that is needed for the RID pool.

Acceptable values:	0, 128 to 10000000
Default:	400000
Update:	press Enter twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAXRBLK

You have a choice of:

- Accepting the default value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the default value.
- Typing your own value in the Override column of panel DSNTIPC.

If you decide to change this field, estimate the storage that is required for the RID pool by using the following formula:

```
Number of concurrent RID processing activities *
average number of RIDs * 2 * 5 (bytes per RID)
```

Choosing 0 disables the use of the RID pool. In this case, DB2 does not use access paths or join methods that depend on RID pool storage.

This storage is located above the 2 GB bar.

Related tasks:

Increasing RID pool size (DB2 Performance)

CLIST messages fields for panel DSNTIPC

The CLIST messages fields display storage sizes that are calculated by the CLIST. These fields are protected and cannot be changed by the user.

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

Storage messages fields

The storage messages fields indicate the results of the calculations described on panel DSNTIPC. These fields are protected and cannot be changed by a user.

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

CLIST calculations panel 2: DSNTIPC1

The DSNTIPC1 panel displays CLIST messages that indicate a variety of calculations.

```
 DSNTIPC1        INSTALL DB2 - CLIST CALCULATIONS - PANEL 2
===> _

1  DSNT488I  DATA SETS CREATED BY THE INSTALLATION CLIST
             WILL REQUIRE AT LEAST 4000 KILOBYTES (83 TRACKS)
2  DSNT488I  OTHER NON-VSAM DATA SETS CREATED BY INSTALLATION AND IVP JOBS
             WILL REQUIRE AT LEAST 4000 KILOBYTES (83 TRACKS)
3  DSNT488I  DATA SETS FOR STORAGE GROUPS CREATED BY INSTALLATION AND IVP JOBS
             WILL REQUIRE AT LEAST 44360 KILOBYTES  (924 TRACKS)
4  DSNT488I  VSAM DATA SETS CREATED FOR THE DB2 CATALOG AND DIRECTORY
             WILL REQUIRE AT LEAST 323972 KILOBYTES (6749 TRACKS)
5  DSNT488I  VSAM DATA SETS CREATED FOR DB2 CATALOG AND DIRECTORY INDEXES
             WILL REQUIRE AT LEAST 250764 KILOBYTES (5224 TRACKS)
6  DSNT488I  VSAM DATA SETS CREATED FOR ACTIVE LOG 1 AND BSDS 2
             WILL REQUIRE AT LEAST 103980 KILOBYTES (2166 TRACKS)
7  DSNT488I  VSAM DATA SETS CREATED FOR ACTIVE LOG 2 AND BSDS 1
             WILL REQUIRE AT LEAST 103980 KILOBYTES (2166 TRACKS)
-------------------------------------------------------------------------------
8  DSNT488I  DATA SETS CREATED FOR DB2 INSTALLATION AND VERIFICATION
             WILL REQUIRE AT LEAST 835056 KILOBYTES (17397 TRACKS)


PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 70. CLIST calculations panel 2: DSNTIPC1

CLIST messages fields for panel DSNTIPC1

The messages in these fields might be used depending on how many unique volume names were supplied on installation panel DSNTIPA2. These fields are protected and cannot be changed.

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

Completing the CLIST processing

After you receive CLIST messages (on installation panel DSNTIPC1) that indicate the calculated sizes, you can complete CLIST processing by pressing Enter. You then receive a series of messages that give details about the CLIST processing.

CLIST messages

The CLIST issues messages as it completes processing.

You first receive the following message:

```
DSNT478I BEGINNING EDITED DATA SET OUTPUT
```

The CLIST is checking the parameter values that you entered. If it detects a problem, you receive an error or warning message indicating the name of the parameter and the type of problem. If you receive an error message, the CLIST cannot edit the installation or migration jobs properly. If you receive a warning message, check the conditions. A warning message can sometimes be issued even when the conditions are normal or acceptable. If you specify several large numbers in the panels, the CLIST might send a message indicating an overflow in CLIST arithmetic.

At this point the CLIST displays the Main Panel again. You can proceed through the panels, rechecking or changing parameter values.

If the CLIST does not find any errors, you receive messages on DSNTIPC1 that indicate the amount of required disk storage and virtual storage. You might also receive some other information messages. For example, if you specified an existing data set for the TEMP CLIST LIBRARY field or the SAMPLE LIBRARY field on panel DSNTIPT, the following message is displayed:

```
WARNING(S) ISSUED
```

Related reference:

"CLIST calculations panel 2: DSNTIPC1" on page 495

Installation jobs that the CLIST tailors

The installation CLIST tailors each installation job according to the panel values that you specified.

For each edited job, you receive the following message:

```
DSNT489I CLIST EDITING dsname(member), explanation
```

Important: If an error occurs while the installation CLIST edits your jobs, you will receive this message:

```
IKJ52555I NOTHING SAVED
ENTER SAVE OR END-
```

Enter END to prevent modification of the original copies of your installation jobs.

After the CLIST finishes tailoring the jobs, it displays the Main Panel again. If you need to continue your tailoring at another time, conclude this session. Then, when you start a new session, use the value that you specified for OUTPUT MEMBER NAME during this session as the value for INPUT MEMBER NAME during the new session. Enter these values on the Main Panel.

If you receive a message from the editor, such as `TEXT NOT FOUND`, enter END NOSAVE to exit. That message can indicate an error. You can rerun the CLIST with the trace control parameter set to CONTROL(SYMLIST) to learn what caused the problem. In some cases, specifying CONTROL(LIST) as the trace control parameter may provide enough information for you to find the source of the problem.

The installation CLIST uses the values that you specify on the installation panels to tailor and load the installation or migration jobs. Each job is composed of one or more JCL procedures or job steps. The CLIST loads each job as a separate member of the newly created *prefix*.NEW.SDSNSAMP. Before you run any of these jobs, however, you might want to perform some editing that the CLIST does not do. If disk allocation is completely controlled by SMS for your installation, verify that the input to IDCAMS from the install jobs does not conflict with the requirements for SMS.

Which jobs you edit depends on the task that you are performing: installation, migration, or update. In data sharing environments, you edit different jobs depending on the data sharing function: group, member, or enable. The installation CLIST tailors a different set of jobs for each task.

If you are installing, the CLIST tailors these jobs:

DSNTEJ0	DSNTEJ2E	DSNTEJ6U	DSNTESD	DSNTIJMV

DSNTEJ1	DSNTEJ2F	DSNTEJ7	DSNTESE	DSNTIJRT
DSNTEJ1L	DSNTEJ2H	DSNTEJ71	DSNTIJCA	DSNTIJRV
DSNTEJ1P	DSNTEJ2P	DSNTEJ73	DSNTIJDE	DSNTIJRW
DSNTEJ1S	DSNTEJ3C	DSNTEJ75	DSNTIJEX	DSNTIJSG
DSNTEJ1T	DSNTEJ3M	DSNTEJ76	DSNTIJIC	DSNTIJTC
DSNTEJ1U	DSNTEJ3P	DSNTEJ77	DSNTIJID	DSNTIJTM
DSNTEJ2A	DSNTEJ4C	DSNTEJ78	DSNTIJIE	DSNTIJUZ
DSNTEJ2C	DSNTEJ4P	DSNTESA	DSNTIJIF	DSNTIJVC
DSNTEJ2D	DSNTEJ67	DSNTESC	DSNTIJIN	

If you specified a default WLM environment name in the WLM ENVIRONMENT field of installation panel DSNTIPX, the CLIST edits DSNTEJ2U, DSNTEJ6O, DSNTEJ6R, DSNTEJ6V, and DSNTEJ6W.

If you activated DDF and specified a value in the REMOTE LOCATION field of panel DSNTIPY, the CLIST also tailors job DSNTEJ6.

If you activated DDF and specified a default WLM environment name in the WLM ENVIRONMENT field of installation panel DSNTIPX, the CLIST edits DSNTEJ6Z, DSNTEJ63, DSNTEJ64, DSNTEJ65, and DSNTEJ66.

If you activated DDF, specified a value in the REMOTE LOCATION field of panel DSNTIPY, and specified a default WLM environment name in the WLM ENVIRONMENT field of installation panel DSNTIPX, the CLIST edits DSNTEJ6D, DSNTEJ6P, DSNTEJ6S, DSNTEJ6T, DSNTEJ61, and DSNTEJ62.

If CICS is selected, the CLIST edits DSNTEJ5A, DSNTEJ5C, and DSNTEJ5P.

If you are using data sharing, the CLIST edits DSNTIJGF and DSNTIJFT.

The installation CLIST tailors the DSNHC, DSNH, DSNU, and DSNEMC01 CLISTs for installation.

If you are migrating to conversion mode (from Version 8 or Version 9.1), the CLIST tailors the following jobs:

DSNTIJCX	DSNTIJIN	DSNTIJRV	DSNTIJTM
DSNTIJEX	DSNTIJMV	DSNTIJRW	DSNTIJUZ
DSNTIJFV	DSNTIJRI	DSNTIJSG	DSNTIJVC
DSNTIJIC	DSNTIJRT	DSNTIJTC	

The CLIST also tailors the DSNH, DSNHC, DSNU, and DSNEMC01 CLISTs for migration to conversion mode (from Version 8 or Version 9.1).

If you are updating, the CLIST tailors the following jobs: DSNTIJRT and DSNTIJUZ. DSNTIJRT is updated only if you select installation panel DSNTIPR1.

If you are migrating to new-function mode, the CLIST tailors the following jobs: DSNTIJEN, DSNTIJNF, DSNTIJES, DSNTIJCS, DSNTEJxx, DSNTIJNG, and DSNTIJNH.

All of these jobs are described in the following topics. Recovery information is provided, along with a description of each job. Unless otherwise stated in the job description, a return code of 0 or 4 from any of the jobs indicates successful

completion. Some of the jobs contain statements that could fail without causing the job to fail. For instance, delete commands for data sets, drop statements for SQL objects, and stop commands could fail when you first run a job because the data sets or objects do not exist. Unless otherwise stated, you can ignore these failures. The statements are needed to allow you to rerun the job (if necessary) without performing the deletes, drops, and stops manually; they are merely for cleanup or initialization processing.

When a job fails, follow the instructions that are provided in the recovery information for the job.

Related information:

DSNT489I (DB2 Messages)

Tailoring the installation jobs

You might want to tailor several items in the installation jobs, such as data set names or library names, to meet the requirements of your site.

About this task

The changes that are described below are general; that is, they apply to all the jobs processed by the CLIST.

Before you begin editing, you might want to print or back up the jobs. You can print the JCL for these jobs by using IEBPTPCH or any other print facility that is available at your site.

Procedure

To prepare the installation jobs for your site:

1. Tailor the jobs to suit the needs of your site. You should edit the jobs to conform to any unique requirements that you might have. Also, you might want to make any minor JCL changes for items that the ISPF panels did not handle.
2. If you specified volume serial numbers on installation panel DSNTIPA3, examine the volume serial numbers that are used in the various jobs. The volume serial number fields of installation panel DSNTIPA3 enable you to specify up to five volumes for the data sets that are defined during installation or migration. If you want to use more than five volumes, specify them before continuing with installation or migration tasks.

   If you specified volume serial numbers on installation panel DSNTIPA3, the DSNTINST CLIST spreads the data sets across these volumes. If you use explicit volume serials instead of SMS, adding more volumes to provide more separation of data sets can help improve system performance and recoverability. Many of the log data sets are large and easy to place on separate volumes. The CLIST produces a series of messages that estimate space for each type of data set that is created during the installation or migration process.
3. Edit the DSNH CLIST if needed. The DSNH CLIST allows you to precompile, compile, prelink-edit, link-edit, bind, and run an application by issuing a single command. You might need to edit the DSNH CLIST to change values for some of the entries. Verify that DSNH keyword parameters for all DB2-supported compilers that your applications use.

a. Check the default data set names for the licensed programs that you have installed. These defaults are in the parameter definitions at the beginning of each program. If the names and prefixes are not correct for your site, change them.

b. Check default library names. If the names and prefixes are not correct for your site, change them. Ensure that the data sets exist and are cataloged for BLIB, CLIB, LLIB, and PLIB. When the DSNH CLIST runs, it creates DBRMLIB and LOAD data sets if they do not already exist. The DBRMLIB data set is created only if the DBRMLIB(DEFAULT) is set. The following are the default library names:

   - BLIB(NONE)
   - DBRMLIB(DEFAULT) - The DBRM library must be allocated exclusively when the precompiler writes to it.

     Recommendation: Set up a temporary library, or one per user, rather than trying to share libraries.
   - CLIB(NONE)
   - LLIB(NONE)
   - LOAD(RUNLIB.LOAD) - This library is allocated exclusively when it is being written.

     Recommendation: Set up a temporary library, or one per user, rather than trying to share libraries.
   - PLIB(NONE)

c. Check default processor options. If you prefer other default options, change them. The default processor options follow:

CICSOPT(NONE)
COPTION(NONE)
LOPTION(NONE)
PASS(DEFAULT)

d. Check print and work space defaults. If the default allocation sizes are not acceptable for your site, change them. The print and work space defaults follow:

PSECSPAC(20)
PSPACE(20)
WORKUNIT(DEFAULT)
WSECSPAC(20)
WSPACE(20)

4. Examine the data set names for other products. Many data set names for other products appear in the jobs. These names are shown in the table on panel DSNTIPU. Change them if they are different at your site.

Related reference:

"Data parameters panel: DSNTIPA3" on page 281

"Data set names panel 2: DSNTIPU" on page 300

DSNH (TSO CLIST) (DB2 Commands)

Job DSNTIJUZ and the subsystem parameter load module, application defaults load module, and DSNHMCID

Job DSNTIJUZ generates the subsystem parameter module each time that you install, migrate, or update DB2. DSNTIJUZ also creates the application defaults load module and DSNHMCID.

Several macros expand to form the data-only subsystem parameter load module, which contains the DB2 execution-time parameters that you selected using the ISPF panels. The macros are:

- DSN6ARVP
- DSN6FAC
- DSN6LOGP
- DSN6SPRM
- DSN6SYSP
- DSN6GRP

Job DSNTIJUZ also generates the application defaults load module (referred to here as *dsnhdecp* with either the traditional name of DSNHDECP or a user-specified name. DB2 is shipped with a default *dsnhdecp* module called DSNHDECP for compatibility with older applications. You cannot start DB2 or precompile applications with the default *dsnhdecp* module. During DB2 start-up processing or for jobs that precompile a DB2 application, you must ensure that the *dsnhdecp* module that was created by job DSNTIJUZ resides in a library, usually *prefix*.SDSNEXIT, that is concatenated before the *prefix*.SDSNLOAD library where the DB2-supplied *dsnhdecp* resides.

Job DSNTIJUZ also creates the data-only load module DSNHMCID, which contains the EBCDIC CCSIDs for text conversion in offline messages. When you install DB2, job DSNTIJUZ creates DSNHMCID in *prefix*.SDSNEXIT and *prefix*.SDSNLOAD.

In general, DSNHMCID should exist in both *prefix*.SDSNEXIT and *prefix*.SDSNLOAD. If it cannot reside in *prefix*.SDSNLOAD, take one of the following actions:

- Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in the system link list.
- Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statements for all DB2 applications, address space start-up procedures, TSO log-on procedures, CICS tasks, and IMS tasks that use DB2.

DB2 allows online changes to many of the subsystem parameters. The SET SYSPARM command enables a function that allows reloading.

Related tasks:

"Migration step 11: Define DB2 initialization parameters: DSNTIJUZ" on page 590

Related reference:

-SET SYSPARM (DB2) (DB2 Commands)

"Directory of subsystem parameters and application default values" on page 242

The update process

You can modify some of the parameters that you specified during installation or migration of DB2. The update process allows you to tailor DB2 more precisely to your needs.

The update process does not generate a complete set of installation or migration jobs, as the installation and migration process does. It generates only one job, DSNTIJUZ. This job assembles and link-edits the DB2 data-only subsystem parameter module and the application defaults load module, *dsnhdecp*.

Updating parameters through the Update selection menu panel: DSNTIPB

You can update most of the subsystem parameters by using the update selection menu panel, DSNTIPB.

Procedure

To update most parameters:

1. Run the installation CLIST, and specify UPDATE on installation panel DSNTIPA1.
2. Choose the output SDSNSAMP data set on installation panel DSNTIPT.

   The CLIST then takes you to installation panel DSNTIPB.
3. From installation panel DSNTIPB, select the installation panel that you want to update.
4. When you finish making changes to that panel, press ENTER to return to the Update Selection Menu Panel. You can select another panel to update, or press ENTER again to complete the update process.
5. To cancel the update session, press END.
6. Run job DSNTIJUZ

Related reference:

"Main panel: DSNTIPA1" on page 265

"Data set names panel 1: DSNTIPT" on page 295

"Update selection menu panel: DSNTIPB"

Update selection menu panel: DSNTIPB

During the update process, the DSNTIPB panel allows you to access all the installation panels and view the values that you specified during installation or migration.

Parameters that you can update are highlighted. Panels containing fields that cannot be updated are marked with an asterisk.

```
DSNTIPB          UPDATE DB2 - SELECTION MENU
===>

 Select one of the following:
 1 CATALOG AND DIRECTORY                19 PERFORMANCE AND OPTIMIZATION
 2 DATA PARAMETERS PANEL                20 IRLM PANEL 1
 3 DEFINE GROUP OR MEMBER               21 IRLM PANEL 2
 4 SYSTEM RESOURCE DATA SET NAMES       22 DB2 UTILITIES PARAMETERS
 5 DATA SET NAMES PANEL 1               23 DB2 UTILITIES PARAMETERS PANEL 2
 6 DATA SET NAMES PANEL 2 *             24 PROTECTION
 7 DATA SET NAMES PANEL 3 *             25 PROTECTION PANEL 2
 8 SIZES                                26 MVS PARMLIB UPDATES
 9 SIZES PANEL 2                        27 ACTIVE LOG DATA SET PARAMETERS
10 WORK FILE DATABASE                   28 ACTIVE LOG CHECKPOINT PARAMETERS
11 THREAD MANAGEMENT                    29 ARCHIVE LOG DATA SET PARAMETERS
12 BUFFER POOL SIZES PANEL 1            30 DATABASES TO START AUTOMATICALLY
13 BUFFER POOL SIZES PANEL 2 *          31 DISTRIBUTED DATA FACILITY
14 TRACING AND CHECKPOINT PARAMETERS    32 DISTRIBUTED DATA FACILITY PANEL 2
15 OPERATOR FUNCTIONS                   33 ROUTINE PARAMETERS
16 DEFAULT STARTUP MODULES              34 PARAMETERS FOR DB2-SUPPLIED ROUTINES
17 APPLICATION PROGRAMMING DEFAULTS 1   35 DATA DEFINITION CONTROL SUPPORT
18 APPLICATION PROGRAMMING DEFAULTS 2   36 JOB EDITING
 * None of the fields on these panels can be updated.
PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 71. Individual update menu panel: DSNTIPB

Panels to update field:

From the DSNTIPB panel, you can select a family of parameters to display and update. Some parameters cannot be updated.

On the command line, enter a number to select the family of parameters that you want to update. These numbers correspond to the installation panels in the following table.

Table 80. Panel identifiers

Panel ID	Panel title	See
1. DSNTIPA2	DB2 Catalog and Directory	“DB2 catalog and directory panel: DSNTIPA2” on page 275
2. DSNTIPA3	Data Parameters	“Data parameters panel: DSNTIPA3” on page 281
3. DSNTIPK	Define Group or Member	“Define group or member panel: DSNTIPK” on page 286
4. DSNTIPH	System Resource Data Set Names	“System resource data set names panel: DSNTIPH” on page 290
5. DSNTIPT	Data Set Names Panel 1	“Data set names panel 1: DSNTIPT” on page 295
6. DSNTIPU	Data Set Names Panel 2	“Data set names panel 2: DSNTIPU” on page 300
7. DSNTIPW	Data Set Names Panel 3	“Data set names panel 3: DSNTIPW” on page 309
8. DSNTIPD	Sizes	“Sizes panel 1: DSNTIPD” on page 313
9. DSNTIP7	Sizes Panel 2	“Sizes panel 2: DSNTIP7” on page 318
10. DSNTIP9	Work File Database	“Work file database panel: DSNTIP9” on page 323

Table 80. Panel identifiers (continued)

Panel ID	Panel title	See
11. DSNTIPE	Thread Management	"Thread management panel: DSNTIPE" on page 333
12. DSNTIP1	Buffer Pool Sizes Panel 1	"Buffer pool sizes panel 1: DSNTIP1" on page 339
13. DSNTIP2	Buffer Pool Sizes Panel 2	"Buffer pool sizes panel 2: DSNTIP2" on page 342
14. DSNTIPN	Tracing and Checkpoint Parameters	"Tracing parameters panel: DSNTIPN" on page 343
15. DSNTIPO	Operator Functions	"Operator functions panel: DSNTIPO" on page 351
16. DSNTIPO3	Default Startup Modules	"Default startup modules panel: DSNTIPO3" on page 358
17. DSNTIPF	Application Programming Defaults Panel 1	"Application programming defaults panel 1: DSNTIPF" on page 362
18. DSNTIP4	Application Programming Defaults Panel 2	"Application programming defaults panel 2: DSNTIP4" on page 372
19. DSNTIP8	Performance and Optimization	"Performance and optimization panel: DSNTIP8" on page 379
20. DSNTIPI	IRLM Panel 1	"IRLM panel 1: DSNTIPI" on page 386
21. DSNTIPJ	IRLM Panel 2	"IRLM panel 2: DSNTIPJ" on page 391
22. DSNTIP6	DB2 Utilities Parameters Panel 1	"DB2 utilities parameters panel 1: DSNTIP6" on page 397
23. DSNTIP61	DB2 Utilities Parameters Panel 2	"DB2 utilities parameters panel 2: DSNTIP61" on page 404
24. DSNTIPP	Protection	"Protection panel: DSNTIPP" on page 410
25. DSNTIPP1	Protection Panel 2	"Protection panel 2: DSNTIPP1" on page 412
26. DSNTIPM	MVS PARMLIB Updates	"MVS PARMLIB updates panel: DSNTIPM" on page 422
27. DSNTIPL	Active Log Data Set Parameters	"Active log data set parameters: DSNTIPL" on page 426
28. DSNTIPL1	Active Log Checkpoint Parameters	"Checkpoint parameters: DSNTIPL1" on page 428
29. DSNTIPA	Archive Log Data Set Parameters	"Archive log data set parameters panel: DSNTIPA" on page 434
30. DSNTIPS	Databases to Start Automatically	"Databases and spaces to start automatically panel: DSNTIPS" on page 441
31. DSNTIPR	Distributed Data Facility	"Distributed data facility panel 1: DSNTIPR" on page 442
32. DSNTIP5	Distributed Data Facility Panel 2	"Distributed data facility panel 2: DSNTIP5" on page 448
33. DSNTIPX	Routine Parameters	"Routine parameters panel: DSNTIPX" on page 453

Table 80. Panel identifiers (continued)

Panel ID	Panel title	See
34. DSNTIPR1	Parameters for DB2-Supplied Routines	"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458
35. DSNTIPZ	Data Definition Control Support	"Data definition control support panel: DSNTIPZ" on page 485
36. DSNTIPY	Job Editing	"Job editing panel: DSNTIPY" on page 488

When the panel that you selected is displayed, enter the new parameters; press the Enter key to return to the Update Selection Menu Panel. Make another panel selection or press Enter again to process. Press End to leave the Update Selection Menu Panel and return to the Main Panel.

Updating other parameters

Some subsystem parameters cannot be updated through panel DSNTIPB.

Procedure

To modify some of the parameters that you cannot update through panel DSNTIPB:

- To update the CATALOG ALIAS and DEFINE CATALOG fields on DSNTIPA2, see Changing the high-level qualifier for DB2 data sets (DB2 Administration Guide). The CATALOG ALIAS parameter establishes an alias name for your ICF catalog. This name is also used as the high-level qualifier name for DB2 VSAM data sets. The DEFINE CATALOG parameter controls the creation of the ICF catalog.
- To update DB2 to use the distributed data facility (DDF), follow these steps:
  1. Go through the normal update process of running the CLIST to add DDF information to installation panel DSNTIPR.
  2. Run job DSNTIJUZ.
  3. Populate the CDB.
  4. Stop and start DB2.
  5. Bind or rebind these plans:

     GUPI

     ```
     BIND PLAN(DSNESPCS) PKLIST(*.DSNESPCS.DSNESM68)
          ISOLATION(CS) ACTION(REPLACE)

     BIND PLAN(DSNESPRR) PKLIST(*.DSNESPRR.DSNESM68)
          ISOLATION(RR) ACTION(REPLACE)

     BIND PLAN(DSNESPUR) PKLIST(*.DSNESPUR.DSNESM68)
          ISOLATION(UR) ACTION(REPLACE)
     ```

     GUPI

  6. Start DDF if you specified COMMAND instead of AUTO as the DDF STARTUP OPTION on installation panel DSNTIPR.
- To change the data set sizes for the DB2 catalog and directory:
  1. Copy the catalog and directory table spaces.

2. Stop the table spaces or their databases.
3. Delete the data sets and redefine them, using VSAM commands.
4. Use the RECOVER utility to recover the catalog and directory to the new data sets.
5. Start the table spaces or databases again.

- To change from single to dual logging for the active log:
  1. Define the second copy of the log with a VSAM IDCAMS DEFINE statement. Refer to job DSNTIJIN, which contains the DEFINE statement for the first copy of the log.
  2. Run the DSNJU003 (change log inventory) utility. This adds the second copy of the log to the BSDS.
  3. Update the NUMBER OF COPIES field on installation panel DSNTIPH from 1 to 2.
  4. Run job DSNTIJUZ to make the change effective.
- To move or expand the bootstrap data sets, use the IMPORT and EXPORT commands of access method services. The bootstrap data sets are accessed using JCL when DB2 starts.
- To access the log data sets, you can use stand-alone access macros or the IMPORT and EXPORT commands of access method service.
- To change the number of data sets for active logs, you can use the DSNJU003 utility.

Related tasks:

"Populating the communications database for use with TCP/IP" on page 925

Defining a new integrated catalog alias (DB2 Administration Guide)

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

Related information:

Recovering from active log failures (DB2 Administration Guide)

Chapter 5. Installing or migrating DB2

Before installation or migration, you must prepare your system, load the DB2 libraries, and use the installation CLIST to tailor the DB2 jobs to your environment. When you complete these tasks, you are ready to install or migrate DB2.

Installing DB2

Installing DB2 involves running the installation jobs, connecting the facilities that allow TSO, batch, IMS, and CICS to access DB2 resources, and preparing DB2 for use. After you complete the installation steps, your DB2 subsystem will be in new-function mode. In new-function mode, all DB2 Version 10 function is available for use.

Before you begin

Before you begin installing DB2:

- Complete the SMP/E steps and run the installation CLIST.
- Define an SMS environment for DB2. At a minimum, the data sets for the DB2 catalog and directory must be SMS-managed. For information on using sample job DSNTIJSS to define an SMS environment for data sets that are associated with the DB2 catalog and directory, see "Installation step 1: Define the SMS environment for DB2 catalog and directory data sets: DSNTIJSS."
- If you plan to have data sharing enabled (Enable option on panel DSNTIPP1), you should save all your jobs from the original installation or migration in a different data set than the enabling jobs. If you save them in *prefix*.NEW.SDSNSAMP, the jobs might be overwritten.
- See the DB2 Program Directory for keyword specifications used for Preventive Service Planning (PSP). Use Information/Access or the ServiceLink facility of IBMLink to check the most current information about DB2 and other products. Contact the IBM Support Center if you do not have access to IBMLink.

You must not use secondary authorization IDs to perform any of the following installation steps.

Related concepts:

"Strategy for creating a data sharing group" on page 669

Chapter 3, "Loading the DB2 libraries," on page 207

Related tasks:

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

Installation step 1: Define the SMS environment for DB2 catalog and directory data sets: DSNTIJSS

All data sets for the DB2 catalog and directory must reside on SMS-managed storage and belong to a SMS data class that is defined with the extended addressability attribute.

© Copyright IBM Corp. 1982, 2012

About this task

To define a data class with the extended addressability attribute, specify EXT in the DATA SET NAME TYPE field of the DATA SET CLASS DEFINE panel of ISMF and ensure that the automatic class selection (ACS) routine associates the DB2 catalog and directory data sets with this data class. Job DSNTIJSS shows how to create a stand-alone SMS environment for the DB2 catalog and directory data sets. DSNTIJSS is designed for use on systems that do not already have a SMS environment, but it can also be used as a reference for adapting an existing one. In particular, job step STEP061 shows an example of defining a data class with the extended addressability attribute, and job step STEP071 shows example ACS routines that associate the DB2 catalog and directory data sets with the data class and a storage class.

Procedure

To create the stand-alone SMS environment for the DB2 catalog and directory:

1. Customize job DSNTIJSS according to the directions in the job prolog.
2. Run job DSNTIJSS.
3. To activate the SMS environment, use this z/OS command, where *scds-name* is the name of the SMS source control data set that was specified by DSNTIJSS:

   ```
   SETSMS SCDS(scds-name)
   ```

   Attention: This command will deactivate any existing SMS environment that is defined from another SCDS.

Installation step 2: Define DB2 to z/OS: DSNTIJMV

Job DSNTIJMV completes some of the steps that are required to identify DB2 to z/OS. These steps include updating members of SYS1.PARMLIB and SYS1.PROCLIB and creating Language Environment options data sets for the DB2 core WLM environment for Java routines.

Before you begin

Before you begin to install DB2 Version 10, you must have the prerequisite level of z/OS installed.

About this task

Do not overwrite the z/OS-supplied entries for DB2 and IRLM in the program properties tables (PPT) in SYS1.PARMLIB(SCHED*xx*). The operating system supplies default PPT values for all DB2 modules that require them. You should not change these values.

Procedure

To define DB2 to z/OS:

1. Run job DSNTIJMV. If job DSNTIJMV runs successfully, it produces return codes of 0.
2. Review the content of the Language Environment options data set and adjust the following environment variables as needed. The data set is created by DSNTIJMV and must be allocated by the JAVAENV DD of the address space procedure for the DB2 core WLM environment for Java routines.

- _CEE_ENVFILE, if present, needs to be set to the name and path of the environment variable file that was created by job step DSNTIJR.
- JVMPROPS needs to be set to the name and path of the JVM startup options file that was created by job step DSNTIJS.
- DB2_BASE and JCC_HOME need to be set to the names of the HFS directories that you selected for the DB2 for z/OS and JCC product classes.

What to do next

z/OS requirements: Each DB2 and each IRLM that you define to z/OS in the IEFSSN*xx* PARMLIB member requires a z/OS system linkage index (LX). The default number of these indexes that z/OS reserves is 165. If you place all of your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 165 LXs; otherwise, your subsystems might not start. If you need more than 165 LXs, use the NSYSLX option on the z/OS IEASYS*xx* PARMLIB member to increase this number.

For later diagnosis of IRLM problems, also ensure that the following requirements are met:

- Specify the IRLM dump formatting modules DXRRLM50, DXRRLM55, and DXRRLS55 in a BLSCUSER member in a data set that is in the system's PARMLIB concatenation.
- Copy the DXRRL186, DXRRLFTB, DXRRLM50, DXRRLM55, and DXRRLS55 modules from the IRLM SDXRRESL data set to a data set in the system's link list. Or, add the SDXRRESL data set to the system's link list in a PROG*xx* member of a data set in the system's PARMLIB concatenation. If you use more than one level of IRLM, you will need to use a STEPLIB concatenation when using IPCS to process dumps that are taken for the levels whose modules are not in the link list.

Additional changes to SYS1.PARMLIB and SYS1.PROCLIB: Because different sites have different requirements for identifying DB2 to z/OS, job DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes to SYS1.PARMLIB and SYS1.PROCLIB might be incomplete. You might have additional procedures of your own to rename. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB, or by editing DSNTIJMV.

Recommendation: Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV step. For SYS1.PROCLIB, submit the procedure-update section of job DSNTIJMV. Before you make the updates, read the following information and examine job DSNTIJMV to study the updates that it makes. Then use an editor such as ISPF/PDF to make the updates to SYS1.PARMLIB.

Related concepts:

Runtime environment for Java routines (Application Programming for Java)

DSNTIJMV updates to SYS1.PARMLIB

Job DSNTIJMV updates the following SYS1.PARMLIB members: IEFSSN*xx*, IEAAPF*xx* or PROG*xx*, and LNKLST*xx*.

- **IEFSSN*xx***

  This member contains an entry for every z/OS subsystem. DB2 adds to this list of entries, making one entry for DB2 and two entries for the IRLM. The second IRLM entry, whose subsystem name is JRLM, is there to make it easier to add a

second IRLM to your system if the first is damaged. You must provide your own procedure to add JRLM. Unique names must be used for each entry.

z/OS provides subsystem entries for DB2 and IRLM in IEFSSN00. Examine these entries to determine whether they are appropriate for your needs. Make sure that a subsystem name appears only once in the subsystem name list.

Place the SMS line as the first line in an IEFSSN*xx* member. Place the primary system's record (JES2 or JES3) record next. Other products might affect the position of the entries during system initialization. The DB2 line should come after SMS, the JES subsystem, and other vendor products.

The DB2 entry has the following format:

```
SUBSYS SUBNAME(ssname) INITRTN(DSN3INI)
   INITPARM('DSN3EPX,cmd-prefix<,scope<,group-attach<<,subgroup-attach>>>')
```

where:

ssname
is the DB2 subsystem name.

DSN3INI
is the name of the DB2 load module z/OS invokes during master scheduler initialization. This module must be located in a link list data set (or in SYS1.LINKLIB).

DSN3EPX
is the name of the DB2 load module that responds to DB2 requests that are received from the z/OS subsystem interface. (DB2 can be active or inactive when the requests are received.) This module must be located in a link list data set (or in SYS1.LINKLIB).

cmd-prefix
is the one- to eight-character command prefix.

The first character of the command prefix must be one of the following characters: @ # . / ') * + - = ¢ < | & ! ; % _ ? : ". The remaining characters of the command prefix must be one of the above characters, A-Z, or 0-9. Do not use the JES2 backspace character or command prefix character. The default is the hyphen (-).

Do not assign a command prefix that is used by another subsystem or that can be interpreted as belonging to more than one subsystem or z/OS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix beginning from the first character. For example, you cannot assign '-' to one subsystem and '-DB2A' to another. Similarly, you also cannot assign '?DB2' to one subsystem and '?DB2A' to another. You can assign '-DB2A' and '-DB2B' to different DB2 subsystems.

scope
is the one-character scope for the command prefix. DB2 registers its command prefix with the operating system. When this is done, the scope of the command prefix is controlled by the value you choose:

S Started; and the prefix is registered with Sysplex scope at DB2 startup instead of during z/OS IPL. This is the default.

Recommendation: Choose S, which allows you to have a single IEFSSN*xx* PARMLIB member that all z/OS systems in the Sysplex can use. Choosing S also simplifies the task of moving a DB2 from one system to another; you can stop DB2 on one z/OS and start it up on another without having to re-IPL the system.

M z/OS system scope; the prefix is registered during z/OS IPL.

X Sysplex scope; the prefix is registered during z/OS IPL. As a result, this DB2 cannot be restarted on another z/OS without changing the definitions and re-IPLing both z/OSs.

group-attach
is the group attachment name, which is used for data sharing. You can specify this on installation panel DSNTIPK.

subgroup-attach
is the name of the subgroup attachment, which is used in data sharing environments to provide more control over how TSO, CAF, RRSAF, JDBC, ODBC, and DB2 utilities connections find active DB2 subsystems. You can specify this value by entering a subgroup name in the SUBGRP ATTACH field on installation panel DSNTIPK.

- **IEAAPF*xx* or PROG*xx***

  Job DSNTIJMV updates IEAAPF*xx* to include the DB2 program libraries (*prefix*.SDSNEXIT, *prefix*.SDSNLOAD, *prefix*.SDXRRESL, and *prefix*.SDSNLINK) as APF-authorized libraries.

  If a program library that contains DFSORT or DB2 Sort is not already APF-authorized, you can edit DSNTIJMV to authorize it. To do this, you can include the authorization either in this list or in LNKLST*xx*. All libraries that are concatenated with *prefix*.SDSNLOAD in STEPLIB and JOBLIB statements must be APF-authorized. Ensure that the volume serial number in this member is the volume on which the data set resides.

  If you are using the PROG*xx* member instead of the IEAAPF*xx* member, you need to update the member manually—job DSNTIJMV does not edit it.

- **LNKLST*xx***

  Job DSNTIJMV updates this member to include the DB2 load module library, *prefix*.SDSNLINK. If you moved the modules from *prefix*.SDSNLINK into another library, edit DSNTIJMV to include that library in the LNKLST*xx* member. If you have combined *prefix*.SDSNLINK and *prefix*.SDSNLOAD into one library, edit DSNTIJMV to include the combined library in the LNKLST*xx* member.

  Any data set that is added to the LNKLST*xx* member must be cataloged in the master catalog of the system. This is normally true of *prefix*.SDSNLINK; however, if an alias points to a user catalog when you run DSNALLOC, *prefix*.SDSNLINK is cataloged in a user catalog. In this case, you must either ensure that *prefix*.SDSNLINK is also cataloged in the master catalog or give *prefix*.SDSNLINK a high-level qualifier other than *prefix*, the high-level qualifier for this release. You must give a high-level qualifier other than *prefix* to all release-sensitive data sets that are placed in the LNKLST*xx* member.

  If you do not include the DFSORT library, the DB2 Sort library, or both in the LNKLST*xx* member, you must provide a JOBLIB or STEPLIB statement for all utility jobs that include one of those libraries. You can accomplish this by placing a STEPLIB statement in DSNUPROC, which appears later in this job. If you use customized modules and exits, *prefix*.SDSNEXIT must precede *prefix*.SDSNLOAD in JOBLIB and STEPLIB statements.

You must do additional editing for the SYS1.PARMLIB updates. If you are editing DSNTIJMV, rather than making the changes directly, you have two options. One option is to include additional entries for the SYS1.PARMLIB members (IEAAPF*xx* and LNKLST*xx*) at the end of the existing entry list. The other option is to place the entries earlier in the list.

If you include them at the end of the existing SYS1.PARMLIB entries, ensure that commas (the continuation character) delimit each entry except the last.

Another SYS1.PARMLIB change to consider at this time is the extended common storage area (ECSA) size, which is specified in the CSA parameter of the IEASYS00 parameter. Ensure that you have specified an adequate size for this subsystem.

The IOP parameter is another SYS1.PARMLIB change to consider at this time. DB2 can schedule I/O priority. To enable this, you must:

- Use the IOP parameter to set the I/O priority for the address space of a performance group. The IOP parameter is in the IEAIPS*xx* member of SYS1.PARMLIB.
- Enable z/OS I/O priority scheduling by specifying IOQ=PRTY in the IEAIPSxx member of SYS1.PARMLIB.

You must issue an IPL command for z/OS for the PARMLIB updates to take effect. To avoid issuing an IPL command for z/OS during DB2 installation, you can make these updates and issue the IPL in advance of your DB2 installation or migration session.

Related concepts:

How DB2 assigns I/O priorities (DB2 Performance)

Related reference:

"COMMAND PREFIX field" on page 423

Related information:

Using the command prefix facility (z/OS MVS Planning: Operations)

DSNTIJMV updates to SYS1.PROCLIB

Job DSNTIJMV updates SYS1.PROCLIB to include the DB2 procedures.

DSNTIJMV includes the following procedures, where *ssnm* is the subsystem name:

- System services address space startup procedure (must be named *ssnm*MSTR)
- Database services address space startup procedure (must be named *ssnm*DBM1)
- Distributed data facility address space startup procedure (must be named *ssnm*DIST)
- Administrative task schedule address space startup procedure
- IRLM address space startup procedure (IRLMPROC or user-defined address space name)
- Precompiler procedures
- Utilities procedure (DSNUPROC)
- z/OS Workload Manager (WLM) procedure
- Address space startup procedures for the DB2 core WLM environments (*ssnm*WLM1, *ssnm*WLMC, *ssnm*WLMD, *ssnm*WLMG, *ssnm*WLMJ, *ssnm*WLMM, *ssnm*WLMP, *ssnm*WLMR, *ssnm*WLMU, *ssnm*WLMW, and *ssnm*WLMX)

Examine the SYS1.PROCLIB updates carefully. You might want to use a procedure library other than SYS1.PROCLIB for the procedures. Four of the procedures are used for start-up tasks; the other procedures are used to prepare application programs for execution and to invoke DB2 utilities. The program preparation procedures are required for the sample applications and can be helpful in generating other JCL procedures.

Change any data set names that differ at your site. If you specified a library name suffix in the LIBRARY NAME SUFFIX field of panel DSNTIPA1, that suffix is

appended to data sets &USER..DBRMLIB.DATA, &USER..RUNLIB.LOAD, and &USER..SRCLIB.DATA. If you specified a data set name suffix in the DATA SET NAME SUFFIX field of DSNTIPA1, that suffix is appended to libraries that are created during installation. To override these data set names, you must edit the updates to SYS1.PROCLIB.

The language preparation procedures in job DSNTIJMV use the DISP=OLD parameter to enforce data integrity. However, when the installation CLIST runs, the DISP=OLD parameter for the DBRM library data set is modified to DISP=SHR. This might cause data integrity problems when you run multiple precompiler jobs. To avoid these data integrity problems, if you are not using DFSMSdfp partitioned data set extended (PDSE), you must change the language preparation procedures (DSNHICOB, DSNHICB2, DSNHFOR, DSNHC, DSNHCPP, DSNHCPP2, DSNHPLI, DSNHASM, and DSNHSQL) to specify the DISP=OLD parameter instead of the DISP=SHR parameter.

If compiler STEPLIB statements are needed, add them.

The STEPLIB concatenation of the *ssnm*DBM1 address space procedure includes a commented-out DD for the IBM Language Environment data set SCEERUN. If your system does not include SCEERUN in the system link list, you must uncomment this DD. Also, if your system link list does not include the Language Environment data set SCEERUN2, you must add it to the STEPLIB concatenation of your *ssnm*DBM1 address space procedure.

Examine the size of the private area on the DB2 start procedures. If necessary, modify the procedures to satisfy the requirements for environmental descriptor manager (EDM) pool size, buffers, number of data sets open, and amount of available private address space.

Related tasks:

"Calculating working storage requirements" on page 60

Language Environment options data sets for the DB2 core Java WLM environment

Job DSNTIJMV creates the Language Environment options data sets for the DB2 core Java WLM environment.

Job step DSNTIJJ of DSNTIJMV creates the Language Environment options data set for DSNWLM_JAVA, which is the WLM environment for DB2-supplied Java stored procedures and user-defined functions. This data set is allocated by the JAVAENV DD of the *ssnm*WLMJ address space procedure that is created in job step DSNTIPM of DSNTIJMV.

In job step DSNTIJJ, review and optionally update the *JAVA_HOME* path, which is the path where the JDK is located.

```
Example: /usr/lpp/java150/J5.0
```

Installation step 3: Define the ICF catalog and alias: DSNTIJCA (optional)

The ICF catalog is the VSAM object in which DB2 records the data sets that you create during installation. DB2 uses the catalog alias as the prefix for your DB2 VSAM data sets.

About this task

Creating the catalog is optional. If you specified YES for the DEFINE CATALOG option on installation panel DSNTIPA2, you must create the catalog. Do not run this job if you want to use an existing ICF catalog and alias (that is, you specified NO for the DEFINE CATALOG parameter on installation panel DSNTIPA2). However, ensure that the ICF catalog that you are going to use is created and that you defined an ICF catalog alias.

Procedure

To create the ICF catalog and its alias:

1. Before running job DSNTIJCA, examine the DEFINE UCAT statement carefully to ensure that the parameters are appropriate for your needs.
2. Run job DSNTIJCA.

   If DSNTIJCA runs successfully, it produces return codes of 0.
3. If DSNTIJCA fails or abends, delete the ICF catalog (if it was created) and run the job again.

Related tasks:

"Installation step 1: Define the SMS environment for DB2 catalog and directory data sets: DSNTIJSS" on page 507

Related reference:

"DEFINE CATALOG field" on page 279

Installation step 4: Define system data sets: DSNTIJIN

Use job DSNTIJIN to define the VSAM and non-VSAM data sets for DB2.

Before you begin

Before you define the system data sets, ensure that you have defined an SMS environment for data sets that are associated with the DB2 catalog and directory. Starting in DB2Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

Procedure

To define system data sets:

1. Check the DEFINE CLUSTER statements in job DSNTIJIN to ensure that they allocate adequate disk space for your system.

   Recommendation: For recovery purposes, place system data sets such as the DB2 recovery log and the VSAM catalog on different disk volumes. Because these data sets are used frequently, do not migrate them by using DFSMShsm.
2. Run job DSNTIJIN.

   If DSNTIJIN runs successfully, it produces return codes of 0 for all DEFINE statements and steps. Check any VSAM messages carefully.

3. If job DSNTIJIN fails or abends, complete the following steps:
   a. Remove the z/OS catalog delete statements from job DSNTIJDE.
   b. Run job DSNTIJDE. This job deletes the previously created data sets for the DB2 directory and DB2 catalog.
   c. Run DSNTIJIN again.

Related concepts:

Allocating and extending data sets (DB2 Performance)

Related tasks:

"Installation step 1: Define the SMS environment for DB2 catalog and directory data sets: DSNTIJSS" on page 507

Job DSNTIJIN

During installation, job DSNTIJIN defines VSAM and non-VSAM data sets for DB2.

DSNTIJIN performs the following tasks:

- Defines three non-VSAM data sets for the DB2 sample objects:
  - *prefix*.DBRMLIB.DATA
  - *prefix*.SRCLIB.DATA
  - *prefix*.RUNLIB.LOAD
- Defines the VSAM clusters for the bootstrap data sets.

  Each bootstrap data set (BSDS) consists of a VSAM key-sequenced data set. You defined the BSDS names during the ISPF tailoring session
- Defines the VSAM clusters for the active log data sets.

  You specified up to 93 primary active log data sets during the ISPF tailoring session (NUMBER OF LOGS field on installation panel DSNTIPL). You might also have requested dual logging to generate two copies of each active log data set. Consequently, job DSNTIJIN can define up to 186 active log data sets. After installation, you can run the DSNJU003 utility to add additional active log data sets to the BSDS.
- Defines the VSAM clusters for the task list data set of the administrative task scheduler.
- Defines the DB2 directory database.

  Job DSNTIJIN creates and catalogs the DB2 directory database (DSNDB01). The DB2 directory database contains information that is required to start DB2 and is also used by DB2 during its normal operation. It contains table spaces and index spaces that the installation job allocates.
- Defines the DB2 catalog database.

  Job DSNTIJIN creates and catalogs the DB2 catalog database (DSNDB06). The DB2 catalog contains information about every object that DB2 maintains.
- Invokes the LISTCAT command of access method service so that you can verify that the VSAM definitions were successful.
- Creates control intervals based on the value you specified in VARY DS CONTROL INTERVAL on panel DSNTIP7. If you specified YES, DB2 creates 4-KB, 8-KB, 16-KB, and 32-KB control intervals as appropriate in the AMS DEFINE CLUSTER commands for the DB2 catalog and directory data sets. If you specified NO, these data sets are created using a fixed control interval of 4-KB.

Job DSNTIJDE

Job DSNTIJDE deletes the previously created data sets for the DB2 directory and DB2 catalog. If a job fails or abends, you might need to run DSNTIJDE job before restarting the DB2 installation process.

Job DSNTIJDE is not part of the normal installation process; use this job only for rerunning part of the process. Do not run this job during migration or fallback.

In most cases, you must remove or comment out the delete statement in this job for the ICF catalog (if the statement is present). The ICF catalog probably does not need to be deleted and redefined.

Deletes might fail for data sets that do not exist. This does not necessarily indicate that the job failed. If you receive other messages, check them carefully.

Job DSNTIJDE does not work properly if job DSNTIJSG or job DSNTIJTM has been executed. Job DSNTIJDE does not delete the resource limit specification table, the data sets that are used by the distributed data facility, or the data sets that are created for work file table spaces.

If job DSNTIJDE fails or abends, correct the error conditions and rerun the job. If you want to delete the ICF catalog, first list its contents and delete the data sets that are cataloged there. This could include sample data sets, user-defined data sets, or subsystem data sets that were not deleted properly. You can use a FORCE command to delete the user catalog. If you delete the catalog using FORCE before deleting all the data sets, you can use the RECATALOG option of DEFINE CLUSTER and delete the data sets.

Installation step 5: Define DB2 initialization parameters: DSNTIJUZ

Job DSNTIJUZ generates the DB2 data-only load module DSNHMCID, the application defaults load module (*dsnhdecp*), and the subsystem parameter module. The application defaults load module is the name that was specified for the APPL DEFAULTS field on installation panel DSNTIPO3. The subsystem parameter module is the name that was specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

Procedure

To define DB2 initialization parameters:

1. If you added a STEPLIB statement to the DB2 start procedures, modify the SYSLMOD steps to point to the library in the STEPLIB statement instead of to the library in *prefix*.SDSNEXIT.
2. If the DB2 distribution library prefix is different from the target library prefix, edit DSNTIJUZ to correct the data set name for *prefix*.ADSNLOAD.
3. If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBAA10, edit DSNTIJUZ to include the SMP/E temporary data set (SMPTLIB) in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. SMPTLIB is *hlq*.HDBAA10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

   ```
   SET   BOUNDARY (GLOBAL).
   LIST  DDDEF ( SMPTLIB ).
   ```

4. Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD.
5. Run job DSNTIJUZ.

If you want DSNTIJUZ to establish SMP/E GENASM entries in the DB2 system parameter (DSN6*xxxx*) and application defaults module macros, you need to uncomment job step DSNTIMQ. You might receive message GIM65001 when you run steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when you run step DSNTIMQ. You can ignore these messages.

6. If job DSNTIJUZ fails or abends, correct the problem and rerun the job.

Related tasks:

"Installation step 6: Initialize system data sets: DSNTIJID, DSNTIJIE, and DSNTIJIF"

Improving outer join processing (DB2 Performance)

Using a subsystem parameter to optimize queries with IN predicates (DB2 Performance)

Favoring index access (DB2 Performance)

Related reference:

"Default startup modules panel: DSNTIPO3" on page 358

"Subsystem parameters that are not on installation panels" on page 253

Installation step 6: Initialize system data sets: DSNTIJID, DSNTIJIE, and DSNTIJIF

After you define your system data sets and DB2 initialization parameters, you must initialize your system data sets.

Procedure

To initialize system data sets:

Run jobs DSNTIJID, DSNTIJIE, and DSNTIJIF, in that order.

- If the jobs run successfully, they produce return codes of 0. Otherwise, check for and resolve error conditions that might be reported by VSAM messages.
- If any of the jobs fail or abend, remove the ICF catalog delete statements (job step DSNTIVC) from job DSNTIJDE, run job DSNTIJDE, and then rerun DSNTIJIN, DSNTIJUZ, DSNTIJID, DSNTIJIE, and DSNTIJIF.

Results

Job DSNTIJID records the active log data set names to the BSDS, formats the active log data sets, and initializes the DB2 directory table spaces and indexes.

Job DSNTIJIE initializes the DB2 catalog table spaces and indexes (through the SYSGRTNS table space).

Job DSNTIJIF initializes the remaining DB2 catalog table spaces and indexes.

Related concepts:

"Installation step 7: Define user authorization exit routines: DSNTIJEX (optional)"

Installation step 7: Define user authorization exit routines: DSNTIJEX (optional)

Job DSNTIJEX builds sample authorization exit routines from the source code in *prefix*.SDSNSAMP. These sample authorization exit routines are DSN3@SGN and DSN3@ATH, and the user version of the access control authorization exit routine, DSNX@XAC.

Job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. Then DSNTIJEX places the exit routines in the *prefix*.SDSNEXIT library. The DB2 CLIST tailors the JCL in DSNTIJEX to match your site's environment.

The sample authorization exit routines are not the same as the default authorization exit routines that are supplied by DB2. By implementing the sample authorization exit routines, you can provide group names as secondary authorization IDs. By modifying the sample authorization exit routines, you can tailor authorization processing for your subsystem.

DSNXSXAC is a copy of the default access control authorization exit routine that users can modify. This exit routine allows you to bypass some or most of DB2 authorization checking to specify your own authorization checking. If you do not modify it, this step is not needed and you should delete it.

DSNACICS is a stored procedure that invokes user exit routine DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. If you do not need to modify the caller's parameter values, you can use the default DSNACICX exit routine. However, if you need to modify the caller's parameter values, you need to perform the following tasks:

1. Write a user exit routine in assembler, COBOL, C, or PL/I
2. Assemble or compile the source code
3. Link-edit the object code into the DB2 exit routine library

Installation job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX. You can use this step as a model for your program preparation job.

You have the following options regarding exit routines:

- To use the sample authorization exit routines, run job DSNTIJEX.
- To use the default authorization exit routines, skip job DSNTIJEX.
- To use the modified sample authorization exit routines, modify DSNTIJEX to reference the correct library before you run it.

If you will use the RACF/DB2 external security module (DSNXRXAC) as your DB2 access control authorization exit routine, modify DSNTIJEX to refer to DSNXRXAC instead of DSNXSXAC.

If job DSNTIJEX runs successfully, it produces return codes of 4.

If job DSNTIJEX fails or abends, correct the problem and rerun the job.

Related concepts:

DB2 data access control (Managing Security)

The default DB2 exit routine (RACF Access Control Module Guide)

Related reference:

"Default startup modules panel: DSNTIPO3" on page 358

Related information:

Exit routines (DB2 Administration Guide)

Installation step 8: Record DB2 data to SMF (optional)

When you install DB2, you can specify whether DB2 statistical, accounting, and audit trace data is to be collected.

Procedure

To specify the data that is to be collected:

1. Choose from the following options:

Option	Description
To have DB2 collect statistical information	Accept the default (YES class 1) for the SMF STATISTICS option on installation panel DSNTIPN. To collect statistical information for deadlock or timeout, specify class 3. To collect information about DDF error conditions, specify class 4.
To have DB2 collect accounting information	Accept the default (1) or specify * (all classes) for the SMF ACCOUNTING option on installation panel DSNTIPN.
To have DB2 collect auditing information	Specify * (all classes) for the AUDIT TRACE option on installation panel DSNTIPN.

   In all cases, DB2 invokes a trace, passing the data it collects to the System Management Facility (SMF) of z/OS.

   DB2 also passes performance data to SMF whenever an accounting, statistics, or audit trace is successfully started or stopped. DB2 can also record other performance data. After you complete the installation process, you can use commands to have DB2 record performance data for over 230 different subsystem events.

2. If, during installation, you requested that DB2 pass accounting and statistics data to SMF, update the SMFPRM*xx* member of SYS1.PARMLIB as follows:
   - Specify the ACTIVE parameter.
   - Specify STC,NODETAIL for the SUBSYS parameter.
   - Specify the proper TYPE subparameter of the SYS and SUBSYS parameters.

What to do next

During DB2 execution, you can use the SMF SET or SS command to alter the SMF parameters. For example, you can record the statistics trace class 1 IFCIDs 0001, 0002, and 0202 (SMF record type 100); accounting trace class 1 IFCIDs 0003 and 0239 (SMF record type 101); and all other DB2 trace records (SMF record type 102) to SMF. To record this information, issue the following command:

```
SYS(TYPE(100:102))
```

For DB2 to pass data to SMF, you must allocate an adequate supply of SMF buffers. The default buffer settings are probably insufficient.

You can specify SMF buffering on the VSAM BUFSP parameter of the Access Method Services DEFINE CLUSTER statement. Do not use the default settings if DB2 data is sent to SMF. Specify CISZ(4096) and BUFSP(81920) on the DEFINE CLUSTER statement for each SMF VSAM data set. These values for CISZ and BUFSP are the minimum requirement for DB2. You might need higher values for CISZ and BUFSP, depending on the requirements of all your z/OS subsystems.

You can also code an IEFU84 SMF exit routine to process the records that are produced.

Related tasks:

Controlling traces (DB2 Administration Guide)

Related reference:

-START TRACE (DB2) (DB2 Commands)

z/OS MVS Initialization and Tuning Guide

Related information:

z/OS MVS Initialization and Tuning Reference

Installation step 9: Establish subsystem security (optional)

DB2 can control access to data within a DB2 subsystem. DB2 also works with outside security systems, such as RACF, that control access to the DB2 subsystem.

Before you begin

The following IDs should have been specified on installation panel DSNTIPP1:

- Two system administrator (installation SYSADM) authorization IDs
- Two system operator (installation SYSOPR) authorization IDs
- Two security administrator (installation SECADM) authorization IDs or roles
- One authorization ID (installation IBMUSER) if RACF is not available for batch access and USER= is not specified in the JOB statement

Procedure

To establish subsystem security:

- Ensure that the specified installation SYSADM IDs are defined in your TSO and RACF systems before attempting to access DB2. You can also define installation SYSOPR IDs, SECADM IDs, and the installation IBMUSER ID.
- To enable primary and secondary user IDs to issue DB2 commands from the z/OS console or TSO SDSF, define RACF classes to authorize DB2 commands. Use the following RACF commands:

```
SETR CLASSACT(DSNADM)
RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
SETR RACLIST(DSNADM) REFRESH
PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
SETR RACLIST(DSNADM) REFRESH
```

You can grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF, but only after DB2 can process SQL.

- To separate security tasks from system administration tasks, use the SEPARATE SECURITY field on the DSNTIPP1 panel. Before setting the SEPARATE SECURITY field to YES, set at least one SECADM system parameter to an authorization ID, or create the necessary trusted contexts and roles. If you specify YES for SEPARATE SECURITY, system administrator authority cannot be used to perform security tasks, and the SECADM authority is required to manage trusted contexts and roles. If both SECADM system parameters are set to roles and those roles have not been created, no one will have the authority to manage trusted contexts and roles.
- Run job DSNTIJRA to complete the following tasks:
  - Define the administrative task scheduler started task module to RACF program control
  - Define the administrative task scheduler as a trusted context in RACF

  If you are using a DB2 data sharing environment, customize and run DSNTIJRA for each member of the group.

Related concepts:

Getting started with DB2 security (Managing Security)

Installation step 10: Connect DB2 to TSO

Although you can eventually connect DB2 to IMS, CICS, or both, connecting initially only to TSO is recommended for a first-time installation.

About this task

For a first-time installation, you can run the sample applications that do not require CICS or IMS, allowing your database and system administrators to gain familiarity with the administrative facilities of this version of DB2.

If you have previously installed DB2 and are performing that task again, your database and system administrators are probably already familiar with DB2. In this case, you can connect IMS, CICS, or both, at the same time that you connect TSO. You can then run the sample applications that require CICS and IMS at the same time that you run the sample applications for TSO and batch.

Procedure

To attach DB2 to TSO:

1. Make DB2 load modules available to TSO and batch users.
2. Make DB2 CLISTs available to TSO and batch users.
3. Make PL/I options available (if applicable).
4. Make panels, messages, and load modules available to ISPF and TSO.
5. Connect the DB2I panels to the ISPF Main Panel.

Make DB2 load modules available to TSO and batch users

As part of installation, you must make DB2 load modules available to TSO and batch users. If *prefix*.SDSNEXIT and *prefix*.SDSNLOAD are not included in your LNKLST*xx*, you can skip this step.

Procedure

To make DB2 load modules available to TSO and batch users:

If you have not included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, you must add STEPLIB statements to your logon procedures and to the JCL for jobs to ensure that you access the DB2 Version 10 load modules. If *prefix*.SDSNEXIT is not in your LINK*xx*, add it to your STEPLIB and JOBLIB concatenations before *prefix*.SDSNLOAD.

Related tasks:

"Choosing link list options" on page 208

Make DB2 CLISTs available to TSO and batch users (DSNTIJVC)

To connect DB2 to TSO, you must make DB2 CLISTs available to TSO and batch users by running the DSNTIJVC job.

About this task

From *prefix*.SDSNCLST, the CLIST reads and edits these four CLISTs: DSNEMC01, DSNH, DSNU, and DSNHC. It then places those CLISTs in *prefix*.NEW.SDSNTEMP. You might want to modify the default values. The DSNEMC01 CLIST provides installation default values for the DB2I Default panels (option D on panel DSNEPRI). The first time that a TSO user executes DB2I on a specific ISPF application such as DSNE (NEWAPPL), the DSNEMC01 CLIST sets the defaults based on the values that are specified on installation panel DSNTIPF. DSNEMC01 stores the values in the ISPF profile member DSNEPROF.

Job DSNTIJVC merges the tailored CLISTs from *prefix*.NEW.SDSNTEMP with unchanged CLISTs and REXX execs from *prefix*.SDSNCLST. Then job DSNTIJVC places all CLISTs and REXX execs in *prefix*.NEW.SDSNCLST. It also converts the record format of the DB2 CLISTs from fixed-block to variable-block format with a record length of 84 and a block size of 3120. A CLIST that has been converted from fixed-block to variable-block format cannot be used as input to the DSNTINST CLIST; use the unedited version of the SDSNCLST data set, as created by SMP/E.

Procedure

To make the CLISTs available to TSO and batch users:

1. If you use fixed-block format for your CLIST libraries, modify job DSNTIJVC as follows:
   - Change the SYSIN DD statement to DUMMY.
   - Change the allocation of *prefix*.NEW.SDSNCLST to match the data control block (DCB) attributes of your other CLIST libraries.
2. Run job DSNTIJVC.
3. Either concatenate *prefix*.NEW.SDSNCLST with your existing CLIST libraries or copy *prefix*.NEW.SDSNCLST into an existing CLIST library.

What to do next

If you need to rerun job DSNTIJVC, first delete data set *prefix*.NEW.SDSNCLST, which is created by this job.

When corrective service is applied to a CLIST, SMP/E changes only the *prefix*.SDSNCLST data set. You need to redo any record format changes and

reapply any special tailoring that is required. You also need to move the CLIST to *prefix*.NEW.SDSNCLST. Corrective service (program temporary fixes) for these CLISTs is sent with ++HOLD statements, calling to your attention the possibility of additional work.

Related tasks:

"Completing the CLIST processing" on page 496

Ensure that PL/I options are available

If you are using PL/I, ensure that the options that your DB2 programmers use are included in the compiler.

Procedure

To ensure that PL/I options are included in the compiler:

1. Ensure that the following options that are used by the program preparation function are available:

   ```
   FLAG    OBJECT    SOURCE    TERMINAL    XREF
   ```

   Restrictions that are imposed by your site on PL/I compiler options affect how you can use DB2 program preparation.
2. If the macro pass is used, ensure that the following options are also available:

   ```
   MACRO    MDECK    SYNTAX
   ```

Make DB2I panels, messages, and load modules available to ISPF and TSO

During installation, to connect the DB2 Interactive facility (DB2I) to TSO, you must make panels, messages, and load modules available to ISPF and TSO.

About this task

DB2I uses the ISPF PROFILE and SHARED variable pools for most panel variable fields. As a result, you can easily re-enter a panel when panel variables have previously been specified. For the DB2 subcommands that permit lists of plan names, package names, DBRMs, and ENABLE and DISABLE statements, DB2I provides ISPF to contain all the user-specified variables for these subcommand keywords.

DB2I creates and maintains a set of ISPF tables in a user-defined TSO data set that is allocated to a data set with a ddname of DSNETBLS. The DB2I-generated tables in this library are DSNCONNS, DSNDBRMS, and DSNPLPKN. The following table shows the library table member names and their contents.

Table 81. The DB2 ISPF table library

DSNCONNS	ENABLE or DISABLE connection type and connection name variables that are referenced by plan or package name
DSNDBRMS	Subcommand DBRM member and LIBRARY name variables that are referenced by plan name
DSNPLPKN	Package list variables that are referenced by package name
DSNPATHS	Schema name variables that are referenced by plan or package name and that are to be included in the PATH keyword

DB2I uses ISPF table services to maintain individual ISPF tables within the DSNETBLS data set. For performance reasons, ISPF keeps this table library in an

open state once an individual table has been updated. Attempts to close this data set by using the TSO FREE command results in error message IKJ56861I.

Procedure

To make panels, messages, and load modules available to ISPF and TSO:

1. If you added the DB2I libraries as part of the step to prepare for running the DB2 installation CLIST, you can skip this step. Otherwise, concatenate the DB2 ISPF libraries with the ISPPLIB, ISPTLIB, ISPSLIB, and ISPMLIB DD statements in your logon procedures. You must also concatenate the DB2 ISPF libraries with these DD statements in any of your CLISTs where the libraries might be allocated. These libraries are *prefix*.SDSNSPFP, *prefix*.SDSNSPFM, *prefix*.SDSNSPFS, and *prefix*.SDSNPFPE or *prefix*.SDSNPFP. If you use English DB2I panels, concatenate SDSNPFPE to ISPPLIB. If you use Kanji DB2I panels, concatenate SDSNPFPK instead of SDSNPFPE.

   ```
   PROFILE WTP MSGID
   ALLOCATE DDNAME(SYSEXEC) DSN('prefix.SDSNCLST') SHR REUSE
   ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
       'ISP.SISPMENU') SHR REUSE
   ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' 'prefix.SDSNPFPE' +
       'ISP.SISPPENU') SHR REUSE
   ALLOCATE DDNAME(ISPSLIB) DSN('prefix.SDSNSPFS' +
       'ISP.SISPSLIB' 'ISP.SISPSENU') SHR REUSE
   ```

2. Create the DSNETBLS data set.

   When allocating this data set, you should assign the following DCB attributes, where *n* is any integer:

   ```
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(n*LRECL)
   ```

   The following example shows how to set up an ALLOCATE statement to create the data set:

   ```
   ALLOC DA(DSNSPFT) NEW SP(1 1) TR DIR(10) +
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(800)
   F(DSNETBLS) REUSE
   ```

   The following example shows how to allocate an existing data set to the data set with the DSNETBLS ddname:

   ```
   ALLOC DA(DSNSPFT)      F(DSNETBLS) REUSE
   ```

3. If you want to run the ISPF/CAF sample application that is provided with DB2, ensure that the data set *prefix*.RUNLIB.LOAD is included in the logon procedures or in the ISPLLIB concatenation. If you chose IBMCOB for your LANGUAGE DEFAULT on panel DSNTIPF, you should also include the data set *prefix*.CEE.SCEERUN.

Related concepts:

"Dynamic SQL and the ISPF/CAF application" on page 766

Related reference:

z/OS ISPF Messages and Codes

Connect DB2I panels to the ISPF main panel

You can connect the DB2I panels to the standard ISPF panels that are already installed on your system.

About this task

Recommendation: Use the following panels for establishing the connection. See your TSO administrator for other possibilities.

- Use panel ISP@MSTR, ISR@PRIM, or ISRFPA for the connection to DB2 Interactive services.
- Use panel ISR00003 for the tutorial menu update.

Two example panels are provided. Their names are DSN8ISPM (the DB2 version of an ISPF primary options panel) and DSN8ISPT (the DB2 version of a tutorial table of contents).

If the DB2 panel library is concatenated before the standard ISPF library, the connection is made.

Procedure

To connect DB2I panels to the ISPF main panel:

1. Using the TSO RENAME command, give DSN8ISPM an alias of ISR@PRIM, and give DSN8ISPT an alias of ISR00003. For example:

   ```
   RENAME 'prefix.SDSNSPFP(DSN8ISPM)' (ISR@PRIM) ALIAS
   RENAME 'prefix.SDSNSPFP(DSN8ISPT)' (ISR00003) ALIAS
   ```

2. Modify the panels as shown in the following examples. If your site has made changes to either of these panels, change your existing panels rather than using the examples.

Examples

```
)ATTR
/*********************************************************************/
/* COPYRIGHT = 5740-XYR (C) COPYRIGHT IBM CORP 1982, 1985, 1990, 2003 */
/* REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083             */
/* STATUS = VERSION 8, LEVEL 0                                       */
/*********************************************************************/
)BODY
%-----------------------  ISPF/PDF PRIMARY OPTION MENU  -----------------------
%OPTION  ===>_ZCMD                                          +USERID   - &ZUSER .
%   0 +ISPF PARMS  - Specify terminal and user parameters   +TIME     - &ZTIME .
%   1 +BROWSE      - Display source data or output listings +TERMINAL - &ZTERM .
%   2 +EDIT        - Create or change source data           +PF KEYS  - &ZKEYS .
%   3 +UTILITIES   - Perform utility functions
%   4 +FOREGROUND  - Invoke language processors in foreground
%   5 +BATCH       - Submit job for language processing
%   6 +COMMAND     - Enter TSO command or CLIST
%   7 +DIALOG TEST - Perform dialog testing
%   8 +DB2I        - Perform DATABASE 2 Interactive functions
%   C +CHANGES     - Display summary of changes for this release
%   T +TUTORIAL    - Display information about ISPF/PDF
%   X +EXIT        - Terminate ISPF using log and list defaults
%
+Enter%END+command to terminate ISPF.
%
 )INIT
   .HELP = ISR00003
   &ZPRIM = YES        /* ALWAYS A PRIMARY OPTION MENU      */
   &ZHTOP = ISR00003   /* TUTORIAL TABLE OF CONTENTS        */
   &ZHINDEX = ISR91000 /* TUTORIAL INDEX - 1ST PAGE         */
 )PROC
   &ZSEL = TRANS( TRUNC (&ZCMD'.')
                0,'PANEL(ISPOPTA)'
                1,'PGM(ISRBRO)'
                2,'PGM(ISREDIT)'
                3,'PANEL(ISRUTIL)'
                4,'PANEL(ISRFPA)'
                5,'PGM(ISRJB1) PARM(ISRJPA) NOCHECK'
                6,'PGM(ISRPTC)'
                7,'PGM(ISRYXDR) NOCHECK'
                8,'CMD(DSNECPRI) NEWAPPL(DSNE)'
                C,'PGM(ISPTUTOR) PARM(ISR00005)'
                T,'PGM(ISPTUTOR) PARM(ISR00000)'
              ' ','  '
                X,'EXIT'
                *,'?' )
   &ZTRAIL = .TRAIL
 )END
```

Figure 72. ISPF primary option panel (DSN8ISPM), edited to include DB2I

The figure above shows panel DSN8ISPM. Notice the added lines in boldface type. Adding these lines allows you to invoke the DB2 Interactive (DB2I) functions. The added lines include one displayed line:

```
%   8 +DB2I        - Perform DATABASE 2 Interactive functions
```

You can also choose to add one of these undisplayed lines:

```
8,'CMD(DSNECPRI) NEWAPPL(DSNE)'
8,'CMD(DSNECPRI SSID(xxxx)) NEWAPPL(DSNE)'
```

The displayed line lets the user choose DB2I. Both of the undisplayed lines invoke the DB2I main panel (DSNEPRI). If you use the first undisplayed line, you accept

the default for the subsystem identifier (SSID) parameter. If you use the second undisplayed lines, you can specify a different SSID parameter.

DSNECPRI is a CLIST and can be invoked directly from another user CLIST. It is an alternative way to invoke DSNEPRI without updating the primary ISPF panel.

By specifying NEWAPPL(DSNE), you define DSNE as the ISPF application that DB2I uses. ISPF uses the name DSNE to create the ISPF profile pool member name (DSNEPROF) in the TSO_*userid*.ISPPROF data set, which contains all ISPF panel variables defined during DB2I execution. Any customized DSNEPROF members can be migrated from Version 10.

Recommendation: Examine any new or changed default panel values to ensure that your custom values are still valid, specifically the option values for the subcommands BIND PLAN, REBIND PLAN, BIND PACKAGE, and REBIND PACKAGE.

Using a NEWAPPL name other than DSNE: You can define any valid ISPF application name.

```
)ATTR
/*********************************************************************/
/* COPYRIGHT = 5740-XYR (C) COPYRIGHT IBM CORP 1982, 1985, 1990, 2003 */
/* REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083             */
/* STATUS = VERSION 8, LEVEL 0                                       */
/*********************************************************************/
)BODY
 %TUTORIAL -------------------- TABLE OF CONTENTS ------------------ TUTORIAL
 %OPTION  ===>_ZCMD

                  ---------------------------------------------
                  | ISPF PROGRAM DEVELOPMENT FACILITY TUTORIAL |
                  |             TABLE OF CONTENTS              |
                  ---------------------------------------------
    The following topics are presented in sequence, or can be selected by
    entering a one-character selection code in the option field on line 2:
       %G+ GENERAL     - General information about ISPF
       %0+ ISPF PARMS  - Specify terminal and user parameters
       %1+ BROWSE      - Display source data or output listings
       %2+ EDIT        - Create or change source data
       %3+ UTILITIES   - Perform utility functions
       %4+ FOREGROUND  - Invoke language processors in foreground
       %5+ BATCH       - Submit job for language processing
       %6+ COMMAND     - Enter TSO command or CLIST
       %7+ DIALOG TEST - Perform dialog testing
       %8+ DB2         - Information about DB2
       %X+ EXIT        - Terminate ISPF using log and list defaults
    The following topics are presented only if explicitly selected:
       %A+ APPENDIX A  - Dynamic allocation interface routine (DAIR) errors
       %B+ APPENDIX B  - ISPF listing formats
       %I+ INDEX       - Alphabetic index of tutorial topics
 )PROC
    &ZSEL = TRANS(&ZCMD
                  G,ISR01000
                  0,ISP05000
                  1,ISR10000
                  2,ISR20000
                  3,ISR30000
                  4,ISR40000
                  5,ISR50000
                  6,ISR60010
                  7,ISR70000
                  8,DSN4V2DB
                  X,ISP90100
                  A,*ISP93030
                  B,*ISR95000
                  I,*ISR91000
                  )
 )END
```

Figure 73. ISPF program development facility tutorial panel (DSN8ISPT), edited to include DB2 tutorial

The figure above shows panel DSN8ISPT. Notice the two added lines in boldface type. Adding these lines allows you to invoke the DB2 tutorial panels. One of the two added lines is displayed:

```
%8+ DB2         - Information about DB2
```

The other added line is undisplayed:

```
8,DSN4V2DB
```

The displayed line presents the user with a choice for the DB2 tutorial. The undisplayed line actually invokes the DB2 tutorial menu (DSN4V2DB).

Related reference:

z/OS ISPF Dialog Developer's Guide and Reference

ISPF User's Guide Vol I

Installation step 11: Connect DB2 to IMS (optional)

Connecting DB2 to IMS is an optional part of installation. Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

1. Make DB2 load modules available to IMS.
2. Define DB2 to IMS.
3. Define new programs and transactions to IMS.
4. Prepare IMS applications for DB2.
5. Depending on your site, you might also need to:
   - Define DB2 plans for IMS applications.
   - Generate a user language interface.

Related tasks:

"Connecting the IMS attachment facility" on page 869

Installation step 12: Connect CICS to DB2 (optional)

To connect DB2 to CICS, you must regenerate several CICS tables with additional entries.

About this task

Coordinate this connection with your CICS support group.

Related tasks:

Defining the CICS DB2 connection

Installation step 13: Define DB2 to z/OS

After you run job DSNTIJMV, you must define DB2 in SYS1.PARMLIB.

About this task

Job DSNTIJMV makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member
- Creates new load module libraries in the LNKLST*xx* member

The load module library SDSNLINK contains the early code. SDSNLINK contains modules that must be placed in the link list look-aside address space (LLA) because they are loaded at subsystem initialization during the IPL.

Procedure

To define DB2 to z/OS:

IPL z/OS.
During the z/OS IPL, message DSN3100I appears on the z/OS console, stating that DB2 is ready for the START command.
Alternatively, to define DB2 to z/OS until the next IPL, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands.

Related tasks:

"Installation step 2: Define DB2 to z/OS: DSNTIJMV" on page 508

Related reference:

SETPROG Command

SETSSI Command

Installation step 14: Start the DB2 subsystem

During installation of DB2, start the DB2 subsystem after you IPL z/OS.

Procedure

To start DB2:

1. Start the IRLM, if you have not requested that DB2 automatically start the IRLM. You must start IRLM before you start DB2.

   Use the following command:

   ```
   START irlmproc
   ```

   where *irlmproc* is the name that you specified for the PROC NAME option on IRLM Panel 1 (DSNTIPI).

   If you specified YES for the AUTO START option on IRLM Panel 1 (DSNTIPI), DB2 starts the IRLM automatically.
2. Start DB2 from the z/OS console. Use the following command:

   ```
   -DSN1 START DB2
   ```

   where (-DSN1) is the subsystem command prefix that you defined for DB2.

   DB2 uses the subsystem parameter module that is specified in the start-up JCL procedure in SYS1.PROCLIB:

   ```
   //IEFPROC EXEC PGM=DSNYASCP,PARM='ZPARM(DSNZPxxx)', ...
   ```

   where *DSNZPxxx* is the value that you specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

   If you need to change the name of the *DSNZPxxx* module, you can edit SYS1.PROCLIB. Alternatively, you can override the *DSNZPxxx* name by using the PARM option as follows:

   ```
   -DSN1 START DB2,PARM(DSNZPxxx)
   ```

   If you have not run job DSNTIJRT, and the administrative task scheduler starts, the administrative task scheduler issues error message DSNA679I to report SQLCODE -805 until you run job DSNTIJRT. The administrative task scheduler starts if you specified an address space procedure name in the ADMIN SCHEDULER field on installation panel DSNTIPX.

   If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, possibly *ssnm*DIST, possibly *ssnm*ADMT, and *irlmproc*, where *ssnm* is the DB2 subsystem name, *ssnm*ADMT is the DB2 administrative task scheduler procedure name, and *irlmproc* is the IRLM procedure name.

If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

```
DSNR002I   RESTART COMPLETED
DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
```

When you start DB2 Version 10 for the first time, DB2 issues message DSNT501I with reason code 00C900A6. This message is expected. When you run job DSNTIJTC, the cause of this message is corrected.

You must run job DSNTIJTC to complete the tailoring of the DB2 catalog. In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

After you start DB2, identify unusual conditions for databases with the command:

```
-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
```

If DB2 does not start successfully, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Ensure that the DSNTIJID, DSNTIJIE, DSNTIJIF, DSNTIJIN, and DSNTIJUZ jobs ran correctly. Also, verify that the subsystem parameter member that you specified (or allowed by default) when you started DB2 is the one that is built by job DSNTIJUZ. Check the JCL for the DB2 startup procedure.

If you use distributed data facility, you must define VTAM before DDF can start. However, you do not need to have TCP/IP configured to start DDF. In addition, transactions such as those from DDF or CICS may fail because work files have not yet been defined.

3. Optional: Start TSO. After DB2 starts, it is available to TSO. If you want to use the TSO SUBMIT command to do housekeeping and installation verification, you must start TSO (if it is not already started).

Installation step 15: Tailor the DB2 catalog: DSNTIJTC

Job DSNTIJTC invokes the CATMAINT utility to tailor your Version 10 catalog, which you must do to fully install DB2. DSNTIJTC contains one step that performs site-specific tailoring of the catalog.

Before you begin

To execute DSNTIJTC, you must have installation SYSADM authority. Also, SMS must be active on the system where this DB2 subsystem is being installed. SMS must be configured to allocate all data sets for the DB2 catalog and directory in extended format and to use extended addressability.

About this task

Important: In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

Procedure

To tailor the DB2 catalog:

Run job DSNTIJTC. You must ensure that the installation job runs on the same z/OS system on which the appropriate DB2 subsystem is running.
A status message, DSNU777I, is issued at several points to indicate progress. Diagnostic error messages are issued when CATMAINT processing fails. If a problem is found during the SQL processing phase of installation, message

DSNU778I is issued. If non-supported functions are encountered, message DSNU776I is issued. All of these messages are written to the SYSPRINT data set. Because this is the first use of DB2, errors from earlier steps might be detected here.

Results

If you receive abend reason code 00C200EF, the catalog and directory data sets are not in a compliant SMS environment.

If you receive an abend reason code from the data manager (X'00C9'*xxxx*) or buffer manager (X'00C2'*xxxx*), carefully recheck jobs DSNTIJIN, DSNTIJID, DSNTIJIE, and DSNTIJIF.

Related reference:

"Job editing panel: DSNTIPY" on page 488

CATMAINT (DB2 Utilities)

Related information:

DSNU776I (DB2 Messages)

DSNU777I (DB2 Messages)

DSNU778I (DB2 Messages)

Installation step 16: Create default storage group, define temporary work files, and bind DB2 REXX Language Support: DSNTIJTM

You must create the default storage group that is for user-defined DB2 tables that are not specifically assigned to a storage group. You must also define the database that is for temporary work files, and bind DB2 REXX Language Support.

Procedure

To create the default storage group, define temporary work files, and bind DB2 REXX Language Support:

Run job DSNTIJTM.
If job DSNTIJTM runs successfully, it produces the return codes that are shown in the following table:

Table 82. DSNTIJTM return codes

Step	PROCSTEP	Return code
DSNTIAD		
	PC	0000
	ASM	0000
	LKED	0000
DSNTIAB	(none)	0000
DSNTIAS	(none)	0000
DSNTICR	(none)	0000
DSNTIRX	(none)	0000
DSNTIST	(none)	0000

Results

DSNTIJTM assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically. It also defines the initial buffer pool sizes that are specified on installation panels DSNTIP1 and DSNTIP2. When you use the ENABLE or MEMBER functions, the steps to prepare and bind the DSNTIAD program are unnecessary and are removed during editing.

DSNTIJTM also provides some cleanup to ease reruns.

The SYSDEFLT storage group is the default storage group, and is used for your database, table space, and table definitions that are not related to a specific storage group. By default, it is also used for work file table spaces that are created during DB2 installation and migration. When DSNTIJTM creates the SYSDEFLT storage group, it performs the following grants:

- Grants use of the default buffer pool and storage group to PUBLIC.
- Grants use of the SYSDEFLT table space. This table space does not really exist, but this GRANT statement is necessary to give users the ability to implicitly create table spaces in the default database.
- Grants authority to create tables and table spaces in the default database to PUBLIC.

What to do next

For non-data-sharing installations, the work file database name is DSNDB07. For data sharing installations, the work file database name is the name that was specified in the WORK FILE DB field on installation panel DSNTIPK. After creating the work file database, the DSNTIJTM job creates the number of 4 KB and 32 KB work file table spaces that was specified in the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9.

You can increase the number of additional temporary work file table spaces by increasing the values for the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9, particularly if you expect a large amount of sorting at your site. Additional temporary work file table spaces can improve DB2 performance by reducing device contention among applications. These additional work files also can be used for sorting indexes on large tables during index creation.

You can choose to have job DSNTIJTM create these additional table spaces, or you can create them after you run DSNTIJTM. To create additional work file table spaces after you run job DSNTIJTM, use the DSNTWFG exec in job step DSNTIST. See the comment block for job step DSNTIST for information about the parameters for DSNTWFG.

If you do not run the installation CLIST, customize DSNTIJTM or the DSNTWFG exec to create at least one 4 KB and one 32 KB work file table space.

Related concepts:

"Work file database storage requirements" on page 35

Installation step 17: Define and bind DB2 objects: DSNTIJSG

Use job DSNTIJSG to bind DB2-supplied packages and plans, initialize the DB2 communications database, and create user-managed databases for the DB2 resource limit facility (RLF) and Data Definition Control support.

Before you begin

Important: In a data sharing environment, you must ensure that the resource limit facility (RLF) is inactive on all members in the data sharing group before running DSNTIJSG. To do this, issue the STOP RLIMIT command for each member.

Procedure

To use job DSNTIJSG:

1. If you use a product that uses a semicolon as a delimiter, replace the semicolons at the end of each SQL statement in job DSNTIJSG before you run the job.

   If you use a product that uses a semicolon as a delimiter, the CLIST adds SQL statements to job DSNTIJSG. Products that use a semicolon as a delimiting character cause semicolons to be removed from the installation CLIST before it is executed.
2. Run job DSNTIJSG.

   If the DSNTIJSG job runs successfully, it produces return codes of 0. It might also produce a return code of 4 because a step within this job attempts to delete a row from a table that might not exist at the time that this job runs. Expect the following messages from the BIND statement for each object that DB2 provides:

   ```
   DSNE932I  WARNING, ONLY IBM-SUPPLIED PLAN NAMES SHOULD BEGIN WITH DSN

   DSNE932I  WARNING, ONLY IBM-SUPPLIED PACKAGE-IDS SHOULD BEGIN WITH DSN

   DSNE932I  WARNING, ONLY IBM-SUPPLIED COLLECTION-IDS SHOULD BEGIN WITH DSN
   ```
3. If the DSNTIJSG job fails or abends:
   a. Be sure that the user that is specified on the JOB statement is authorized. Use the same name that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1.
   b. Correct any other problems with the DSNTIJSG job, and run it again. If you do not have enough resources to run the job, review the values that you specified for the DB2 installation parameters. (See job DSNTIJUZ.) Use the standard update procedure to make any necessary modifications. Then stop DB2, run the DSNTIJUZ job, start DB2, and run the DSNTIJSG job again.

Related concepts:

"The update process" on page 501

Data definition control support database (Introduction to DB2 for z/OS)

Resource limit facility database (Introduction to DB2 for z/OS)

Related tasks:

"Populating the communications database for use with TCP/IP" on page 925

"Special packages and plans for SPUFI" on page 568

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Creating the databases and routines (IBM OmniFind Text Search Server Installation, Administration, and Reference)

Related reference:

"Distributed data facility panel 2: DSNTIP5" on page 448

Job DSNTIJSG

During installation, job DSNTIJSG binds DB2-supplied packages and plans.

When you run job DSNTIJSG during installation, it binds DB2 plans, including the plans for SPUFI and DCLGEN. DSNTIJSG also binds a package for a message routine that is used by SPUFI and DCLGEN, and binds that package into the plans for SPUFI and DCLGEN. If you use SPUFI to access remote sites, you might need to bind different packages and plans for SPUFI. You also might need to bind different packages and plans for SPUFI if some users need to run SPUFI under different terminal CCSIDs.

Job DSNTIJSG also binds the package and plan for the DSNTXAZP program, which updates a specified CLIST defaults input member (DSNTID*xx*) with the current subsystem parameter and buffer pool settings that are defined in the subsystem.

The DSNTIJSG job also completes the following tasks for user-maintained database activity:

- Creates the resource limit facility database.
- Within the resource limit facility database, creates a new resource limit facility table, DSNRLMT*nn*, and a new resource limit facility index, DSNMRL*nn*. By default, the step that creates this new table and index is commented out in the JCL. To have DB2 create DSNRLMT*nn* and DSNMRL*nn*, uncomment the step.
- Creates the data definition control support database.

The DSNTIJSG job inserts a blank row into the communication database (CDB) table SYSIBM.LUNAME. The CDB holds tables that contain information about your connection with remote DB2 subsystems, and a blank row allows all SNA clients to access DDF. TCP/IP remote clients cannot be controlled by using the CDB. If you run DSNTIJSG again, SQLCODE -803 is issued after an INSERT request because the blank row already exists.

The TCP/IP ALREADY VERIFIED field of panel DSNTIP5 defines the minimum security requirements for all TCP/IP clients because inbound security requirements cannot be established on individual clients.

Installation step 18: Populate the user-maintained databases (optional)

Job DSNTIJSG creates user-maintained databases that you might need to populate.

Procedure

Populate the user-maintained databases, including the resource limit specification table, and the data definition control support tables. Although DB2 automatically creates the user-maintained databases, you must complete this step if you plan to use data definition control or the resource limit facility (RLF).

Installation step 19: Configure DB2 for running stored procedures and user-defined functions

Before you can use user-defined functions or the following types of stored procedures, you need to perform some set-up tasks:

- External stored procedures
- External SQL procedures
- Native SQL procedures that satisfy at least one of the following conditions:
  - The native SQL procedure calls at least one external stored procedure, external SQL procedure, or user-defined function.

- The native SQL procedure is defined with ALLOW DEBUG MODE or DISALLOW DEBUG MODE.

- DB2-supplied stored procedures

All routines, except native SQL procedures, must run in an environment that is established by the z/OS Workload Manager (WLM). A set of DB2-supplied routines are installed by job DSNTIJRT and validated by job DSNTIJRV as part of the installation process. You can also define your own routines or use routines that are provided by a non-IBM vendor. Before you can use any stored procedures and user-defined functions, however, you need to set up WLM environments for these routines. If you plan to use certain DB2-supplied stored procedures or user-defined functions, you need to perform some additional steps.

Related concepts:

Stored procedures (DB2 Application programming and SQL)

User-defined functions (DB2 SQL)

Tailoring DB2 subsystem parameters for stored procedures during installation (optional)

Several subsystem parameters should be tailored to your DB2 environment before you use stored procedures. These parameters were set when you installed or migrated DB2. If you are satisfied with those values, you do not need to complete this step.

About this task

The subsystem parameters for stored procedures are:

- MAX_NUM_CUR
- MAX_ST_PROC
- STORMXAB
- STORTIME

Procedure

To modify the values of these subsystem parameters, complete one of the following tasks:

- Run the installation CLIST again in INSTALL or MIGRATE mode. On the installation panels, leave most of the existing values and change the following fields on their respective panels:
  1. On panel DSNTIPA1, specify the input member that contains field values for your current installation.
  2. On panel DSNTIPT, specify different names for the TEMP CLIST LIBRARY and SAMPLE LIBRARY fields to avoid overwriting your original libraries.
  3. On panel DSNTIPX, add or change the values of the MAX ABEND COUNT, TIMEOUT VALUE, MAX OPEN CURSORS, and MAX STORED PROCS fields as appropriate.
  4. On panel DSNTIPY, specify a name for the REMOTE LOCATION field. DB2 uses this name for the stored procedure sample applications.
- Edit job DSNTIJUZ to add or change values for the subsystem parameters for routines.

  This method has the following disadvantages:

– DB2 does not use these new values for any future migrations because your DSNTID*xx* member and DSNTIJUZ parameters are not saved for future input.
– DB2 does not generate the sample jobs for stored procedure and user-defined functions because you are not running the DSNTINST CLIST.

Related reference:

"Main panel: DSNTIPA1" on page 265
"Data set names panel 1: DSNTIPT" on page 295
"Routine parameters panel: DSNTIPX" on page 453
"Job editing panel: DSNTIPY" on page 488
"Directory of subsystem parameters and application default values" on page 242

Setting up Language Environment for stored procedures during installation

Language Environment (LE) is a component of the z/OS operating system that establishes a common runtime environment for all applications, regardless of programming language. DB2 uses Language Environment to provide a runtime environment for external stored procedures and external SQL procedures.

About this task

Language Environment provides a set of essential runtime services that are available through a set of interfaces that are consistent across programming languages. These services include routines for runtime message handling, condition handling, and storage management.

Language Environment provides the following advantages for DB2 stored procedures:

- Language Environment hides the differences between programming languages.
- Language Environment provides the ability to make a stored procedure resident in the stored procedure address spaces.
- Language Environment supports a large number of runtime options, including those options that are needed to use tools to debug your stored procedures.

Procedure

To set up Language Environment for stored procedures:

1. Customize Language Environment by completing the instructions in z/OS Language Environment Customization.
2. Specify the Language Environment runtime library in the JCL startup procedure for each WLM-managed address space for stored procedures. This library must be the only available runtime library. Do not reference any other language runtime libraries within the system link list or within the JOBLIB or STEPLIB statements. If other language runtime libraries are defined in the system link list, use JOBLIB or STEPLIB statement overrides to exclude them.

   Multiple languages can share the same Language Environment runtime library. Therefore, you do not need to specify the language-specific libraries in the JCL startup procedure of each stored procedure address space.

   Restriction: Do not use compilers (or associated runtime libraries) that are no longer supported by IBM.

What to do next

After you set up Language Environment for stored procedures, you can override the Language Environment runtime options for individual stored procedures. You can do this by using the RUN OPTIONS clause in the CREATE PROCEDURE statement or ALTER PROCEDURE statement. The default values for Language Environment runtime options are defined by z/OS. You might have overridden some of these default values when you customized Language Environment.

Related concepts:

External stored procedures (DB2 Application programming and SQL)

SQL procedures (DB2 Application programming and SQL)

Language Environment setup (DB2 9 for z/OS Stored Procedures: Through the CALL and beyond)

Related reference:

CREATE PROCEDURE (external) (DB2 SQL)

CREATE PROCEDURE (SQL - external) (DB2 SQL)

Customizing Language Environment run-time options (z/OS Language Environment Customization)

Implementing RRS for stored procedures during installation

To use some DB2 stored procedures, you must implement Resource Recovery Services (RRS). RRS is a sync point manager program that coordinates two-phase commit processing. DB2 requires that RRS be active because WLM-managed stored procedure address spaces use the RRS attachment facility (RRSAF).

About this task

RRS records events for protected resources in log streams that are managed by a z/OS component called the system logger. The following table identifies the log streams that RRS uses, where *gname* can be your Sysplex name or any name in a non-Sysplex environment:

Table 83. Log streams that RRS uses

Log stream type	Log stream name	Content
Main unit-of-recovery log state stream	ATR.*gname*.MAIN.UR	The state of active units of recover (URs). RRS periodically moves this information into the RRS delayed UR state log when UR completion is delayed.
Delayed unit-of-recovery log state stream	ATR.*gname*.DELAYED.UR	The state of active URs, when UR completion is delayed.
Resource manager data log stream	ATR.*gname*.RM.DATA	Information about the resource managers that use RRS services.

Table 83. Log streams that RRS uses (continued)

Log stream type	Log stream name	Content
Restart log stream	ATR.*gname*.RESTART	Information, which is needed during restart, about incomplete URs. This information enables a functioning RRS instance to assume remaining incomplete work from an RRS instance that failed.
Archive log stream	ATR.*gname*.ARCHIVE	Information about completed URs. This log stream is optional, but recommended.

All of these log streams are required except for the archive log stream. RRS does not start if it is not able to connect to the required log streams.

In a Sysplex environment, these log streams are shared by the systems of the Sysplex.

Procedure

To implement RRS for stored procedures:

1. Define the RRS log streams that are to be written, either on disk or in the coupling facility by completing the following actions:
   a. If you are using the coupling facility, add definitions for the RRS structures in the CFRM policy.

      When you are use coupling facility log streams, each log stream needs to be mapped to a coupling facility structure.

      Example: The following example shows JCL that updates the CFRM policy to add the RRS structures:

```
//DEFCFRM1 JOB MSGCLASS=X,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN DD *
    DATA TYPE(CFRM) REPORT(YES)
    DEFINE POLICY NAME(CFRM18) REPLACE(YES)
    CF NAME(CF01)
        TYPE(009672)
        MFG(IBM)
        PLANT(02)
        SEQUENCE(000000040104)
        PARTITION(1)
        CPCID(00)
        DUMPSPACE(2048)
        CF NAME(CF02)
        TYPE(009672)
        MFG(IBM)
        PLANT(02)
................
................
................
STRUCTURE NAME(RRS_ARCHIVE_1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF1,CF2)
```

```
                REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_RMDATA_1)
                INITSIZE(8000)
                SIZE(16000)
                PREFLIST(CF1,CF2)
                REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_MAINUR_1)
                INITSIZE(8000)
                SIZE(16000)
                PREFLIST(CF1,CF2)
                REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_DELAYEDUR_1)
                INITSIZE(8000)
                SIZE(16000)
                PREFLIST(CF1,CF2)
                REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_RESTART_1)
                INITSIZE(8000)
                SIZE(16000)
                PREFLIST(CF1,CF2)
                REBUILDPERCENT(5)
```

b. Define the log streams in the system logger policy by using the IXCMIAPU utility. This utility program is provided in the SYS1.MIGLIB system library.

   If you are using the coupling facility, also define the structures in the system logger policy. You can map each log stream to a single structure or you can map log streams of like data types to the same structure.

   Example of defining coupling facility log streams: The following example JCL shows definitions for coupling facility structures and RRS log streams. In this case, each RRS log stream is mapped to a structure.

```
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DATA TYPE(LOGR) REPORT(YES)

  DEFINE STRUCTURE NAME(RRS_ARCHIVE_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(262)

  DEFINE STRUCTURE NAME(RRS_RMDATA_1) LOGSNUM(1)
         MAXBUFSIZE(1024) AVGBUFSIZE(252)

  DEFINE STRUCTURE NAME(RRS_MAINUR_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

  DEFINE STRUCTURE NAME(RRS_DELAYEDUR_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

  DEFINE STRUCTURE NAME(RRS_RESTART_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.ARCHIVE) STRUCTNAME(RRS_ARCHIVE_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.RM.DATA) STRUCTNAME(RRS_RMDATA_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)
```

```
DEFINE LOGSTREAM
NAME(ATR.SANDBOX.MAIN.UR) STRUCTNAME(RRS_MAINUR_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.DELAYED.UR) STRUCTNAME(RRS_DELAYEDUR_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.RESTART) STRUCTNAME(RRS_RESTART_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)
/*
```

Example of defining disk log streams: The following example JCL shows definitions for RRS log streams on disk.

```
/DEFSTREXECPGM=IXCMIAPU
//SYSPRINT DDSYSOUT=A
//SYSINDD *
   DATA TYPE(LOGR)

   DEFINE LOGSTREAM NAME(ATR.PLEX1.MAIN.UR)
      LOWOFFLOAD(60)
      HIGHOFFLOAD(80)
      DASDONLY(YES)
      HLQ(TEST)
      LS_SIZE(1024)
      LS_DATACLAS(VSAMLS)
      STG_SIZE(1024)

   DEFINE LOGSTREAM NAME(ATR.PLEX1.DELAYED.UR)
      LOWOFFLOAD(60)
      HIGHOFFLOAD(80)
      DASDONLY(YES)
      HLQ(TEST)
      LS_SIZE(960)
      LS_DATACLAS(VSAMLS)
      STG_SIZE(960)

  DEFINE LOGSTREAM NAME(ATR.PLEX1.ARCHIVE)
     LOWOFFLOAD(0)
     HIGHOFFLOAD(80)
     DASDONLY(YES)
     HLQ(TEST)
     LS_SIZE(960)
     LS_DATACLAS(VSAMLS)
     AUTODELETE(YES)
     RETPD(2)
     STG_SIZE(2000)

  DEFINE LOGSTREAM NAME(ATR.PLEX1.RM.DATA)
     LOWOFFLOAD(60)
     HIGHOFFLOAD(80)
     DASDONLY(YES)
     HLQ(TEST)
     LS_SIZE(192)
     LS_DATACLAS(VSAMLS)
     STG_SIZE(192)
```

```
DEFINE LOGSTREAM NAME(ATR.PLEX1.RESTART)
   LOWOFFLOAD(60)
   HIGHOFFLOAD(80)
   DASDONLY(YES)
   HLQ(TEST)
   LS_SIZE(960)
   LS_DATACLAS(VSAMLS)
   STG_SIZE(960)
```

c. If you are using coupling facility log streams, compile and link-edit the changed CFRM policy.

d. If you are using coupling facility log streams, activate the updated CFRM policy in your Sysplex by using the following operator command:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

2. Establish the priority for RSS within the WLM policy.

   The RRS priority needs to be equal to or higher than the dispatching priority of DB2. You can use the SYSSTC service class for the RRS address space to achieve a higher dispatching priority.

3. Move the ATRRRS procedure that is supplied by IBM from SYS1.SAMPLIB to your SYS1.PROCLIB as member RRS, and make any necessary changes. This JCL procedure is used to start RRS.

   You can use a member name other than RRS, but the first four characters of this name must match the subsystem name that you specify in the IEFSSN*xx* member of SYS1.PARMLIB in the next step.

   The GNAME value in the procedure must match the *gname* that you specified when defining the log streams. The default value for *gname* at RRS startup is the Sysplex name.

   Example: The following example shows a sample procedure to start RRS.

```
//RRS   PROC GNAME='',CTMEM=''
//RRS   EXEC PGM=ATRIMIKE,REGION=0M,TIME=NOLIMIT,
//          PARM='GNAME=&GNAME CTMEM=&CTMEM'
//
```

4. Define RRS as a subsystem to z/OS by editing the active IEFSSN*xx* member of SYS1.PARMLIB to include the following entry:

```
SUBSYS SUBNAME(RRS) /* RESOURCE RECOVERY SERVICES */
```

   The subsystem name can be RRS or any other name that you choose. However, the first four characters (or fewer characters if the name is shorter) must match the name of the JCL procedure to start RRS.

5. Start RRS by using the following operator command:

```
START RRS
```

   If you subsequently need to stop RRS, use one of the following operator commands:

   - `SETRRS CANCEL`
   - `SETRRS SHUTDOWN`

   If neither of these commands stops RRS, use the `FORCE RRS,ARM` command. In this command, RRS is the subsystem name that your installation assigned to RRS in PARMLIB member IEFSSN*xx*.

Related concepts:

RRSAF (DB2 9 for z/OS Stored Procedures: Through the CALL and beyond)

Related reference:

z/OS MVS Programming: Resource Recovery

Systems Programmer's Guide to Resource Recovery Services (RRS)

Setting up a WLM application environment for stored procedures during installation

All stored procedures and user-defined functions (except native SQL procedures) must run in WLM-managed stored procedure address spaces. You must assign each stored procedure to a WLM application environment, which routes the stored procedure work to the appropriate WLM-managed address space.

About this task

If you use data sharing, see "Special considerations for WLM-managed stored procedures in coexistence" on page 723 for additional guidance about setting up WLM environments.

For DB2-supplied routines, you can use the provided definitions for core WLM environments as a starting point for defining your environments. You can use job DSNTIJRW to define and activate these core environments. For more information on using job DSNTIJRW, see "Setting up the DB2 core WLM environments during installation" on page 547. Job DSNTIJMV adds to SYS1.PROCLIB the startup address space procedures for the core WLM environments.

Recommendation: Use partitioned data set extended (PDSE) for load libraries that contain external modules for stored procedures. Using PDSEs might eliminate your need to stop and start the stored procedures address space due to growth of the load libraries. If a load library grows from additions or replacements, the library might need to be extended.

Procedure

To set up a WLM application environment for one or more stored procedures:

1. Gather requirements for the WLM environment. These requirements include:
   - The name of the DB2 subsystem or data sharing group where the stored procedure or user-defined function is to run.
   - The maximum number of task control blocks (TCBs) that can run concurrently in an address space that is initiated by the WLM environment. This setting, referred to as the NUMTCB, is the maximum number of procedures or functions that should run concurrently in that WLM-managed address space. The appropriate setting depends on the requirements or limitations of the individual stored procedure or user-defined function.
     - For recommended NUMTCB values for DB2-supplied stored procedures, see "DB2-supplied stored procedures and user-defined functions" on page 833.
     - For recommended NUMTCB values for your own stored procedures, see Assigning stored procedures and functions to WLM application environments (DB2 Performance). If a routine allocates a large amount of memory or if the external module for the routine is not re-entrant, the NUMTCB value should be 1. If there is a need for serialization, the NUMTCB value should also be 1.

- The WLM environment for a REXX stored procedure must have a NUMTCB setting of 1.

- Requirements for Authorized Program Facility (APF).
  - Stored procedures and user-defined functions that execute privileged instructions (for example, WLM_REFRESH) must run in a WLM environment that has an APF-authorized address space.
  - DB2 REXX stored procedures, such as DSNTPSMP, and other routines that use TSO/E and REXX facilities must run in a WLM environment that does not have an APF-authorized address space.
  - Most stored procedures can run in a WLM environment that either has or does not have an APF-authorized address space. However, running a routine in an APF-authorized environment is recommended only when doing so is necessary.
- Special data sets and DD names that are needed by the stored procedure or user-defined function. For example, all Java stored procedures implicitly require a Language Environment runtime options data set to be allocated to DD name JAVAENV. Other routines might also have an explicit requirement for a work file to be allocated to a particular DD name.

2. Create a JCL startup procedure for the WLM application environment.

   Use *ssnm*WLM (where *ssnm* is the DB2 subsystem ID) as the basis for creating the new startup procedure. *ssnm*WLM is the default name of the sample JCL procedure for starting WLM-established address spaces. If you specified a name in the WLM PROC NAME field on the DSNTIPX installation panel, the sample procedure has that specified name. The sample procedure is placed in your system PROCLIB by installation job DSNTIJMV. For the DSNACICS stored procedure, DSNTIJMV also provides sample procedure *ssnm*CICS, where *ssnm* is the subsystem ID. Tailor your new startup procedure as follows:

   - Change the APPLENV value to the name of the WLM application environment that you plan to define.
   - Change the startup procedure name to an appropriate procedure name.

     Recommendation: Specify the same name for the application environment name and the procedure name. Using the same name simplifies monitoring the WLM environment definitions.
   - Change the value of DB2SSN to your DB2 subsystem name.
   - Ensure that the NUMTCB value is appropriate. If the minimum number of address spaces is specified, the NUMTCB value will be the number of task control blocks that are started in an address space.

     Recommendation: If you want to be able to easily change the NUMTCB value, specify the NUMTCB value here, and do not specify this parameter on the application environment definition panel when you define the WLM application environment. Specifying NUMTCB on the environment definition panel associates this parameter with the WLM application environment definition. Any changes to the JCL startup procedure can be made by refreshing the WLM application environment. However, changes to the WLM application environment require that you reinstall the z/OS service policy at an LPAR or sysplex level. If you specify the NUMTCB parameter for both the WLM application environment and the startup procedure, the value in the WLM application environment overrides the value in the startup procedure.
   - If the environment is one in which the cost of starting new address spaces is very high, you can use the MNSPAS parameter. The MNSPAS parameter

specifies the minimum number of address spaces that are to be started and maintained. The value can be 0 through 50. The default value is 0, which means that no minimum number of address spaces is specified. If the number of address spaces is at the specified minimum, cancelling an address space causes WLM to start another one to maintain the minimum. If you refresh a WLM environment, WLM starts a number of new address spaces as needed (or the specified minimum number) and stops the old address spaces.

If the MNSPAS parameter is specified, the NUMTCB value is the number of task control blocks that are started in an address space. If the MNSPAS parameter is not specified, the NUMTCB value is the maximum number of task control blocks that can be started in an address space.

- Edit the data set names to match your data set names.
- Ensure that the library in which the stored procedure resides is in the STEPLIB concatenation.
- Ensure that the IBM Language Environment runtime library is specified in the STEPLIB concatenation, unless your system includes the Language Environment runtime library in the system link list.
- If you want to run the stored procedure in a WLM environment that has an APF-authorized address space, ensure that all the libraries in the STEPLIB concatenation are defined to the Authorized Program Facility.
- Add any special DD statements that are required for the stored procedures or user-defined functions that are to run in the WLM environment. If you plan to use stored procedures that use the ODBA interface to access IMS databases, make the following additional changes:
  - Add the data set name of the IMS data set that contains the ODBA callable interface code to the end of the STEPLIB concatenation. The name of this data set is usually IMS.RESLIB.
  - After the STEPLIB DD statement, add a DFSRESLB DD statement that names the IMS data set that contains the ODBA callable interface code.

Example of a startup procedure: The following example shows a startup procedure for a WLM application environment for external SQL, COBOL, C, C++, PL/I, and assembler stored procedures. This startup procedure for external SQL stored procedures needs one unauthorized data set included in STEPLIB.

```
//*******************************************************************
//* JCL FOR RUNNING THE WLM-ESTABLISHED STORED PROCEDURES
//* ADDRESS SPACE
//* RGN     -- THE MVS REGION SIZE FOR THE ADDRESS SPACE.
//* DB2SSN  -- THE DB2 SUBSYSTEM NAME.
//* NUMTCB  -- THE NUMBER OF TCBS USED TO PROCESS
//*            END USER REQUESTS.
//* APPLENV -- THE MVS WLM APPLICATION ENVIRONMENT
//*            SUPPORTED BY THIS JCL PROCEDURE.
//*
//*******************************************************************
//DB9AWLM PROC RGN=0K,APPLENV=WLMENV,DB2SSN=DB9A,NUMTCB=40,MNSPAS=3
//IEFPROC EXEC PGM=DSNX9WLM,REGION=&RGN,TIME=NOLIMIT,
//             PARM='&DB2SSN,&NUMTCB,&APPLENV,&MNSPAS'
//STEPLIB  DD  DISP=SHR,DSN=DB9AU.RUNLIB.LOAD
//         DD  DISP=SHR,DSN=CEE.SCEERUN
//         DD  DISP=SHR,DSN=DB9A9.SDSNEXIT
//         DD  DISP=SHR,DSN=DB9A9.SDSNLOAD
```

3. Define the application environment to WLM by using one of the following methods.

- Use the WLM administrative application (IWMARIN0) to define a new environment or copy or modify an existing one.
  a. In the first WLM panel, Definition Menu, enter 9 for Application Environments.

     The application environment definition panel is displayed.
  b. On the application environment definition panel, specify the appropriate action code:
     - Create a new WLM application environment.
     - Copy an existing WLM application environment.
     - Modify an existing WLM application environment.

     The appropriate panel for creating, copying or modifying the application environment is displayed.
  c. If you are creating or modifying a WLM environment, specify the appropriate options for the WLM application environment definition, using the same application environment name and procedure name that you specified for the startup procedure.
  d. After you specify the appropriate options, press Enter to create the application environment definition.
- Modify and run sample job DSNTIJRW. As provided, DSNTIJRW creates the core DB2 WLM environments, but you can modify it to define your own WLM environments.

4. Install the WLM service definition that contains information about the application environment by opening the WLM Definition Menu panel and selecting the **Install** function in the **Utilities** menu bar.
5. Activate the WLM service policy from the installed service definition by completing one of the following actions:
   - Issue the VARY WLM command as follows:

     `VARY WLM,POLICY=policy-name`
   - On the WLM Definition Menu panel, choose the **Activate service policy** option from **Utilities** menu bar.
6. Set up security for the WLM application environments and for creating stored procedures by completing the following actions:
   a. Optional: Specify which WLM-established address spaces can run stored procedures. If you do not complete this step, any WLM-established address space can run stored procedures.
   b. Grant access to users to create procedures in specific WLM address spaces.
   c. Grant access to users to create procedures in specific schemas. Use the GRANT statement with the CREATIN option for the appropriate schema.
   d. Grant access to users to create packages for procedures in specific collections. Use the GRANT statement with the CREATE option for the appropriate collection.

      Requirement: For native SQL procedures, the collection name for the package must be the same as the schema name.
   e. Grant access to refresh the WLM environments to the appropriate people.

   For more detail, see Managing authorization for stored procedures (Managing Security).
7. Ensure that the ID that is used to run the JCL startup procedure for the WLM application environment has permission to use RRSAF. Each time one of the DB2 WLM address spaces is started, it uses RRSAF to attach to DB2.

What to do next

When you create or alter a stored procedure, specify the WLM application environment that DB2 is to use in the WLM_ENVIRONMENT option of the CREATE or ALTER PROCEDURE statement. When you specify environments, be aware that:

- Routines that have low memory requirements and that are frequently called should not share an environment with routines that have high memory requirements and are rarely called. Otherwise, storage is wasted.
- Stored procedures can be grouped according to the authorization settings for a WLM environment.
- WLM environments should not be shared among routines that use the same DD name for different purposes.
- WLM environments should not be shared among routines if one routine allocates data sets in the environment and that routine is run only occasionally. If other routines that do not use the data sets use the environment, the data sets are kept artificially allocated, which might interfere with other tasks.

Related concepts:

"WLM-established stored procedures address spaces storage requirements" on page 48

Related tasks:

Setting up and managing Workload Manager (DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond)

Managing authorization for stored procedures (Managing Security)

Refreshing a WLM application environment for stored procedures (DB2 Administration Guide)

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

"DSNTWLMB tool" on page 847

GRANT (collection privileges) (DB2 SQL)

GRANT (schema privileges) (DB2 SQL)

Related information:

Using Operator Commands for Application Environments (z/OS MVS Planning: Workload Management)

Installation step 20: Set up DB2-supplied routines

Installation jobs are provided for the setup of all DB2-supplied routines and supporting objects.

Setting up the DB2 core WLM environments during installation

DB2 stored procedures (except native SQL procedures) must run in WLM-managed stored procedure address spaces. You must assign each stored procedure to a WLM application environment, which routes the stored procedure work to the appropriate WLM-managed address space.

About this task

For DB2-supplied routines, you can use the provided definitions for core WLM environments as a starting point for defining your environments. You can also

manually define the WLM environments by using the IWMARIN0 application, as described in "Setting up a WLM application environment for stored procedures during installation" on page 543.

Exception: You do not need to complete this configuration task for native SQL procedures unless they call at least one external stored procedure or external SQL procedure. This task also applies to native SQL procedures that call at least one user-defined function or that are defined with ALLOW DEBUG MODE or DISALLOW DEBUG MODE. Otherwise, this task applies to only user-defined functions, external stored procedures, external SQL procedures, and DB2-supplied stored procedures

Procedure

To set up the provided core WLM environments:

Run job DSNTIJRW.
Job DSNTIJRW invokes the DSNTWLMB tool to create the core WLM environments. The CLIST assigns the core WLM environment names in DSNTIJRW according to the values that are specified on panel DSNTIPR1. For detailed information about the environments that are created by DSNTIJRW, see "Core WLM environments for DB2-supplied routines" on page 840.

Related concepts:

"WLM management of stored procedures" on page 839

Related reference:

"DSNTWLMB tool" on page 847

"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458

Installing DB2-supplied routines during installation

Two jobs install and validate the installation of DB2-supplied routines. These jobs are configured with the options that you specified on installation panel DSNTIPR1 and panels DSNTIPRA through DSNTIPRP.

Procedure

To install the DB2-supplied routines:

1. Run job DSNTIJRT to install and configure all DB2-supplied routines.
2. After DSNTIJRT successfully completes, run job DSNTIJRV to validate the installation of the routines.
3. Authorize the appropriate users to use one or more of these stored procedures by issuing the GRANT EXECUTE statement.

   As part of the installation process, job DSNTIJRT defines, binds, and grants access to the stored procedures and user-defined functions. Job DSNTIJRT grants EXECUTE authority to the authorization IDs that are specified for the GRANT EXECUTE fields on installation panels DSNTIPRB through DSNTIPRP. The default setting for GRANT EXECUTE is PUBLIC, meaning that anyone who can access DB2 can use these stored procedures.

   Recommendation: When you run the installation CLIST, use the GRANT EXECUTE fields on panels DSNTIPRB through DSNTIPRP to specify different access lists.

   Important: DSNAEXP can execute the EXPLAIN statement on any explainable SQL statement that is valid on the system, and the EXPLAIN output can reveal

potentially sensitive information. The privileges to run DSNAEXP should therefore be granted with consideration. For example, you should avoid granting access for DSNAEXP to PUBLIC.

4. Complete additional setup steps for any of the following routines that you plan to use.
   - The stored procedures for XML schema support
   - The administrative task scheduler routines and administrative enablement routines
   - The CICS transaction invocation procedure
   - The DSNAEXP EXPLAIN stored procedure
   - The common SQL API stored procedures
   - The WebSphere MQ user-defined functions
   - The web service user-defined functions

What to do next

After installing the DB2-supplied routines, you can choose to disable any of the routines that are unnecessary or unusable on your subsystem.

Related concepts:

"Configuration control statements for DB2-supplied routines" on page 849

Related tasks:

Granting privileges for executing stored procedures and stored procedure packages (Managing Security)

"Disabling DB2-supplied routines for validation" on page 851

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Job DSNTIJRT:

Job DSNTIJRT installs and configures DB2-supplied routines by executing program DSNTRIN.

DSNTIJRT provides a *configuration control statement* for each DB2-supplied routine and the mode for processing those statements. The configuration control statement specifies the WLM environment, list of authorization IDs that are allowed to execute the routine, and optional package owner for the routine. DSNTRIN reads and validates those statements and uses them to create and customize routines and supporting objects.

Format

DSNTRIN accepts the following parameters:

DB2SSN
Identifies the DB2 subsystem on which to install and configure the DB2-supplied routines.

MODE
During SQL processing, DSNTRIN configures each routine with the WLM environment specified in the configuration control statement for that routine. Processing behavior depends on the DSNTRIN run mode, the DB2 catalog mode, and which objects already exist.

INSTALL
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the routine already exists, it is updated, if necessary, to use the WLM environment specified in the configuration control statement for that routine. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. If a supporting object is used by more than one routine, the object is created when one of those routines is first created.

REINSTALL
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. Existing routines and any related created global temporary tables are first dropped before they are re-created. Related databases are not dropped if they already exist, but they are created if they do not exist. If a supporting object is used by more than one routine, the object is created when one of those routines is first created. Drop processing of routines and related objects that do not exist is bypassed.

REINITIALIZE
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. Existing routines, existing related created global temporary tables, and existing related databases are first dropped before they are re-created. After each SQL statement, a COMMIT statement is issued. If a supporting object is used by more than one routine, the object is created when one of those routines is first created.

INSTALL-PREVIEW
Simulate INSTALL mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

REINSTALL-PREVIEW
Simulate REINSTALL mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

REINITIALIZE-PREVIEW
Simulate REINITIALIZE mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

AUTHID
DSNTRIN uses this value as the CURRENT SQLID for creating and configuring most DB2-supplied routines. DSNTRIN also uses this value as the default OWNER for binding packages for the DB2-supplied routines that do not have the PKGOWNER specified in the configuration control statement.

SECDEFID
DSNTRIN uses this value as the CURRENT SQLID setting when creating and configuring DB2-supplied routines that are defined with the SECURITY DEFINER option.

Input

DSNTRIN allocates the following DD statements for input:

DBRMLIB
Location of the DBRMs for DB2-supplied routines. This is typically the *prefix*.SDSNDBRM target library.

SYSUT1
Internal work file for the DB2 installation bind interface (DSNTRBI).

CFIGIN
Location of the configuration control statements for DB2-supplied routines.

DB2OPT
This required DD can contain the following optional parameters:

STOGROUP
Specifies a storage group to use instead of the default SYSDEFLT storage group.

BP4K Specifies an alternate 4 K buffer pool for base table spaces that are created by DSNTRIN. The default 4 K buffer pool is BP0.

BP8K Specifies an alternate 8 K buffer pool for base table spaces that are created by DSNTRIN. The default 8 K buffer pool is BP8K0.

BP16K
Specifies an alternate 16 K buffer pool for base table spaces that are created by DSNTRIN. The default 16 K buffer pool is BP16K0.

BP32K
Specifies an alternate 32 K buffer pool for base table spaces that are created by DSNTRIN. The default 32 K buffer pool is BP32K.

LOBBP8K
Specifies an alternate 8 K buffer pool for LOB table spaces that are created by DSNTRIN. The default 8 K buffer pool is BP8K0.

LOBBP16K
Specifies an alternate 16 K buffer pool for LOB table spaces that are created by DSNTRIN. The default 16 K buffer pool is BP16K0.

Note: DSNTRIN does not create or activate this storage group or any of these buffer pools.

Output

DSNTRIN allocates the following DD statements for output:

SYSPRINT
Destination for echo of configuration control statements that are read and for messages that are generated during processing. Each processed routine generates at least one status message.

SYSTSPRT
Internal message file for the DB2 installation bind interface (DSNTRBI).

CFIGOUT
Destination for configuration control statements that are read by DSNTRIN. This output is useful for diagnostic purposes when DSNTRIN reports an error in a configuration control statement.

SQLOUT
Destination for DDL statements that are generated and processed by DSNTRIN. These statements are useful for diagnostic purposes.

BINDOUT
Destination for bind statements that are generated and processed by DSNTRIN. These statements are useful for diagnostic purposes.

JCLOUT
Destination for a generated sample job that is generated when one of the PREVIEW modes is specified. The job contains steps that process inline DDL and bind statements that are needed to install and configure missing, downlevel, or incorrectly configured DB2-supplied routines. This DD is required if you specify a PREVIEW mode.

Example of executing DSNTRIN in INSTALL mode:

```
//DSNTRIN  EXEC PGM=DSNTRIN
//         PARM=('DB2SSN(VA1A) MODE(INSTALL) AUTHID(AUTHID) SECDEFI-
           D(ADMF001)')
//DBRMLIB  DD DISP=SHR,DSN=DSN!!0.SDSNDBRM
//SYSUT1   DD UNIT=SYSDA,SPACE=(27930,(10,5)),
//            DCB=(RECFM=FB,LRECL=133)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//CFIGOUT  DD SYSOUT=*
//SQLOUT   DD SYSOUT=*
//BINDOUT  DD SYSOUT=*
//DB2OPT   DD *
 STOGROUP(DB2_ROUTINES_GROUP)
 BP4K(BP0)
 BP8K(BP8K0)
 BP16K(BP16K0)
 BP32K(BP32K)
 LOBBP8K(BP8K0)
 LOBBP16K(BP16K0)
/*
//CFIGIN   DD *
  ...
SYSPROC.DSNWSPM
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)

SYSPROC.DSNWZP
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)
  ...
//*
```

Example of executing DSNTRIN in INSTALL-PREVIEW mode:

```
//DSNTRIN  EXEC PGM=DSNTRIN
//         PARM='DB2SSN(VA1A) MODE(INSTALL-PREVIEW) AUTHID(AUTHID)'
//DBRMLIB  DD DISP=SHR,DSN=DSN!!0.SDSNDBRM
//SYSUT1   DD UNIT=SYSDA,SPACE=(27930,(10,5)),
//            DCB=(RECFM=FB,LRECL=133)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//CFIGOUT  DD SYSOUT=*
//SQLOUT   DD SYSOUT=*
//BINDOUT  DD SYSOUT=*
```

```
//JCLOUT   DD DSN=USER.JCLLIB(DB2INST),DISP=SHR
//DB2OPT   DD *
 STOGROUP(DB2_ROUTINES_GROUP)
 BP4K(BP0)
 BP8K(BP8K0)
 BP16K(BP16K0)
 BP32K(BP32K)
 LOBBP8K(BP8K0)
 LOBBP16K(BP16K0)
/*
//CFIGIN   DD *
  ...
SYSPROC.DSNWSPM
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)

SYSPROC.DSNWZP
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)
  ...
//*
```

Related concepts:

"Configuration control statements for DB2-supplied routines" on page 849

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Job DSNTIJRV:

Job DSNTIJRV validates the DB2-supplied routines by executing program DSNTRVFY. You should validate these routines after DB2 installation and after each phase of migration.

DSNTRVFY validates a single routine, multiple routines, or all DB2-supplied routines. DSNTRVFY produces a report that identifies routines that passed the validation and possible problem areas for routines that failed.

Format

DSNTRVFY accepts the following required parameters:

DB2SSN(*ssid*)
Identifies which DB2 subsystem to connect to for validating the DB2-supplied routines.

ROUTINE(*name*)
Identifies which DB2-supplied routine needs to be validated. The following values are supported for *name*:

- A routine name in the form of *schema-name.routine-specific-name*.
- The keyword **DD:SYSIN** to specify a list of routines. List the routines in the form of *schema-name.routine-specific-name* under the SYSIN DD statement with one routine per line.
- The keyword **ALL** to specify all DB2-supplied routines.

DSNTRVFY also accepts the following optional parameter:

AUTHID(*authorization ID*)
DSNTRVFY uses this value as the OWNER for BIND PACKAGE and BIND PLAN statements and as the CURRENT SQLID for issued SQL statements. If

this value is not specified, the primary authorization ID is used for BIND PACKAGE, BIND PLAN, and issued SQL statements.

Output

DSNTRVFY allocates the following required DD statements for output:

SYSPRINT
Contains messages generated during verification processing. Each processed routine can generate one or more messages.

SYSTSPRT
DSNTRVFY writes bind output to SYSTSPRT if an error occurs during bind processing.

The following JCL example is for validating a single routine:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(SYSPROC.DSNUTILS)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
```

The following JCL example is for validating multiple routines:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(DD:SYSIN)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  SYSPROC.DSNWZP
  SYSPROC.DSNACCOR
  SYSPROC.DSNUTILS
  SYSPROC.DSNUTILU
```

The following JCL example is for validating all routines:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(ALL) AUTHID(FVTGRP2)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
```

Processing

DSNTRVFY completes the following validation:

- Verifies that the routine is defined to the SYSIBM.SYSROUTINES catalog.
- Verifies that the collection ID defined in the SYSIBM.SYSROUTINES catalog table matches the collection ID used to bind the package.
- Verifies that the routine is currently defined with the DB2-reserved WLM application environment name DSNWLM_NOT_ENABLED, indicating that it is not enabled to run on the current DB2 subsystem. If it is, then a warning message is issued and the validation is bypassed.
- Verifies if the routine can be executed on the current DB2 catalog mode.
- Verifies the APF authorization requirements.
- Verifies if the WLM application environment is defined and available for use.

- Verifies if the NUMTCB setting is valid or within the recommended range.
- Verifies if the required DD statements for some of the routines are defined.
- Verifies if the WLM environment assigned to Java routines is enabled for Java.
- Verifies security coexistence requirements (for example, some routines should run on only their own WLM application environment).
- Verifies that the selected routine can be called successfully. Certain routines such as DSNACICS, DSNAIMS, MQSeries functions, and most debugger routines are restricted to basic validation and are not called.

If the validation is successful, the following message is issued:

```
DSNT027I DSNTRVFY VALIDATION FOR schema.routine-specific-name SUCCESSFUL
```

If the validation fails, messages are issued to explain the reasons for failure. Examine these messages and make necessary adjustments before running the validation job again.

Input

DSNTRVFY requires the following DD statements for input:

DBRMLIB
Identifies the library where the DBRM for the DSNTRVFY program is located. This is typically the *prefix*.SDSNDBRM target library.

SYSIN
Contains the list of DB2 routines. This is required when the validation type is for multiple routines (DD:SYSIN).

SYSUT1
Work data set required for bind processing.

Limitations

The following limitations apply to the DSNTRVFY program:

- The name of the routine in the form *schema-name.routine-specific-name* is limited to 72 characters, the maximum line length in JCL.
- Only Version 10 DB2-supplied routines are supported for validation.
- Some routines, if enabled, receive basic validation to ensure that they are defined correctly and assigned to a valid WLM application environment. However, validation will not extend to calling these routines for one or more of the following reasons:
  - Optional system support such as CICS, IMS, or MQSeries is required.
  - External setup on a remote server or workstation is required.
  - A call could introduce or modify data unexpectedly in a production environment.

  The DB2-supplied routines that are restricted to basic validation are:
  - DB2DEBUG.CREATE_SESSION
  - DB2DEBUG.DESTROY_SESSION
  - DB2DEBUG.GET_REPORT
  - DB2DEBUG.LIST_SESSION
  - DB2DEBUG.PUT_COMMAND
  - SYSFUN.SYSTS_ENCRYPT

- SYSPROC.ADMIN_COMMAND_UNIX
- SYSPROC.ADMIN_JOB_CANCEL
- SYSPROC.ADMIN_JOB_FETCH
- SYSPROC.ADMIN_JOB_QUERY
- SYSPROC.ADMIN_JOB_SUBMIT
- SYSPROC.DBG_ENDSESSIONMANAGER
- SYSPROC.DBG_INITIALIZECLIENT
- SYSPROC.DBG_RECVCLIENTREPORTS
- SYSPROC.DBG_SENDCLIENTREQUESTS
- SYSPROC.DBG_SENDCLIENTCOMMANDS
- SYSPROC.DBG_TERMINATECLIENT
- SYSPROC.DSNACICS
- SYSPROC.DSNAIMS
- SYSPROC.DSNAIMS2
- SYSPROC.DSNLEUSR
- SYSPROC.SYSTS_ALTER
- SYSPROC.SYSTS_CREATE
- SYSPROC.SYSTS_DROP
- SYSPROC.SYSTS_RESTORE
- SYSPROC.SYSTS_START
- SYSPROC.SYSTS_STOP
- SYSPROC.SYSTS_TAKEOVER
- SYSPROC.SYSTS_UPDATE
- All MQSeries, web services, and SOAP functions (those with schema DB2MQ, DB2MQ1C, DB2MQ2C, or DB2XML)

Installation step 21: Back up the DB2 directory and catalog: DSNTIJIC

For backup purposes, create a copy of the DB2 directory and catalog.

About this task

Attention: If you do not create this copy, DB2 can start. However, if errors that require you to reinstall DB2 occur in the directory or catalog and you do not have backup copies, you lose all your tables and data.

Recommendation: Copy the catalog and directory at least daily if you make any changes in them. Recovery time for these databases is longer if the copies are not current, and the entire subsystem is affected.

Procedure

To create an image copy of the directory and catalog:

1. Examine job DSNTIJIC for the following information:
   - The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also the unit name for the default archive log data sets.
   - Expiration date or retention period. You can add a retention period or an expiration date to the job.

- The user on the JOB statement. Ensure that the user is authorized. If DB2 is started under ACCESS(MAINT), this must be the same user that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1.

2. Run job DSNTIJIC.

   The DSNTIJIC job contains a list of all the DB2 directory and catalog table spaces. When you run job DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. Having copies of table spaces enables you to recover the DB2 catalog and DB2 directory in case of a failure.

3. If the DSNTIJIC job fails or abends, verify that no problems exist with the tape setup for image copy. If you find no problems with the tape setup, examine the utility job output (JOBLOG) or the console log for problems. For example, look for I/O errors or incorrect sizes.

What to do next

Run the DSNTIJIC job periodically, perhaps daily or weekly, to reduce the amount of time required for recovering the directory or catalog. The copied data and log data sets are needed for recovery.

During the ISPF tailoring session, you named one or two IDs to have installation SYSADM authority. One of these users can now grant various levels of authority to other users. You can use SPUFI or a job similar to DSNTIJSG to perform the authorization. To grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF, issue the following statement in SPUFI:

```
GRANT SYSOPR TO userid
```

Related concepts:

Privileges and authorities (Managing Security)

Related tasks:

"Installation step 17: Define and bind DB2 objects: DSNTIJSG" on page 533

"Migrating an existing data sharing group to a new release" on page 711

Installation step 22: Set up the administrative task scheduler

You can use the administrative task scheduler to execute administrative tasks according to a time-based or event-based schedule.

Before you begin

You should have run job DSNTIJRA and job DSNTIJIN before you set up the administrative task scheduler.

About this task

Restriction: After migration to conversion mode (from Version 8 or Version 9.1), the administrative task scheduler is disabled until you run job DSNTIJRT.

The following figure shows an overview of the steps for installing and configuring the administrative task scheduler.

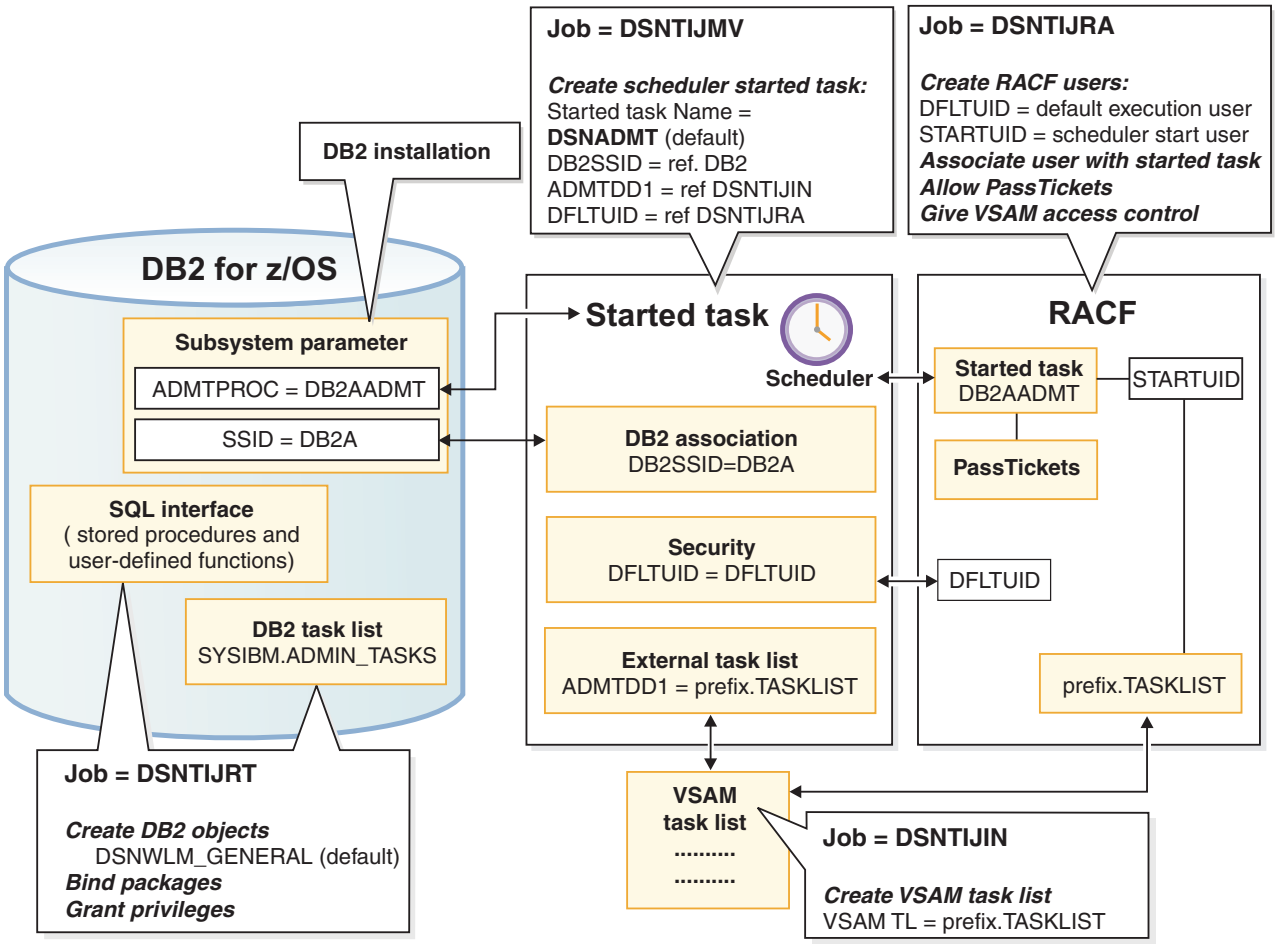


Figure 74. Setting up the administrative task scheduler

Procedure

To set up the administrative task scheduler:

1. Customize the administrative task scheduler address space startup procedure (xxxxADMT) in job DSNTIJMV. The name of the startup procedure needs to match the value of the ADMTPROC subsystem parameter. Before using the startup procedure, locate and review the settings for the following parameters:

   DB2SSID
   The name of this DB2 subsystem.

   DFLTUID
   The default ID that is used by the administrative task scheduler to execute its tasks. DFLTUID must be different than the ID that is used to start this address space.

   TRACE
   Indicates whether to activate tracing for the administrative task scheduler. The value can be ON or the default value of OFF.

   You can specify other parameters, such as ERRFREQ, MAXHIST, MAXTHD, or STOPONDB2STOP in the startup procedure similarly to the three parameters DB2SSID, DFLTUID and TRACE.

   ERRFREQ
   Specifies how frequently message DSNA679I displays on the console,

indicating that one of the redundant, active copies of the task list is not accessible. By default, this message displays on the console once per minute, for example, when DB2 is offline.

MAXHIST
Specifies the number of status entries per task that are stored. This parameter is a positive integer with a default value of 10. When the limit is reached, the oldest status entries are deleted.

MAXTHD
Specifies the maximum number of execution threads for an administrative task scheduler. The default value is 99.

STOPONDB2STOP
Specifies that the administrative task scheduler will terminate when DB2 is stopped.

Also locate and review the ADMTDD1 DD statement. The VSAM data set that is indicated must match the VSAM task list data set that is defined in job DSNTIJIN.

The following example shows how these parameters can be updated.

```
//DSNADMT  PROC LIB='DSN!!0.SDSNLOAD',
//          DB2SSID=DSN,
//          DFLTUID=DFLTUID,
//          TRACE=OFF,
//          MAXTHD=10
//*
//STARTADM EXEC PGM=DSNADMT0,DYNAMNBR=100,REGION=0K,
//          PARM=('DB2SSID=&DB2SSID',
//          ' DFLTUID=&DFLTUID',
//          ' TRACE=&TRACE'
//          ' MAXTHD=&MAXTHD'
//          ' ERRFREQ=1440'
//          ' STOPONDB2STOP')
//STEPLIB  DD DISP=SHR,DSN=&LIB
//ADMTDD1  DD DISP=SHR,DSN=DSNC!!0.TASKLIST
```

2. Ensure that the administrative task scheduler routines were enabled. They were enabled by job DSNTIJRT.
3. Create a job to make image copies of the administrative task scheduler table spaces. Establish a schedule for making image copies. In general, you should back up important databases on a regular basis. The database for the administrative task scheduler is no exception and should be copied on the same frequency as the DB2 catalog and directory. The following example shows how to copy the table spaces in this database to stacked data sets on tape with a retention period of 99 days:

```
//*
//DSNTIC  EXEC PGM=DSNUTILB,PARM='DSN,IMAGCOPY',COND=(4,LT)
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//DSNTRACE DD  SYSOUT=*
//SYSLISTD DD  *
  LISTDEF DSNLDEF
          INCLUDE TABLESPACES DATABASE DSNADMDB
//SYSIN    DD  *

  TEMPLATE DSNTPLT
           DSN(prefix.IMAGCOPY.&DB..;&TS.;)
           DISP (NEW,KEEP,DELETE)
           UNIT TAPE
           STACK YES
```

```
            RETPD 99
       COPY LIST DSNLDEF
            COPYDDN(DSNTPLT)
/*
```

Related concepts:

"Administrative task scheduler address space storage requirements" on page 48

"Job DSNTIJRT" on page 549

Multi-threading in the administrative task scheduler (DB2 Administration Guide)

Related tasks:

"Installation step 4: Define system data sets: DSNTIJIN" on page 514

"Additional steps for enabling the administrative task scheduler and administrative enablement routines" on page 854

Scheduling administrative tasks (DB2 Administration Guide)

Related information:

DSNA677I (DB2 Messages)

Installation step 23: Verify a successful installation

To verify that your installation process ran successfully, you can run the sample applications.

About this task

Select the phases that you need to run based on the attachment facilities you installed, the languages you use, and whether the sample objects exist.

Related tasks:

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Installation step 24: Enable additional capabilities for DB2

After you complete the previous steps, you can enable additional capabilities for DB2, such as support for a communications network and support for Java.

Support for a communications network

After you install DB2, you can install support for a communications network.

Recommendation: If you plan to use the distributed data facility (DDF), become familiar with the DDF function and the kind of network communications that you want to use to communicate with this DB2 subsystem. Depending on the network communications that you use, you must install Virtual Telecommunications Access Method (VTAM), z/OS UNIX System Services TCP/IP support, or both. In Version 10, you can set up DDF not to activate SNA/APPC support. Therefore, in Version 10, DDF does not require VTAM to be installed. You might need to install VTAM for other reasons.

Installing NetView®

If you use DDF, you might want to install NetView so that DB2 can send alerts to NetView if DB2 detects security exposures or protocol errors.

Related concepts:

"Customize VTAM for DB2" on page 885

"TCP/IP terminology" on page 918

Related tasks:

"Connecting systems with VTAM" on page 883

Related reference:

VTAM Network Implementation Guide

IBM Tivoli NetView for z/OS (Installation: Getting Started)

Support for Java

If you plan to use Java, or any function that is dependent on Java, you need to install support for it.

For information about installing support for Java, see Installing the IBM Data Server Driver for JDBC and SQLJ (Application Programming for Java).

Installing a second DB2 subsystem on the same operating system

If you are considering installing a second DB2 subsystem on the same operating system, first see the implications of a second DB2 subsystem.

Implications of a second DB2 subsystem:

The primary consideration in planning for a second DB2 subsystem is its purpose. Using a second subsystem has a substantial impact on your environment.

Second DB2 subsystems are not uncommon; organizations use a second DB2 subsystem to:

- Run separate service levels of the code. This can provide more extensive testing of preventive service before use with a production system.
- Run separate releases of the code. However, two different releases of DB2 on the same system must have separate libraries.
- Separate test and production activities. This setup can improve DB2's performance and availability for production.

  For example, suppose the processor that runs your production subsystem fails. If your test subsystem is on another processor, you can stop the test subsystem and start the production subsystem on that processor. This only works if the requisite DB2 data sets are on shared storage devices and you have used global resource serialization (GRS) (or an equivalent) to protect the production DB2 data sets.
- Prevent access by one class of users to certain data. If this is your primary purpose, reconsider the DB2 authorization scheme.

The table below lists other implications of a second DB2 subsystem.

Table 84. Considerations for installing a second DB2 subsystem

Considerations	Decisions that you need to make
Libraries	A single, shared library or separate libraries
RACF protection	Resource and ID for the two subsystems
DB2 logging	BSDS, active, and archive log space requirements
Database space requirements	DB2 directory, DB2 catalog, and user data

Table 84. Considerations for installing a second DB2 subsystem (continued)

Considerations	Decisions that you need to make
Performance	Processor and main storage use
Distributed data facility requirements	Coordination of location names, logical unit names, and network passwords with remote DB2 subsystems
Common service area (CSA) requirement	DB2 and the IRLM
Shared storage devices	Giving different names to active and archive log data sets, or including those data sets in the GRS inclusion list

Each subsystem must have a separate *prefix*.SDSNEXIT library. Sharing the *prefix*.SDSNSAMP library requires coordination to avoid overlaying parameter members.

Related concepts:

"Active log data sets storage requirements" on page 32

"IRLM address space (IRLMPROC) storage requirements" on page 47

Related tasks:

"Loading DB2 libraries for a second DB2 subsystem"

"Connecting distributed database systems" on page 877

"Enabling multiple DB2 subsystems to share disk storage" on page 567

Establishing RACF protection for DB2 (Managing Security)

Related information:

DB2 for z/OS Program Directories

Loading DB2 libraries for a second DB2 subsystem:

If you are installing a second DB2 subsystem on the same operating system, you might need to create separate libraries for each subsystem.

About this task

If the systems have different code releases or different service levels, they must have separate libraries.

Procedure

To load libraries for a second DB2 subsystem:

1. Separately plan the space for each library.
2. Load the libraries, using different prefixes for each library and for the SMP/E data sets or separate SMP/E zones.

Tailoring installation jobs for a second DB2 subsystem:

Before installing a second DB2 subsystem on the same operating system, change the parameter values that are to be used by the installation jobs during the installation.

About this task

The following table shows the parameter values that you can change. Some of these changes are required. The table includes the installation panel on which the parameter value can be changed, the panel parameter name, and any comments that pertain to the parameter.

Table 85. Parameters to change when installing the second subsystem

Installation panel	Action
DSNTIPA1	For separate libraries, change LIBRARY DATA SET NAME PREFIX and DATA SET NAME PREFIX.
DSNTIPA2	You must change CATALOG ALIAS. You can change the values for the DATA CLASS, MGMT CLASS, and STOR CLASS fields.
DSNTIPA3	You can change the values for the VOL/SER, DATA CLASS, MGMT CLASS, and STOR CLASS fields.
DSNTIPK	Check all values and make appropriate changes.
DSNTIPH	Check all values and make appropriate changes.
DSNTIPT	Check all values and make appropriate changes.
DSNTIPU	Check all values and make appropriate changes.
DSNTIPW	Check all values and make appropriate changes.
DSNTIPD	Check all values and make appropriate changes.
DSNTIP7	Check all values and make appropriate changes.
DSNTIP9	Check all values and make appropriate changes.
DSNTIPE	Check all values and make appropriate changes.
DSNTIP1	Check all values and make appropriate changes.
DSNTIP2	Check all values and make appropriate changes.
DSNTIPN	Check all values and make appropriate changes.
DSNTIPO	Check all values and make appropriate changes.
DSNTIPF	Check all values and make appropriate changes.
DSNTIP4	Check all values and make appropriate changes.
DSNTIP8	Check all values and make appropriate changes.
DSNTIPI	Change SUBSYSTEM NAME.
DSNTIPJ	Check all values and make appropriate changes.
DSNTIP6	Check all values and make appropriate changes.
DSNTIP61	Check all values and make appropriate changes.
DSNTIPP	Change ICF catalog to match the new ICF catalog. Change all passwords.
DSNTIPP1	Check all values and make appropriate changes.
DSNTIPM	Change SUBSYSTEM NAME and SUBSYSTEM PREFIX.
DSNTIPL	Check all values and make appropriate changes. You must change data set names and prefixes.
DSNTIPL1	Check all values and make appropriate changes.
DSNTIPA	Check all values and make appropriate changes. You must change data set names and prefixes.

Table 85. Parameters to change when installing the second subsystem (continued)

Installation panel	Action
DSNTIPS	Check all values and make appropriate changes. You must change data set names and prefixes.
DSNTIPR	Check all values and make appropriate changes. You must change names and password.
DSNTIP5	Check all values and make appropriate changes.
DSNTIPX	Check all values and make appropriate changes.
DSNTIPR1	Check all values and make appropriate changes.
DSNTIPRA - DSNTIPRP	(Optional) Check all values and make appropriate changes.
DSNTIPZ	Check all values and make appropriate changes.
DSNTIPY	Check all values and make appropriate changes.
DSNTIPC	Check values of DSMAX, EDMPOOL, EDMSTMTC, EDMDBDC, SRTPOOL, and MAXRBLK; no other changes can be made.

Procedure

To tailor the installation jobs:

Use the installation CLIST to change parameter values as described in the previous table.

Related tasks:

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

Installing a second DB2 subsystem:

When you install a second DB2 subsystem, some installation jobs are optional, some jobs are required, and some jobs are required under certain circumstances.

Procedure

To install a second subsystem:

1. Optional: Run job DSNTIJSS You can use job DSNTIJSS to create an SMS environment for the DB2 catalog and directory data sets. Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability.
2. Run job DSNTIJCA if you are defining a new ICF catalog.
3. Run job DSNTIJMV. Add another subsystem name and subsystem recognition character to IEFSSN*xx*. LNKLST*xx* modifications are needed only for separate libraries.

   For separate libraries, add STEPLIB statements to the precompile and bind steps for program preparation. Add STEPLIB statements for the DB2 offline utilities. Choose a naming convention for any new procedures, and change those as needed.

For a single library, you must add the exit module data set (*prefix*.SDSNEXIT) to the STEPLIB statement to contain your changed subsystem parameter. Put this data set first in the STEPLIB concatenation.

4. Run job DSNTIJIN. It allocates the following data sets, which contain the sample plans and programs:
   prefix.DBRMLIB.DATA
   prefix.SRCLIB.DATA
   prefix.RUNLIB.LOAD

   Two subsystems cannot share these data sets. If undetected sharing occurs, data could be lost. If you use the same library prefix for both subsystems, change the name of the data sets, unless you do not need them on the first subsystem. Subsequent jobs overwrite them, and then the plans that are bound in the first subsystem do not work with the new load modules.
5. Run job DSNTIJUZ without changes if you use separate libraries. For a single library, provide a separate *prefix*.SDSNEXIT data set for each subsystem.

   The default SSID displayed by certain panels and procedures is the same for every subsystem. Ensure that the correct subsystem is specified in these cases.
6. Run jobs DSNTIJID, DSNTIJIE, and DSNTIJIF to initialize the DB2 data sets.
7. Run job DSNTIJTC to invoke the CATMAINT utility to tailor your Version 10 catalog.

What to do next

If you are using RACF, you can define new user profiles and IDs to provide a separate level of security for each DB2 subsystem.

Related tasks:

Granting users and groups access (Managing Security)

Connecting the TSO attachment facility to a second DB2 subsystem:

After you install a second DB2 subsystem on the same operating system, you should connect the TSO attachment facility to that subsystem.

Procedure

To connect the TSO attachment facility:

1. Add STEPLIB statements to the logon procedure if you are using a separate library for the second subsystem or you are managing two different SDSNEXIT data sets. The STEPLIB statement must be authorized using the authorized program facility (APF).

   A DB2 logon procedure can use only one set of DB2 libraries.
2. Concatenate the CLISTs by updating the SYSPROC library if you decide to have a separate library for the second subsystem.
3. Concatenate the DB2I panels by updating the ISPPLIB library if you decide to have a separate library for the second subsystem.
4. Concatenate messages by updating the ISPMLIB library if you decide to have a separate library for a second subsystem.
5. Update the ISPF primary option panel (ISR@PRIM) if you decide to have a separate library for the second subsystem.
6. Grant authorization on both subsystems as necessary.
7. Specify the new subsystem name on the DB2I defaults panel as required.

Connecting the IMS attachment facility to a second DB2 subsystem:

After you install a second DB2 subsystem on the same operating system, you should connect the IMS attachment facility to that subsystem.

Procedure

To connect the IMS attachment facility:

1. For separate libraries, change the STEPLIB DD statement in your startup procedure to refer to the new libraries.
2. For separate libraries, change the DFSESL DD statement in your startup procedure to refer to the new libraries.
3. Define a new subsystem name, language interface token (LIT), and command recognition character (CRC).
4. Define a new language interface and reassemble.
5. In the linkage editor JCL, specify the library that contains the new language interface.

Preparing a second DB2 subsystem for use:

After you have installed a second DB2 subsystem on the same operating system, you must run some jobs that prepare the subsystem for use.

Procedure

To prepare your second DB2 subsystem for use:

1. IPL z/OS. This step is required for any SYS1.PARMLIB changes. Alternatively, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands to define the subsystem to z/OS until the next IPL.
2. Start DB2 by using the new command prefix (formerly called the subsystem recognition character) that you named during the installation process.
3. Run job DSNTIJTC.
4. Run job DSNTIJTM. If you use the same library prefix for both subsystems, change the name of the data sets listed below, unless you do not need them on the first subsystem. Subsequent jobs write information in these data sets and prevent use of the new load module DSNTIAD on the second subsystem with the previously bound plan DSNTIA91 on the first subsystem.
   prefix.DBRMLIB.DATA
   prefix.RUNLIB.LOAD
5. Run job DSNTIJSG.
6. Optional: Run job DSNTIJRW to define core WLM environments for DB2-supplied stored procedures and user-defined functions.
7. Run job DSNTIJRT to install all DB2-supplied routines.
8. Run job DSNTIJRV to validate the DB2-supplied routines after they have been installed.
9. Run job DSNTEJ*xx*. If you use the same library prefix for both subsystems, change the name of the data sets listed below, unless you do not need them on the first subsystem. Subsequent jobs overwrite information in these data sets and prevent use of the new load modules of sample programs on the second subsystem with the previously bound sample plans on the first subsystem.
   prefix.DBRMLIB.DATA
   prefix.SRCLIB.DATA
   prefix.RUNLIB.LOAD

Related reference:

SETPROG Command

SETSSI Command

Verifying your installation process for a second DB2 subsystem:

After you have installed a second DB2 subsystem and prepared it for use, you should verify a successful installation before you use the subsystem.

Procedure

To verify your installation:

Run select phases of the sample applications. Select the phases that you need based on the attachment facilities that you installed, the languages that you use, and whether the sample objects exist.

Related tasks:

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Enabling multiple DB2 subsystems to share disk storage

If you use a non-data sharing environment, you can enable multiple DB2 subsystems to share disk storage. If you use a data sharing environment, the logs must be on shared disk storage.

Procedure

To share disk storage among DB2 subsystems and avoid problems with your active and archive log data sets:

- Ensure that each subsystem has unique log data set names. This prevents situations like the following:

  Subsystem A on operating system 1 and subsystem B on operating system 2 share the same z/OS catalog name, and their log data set names are the same. You start subsystem B while subsystem A is still running on operating system 1. This causes log data sets to be allocated for subsystem A, even though they already exist.
- Use GRS, or an equivalent, and include your active and archive log data sets in the GRS inclusion list. This prevents situations like the following:

  Subsystem A on operating system 1 and subsystem B on operating system 2 share disk storage, and the active log is in a shared disk volume. Subsystem B fails. You attempt to start subsystem B, but you accidentally start subsystem A on operating system 2, even though it is still running on operating system 1. This causes log data sets to be allocated for subsystem A, even though they already exist.

Data loading from remote locations

You can use an SQL cursor with the LOAD utility to load data from a remote location.

Before loading data, you must bind the DSNUT101 package at each location from which you want to load data. A local package for DSNUT101 is bound by installation job DSNTIJSG when you install or migrate to a new version of DB2 for z/OS.

Special packages and plans for SPUFI

Special packages and plans for SPUFI are required in some situations.

Running SPUFI at remote systems:

You can use SPUFI to connect to and then execute SQL statements at a remote location.

Procedure

GUPI To run SPUFI at a remote location:

1. Ensure that DDF is started on the DB2 subsystem where SPUFI runs.
2. Bind the SPUFI packages on each target remote location that is not a DB2 for z/OS server, or on each target remote DB2 for z/OS location where the SPUFI packages have not been bound. For example, use the following commands:

```
BIND PACKAGE (location_name.DSNTIAP) MEMBER(DSNTIAP)
            ACTION(ADD) ENCODING(EBCDIC) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.DSNESPCS) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(CS) ENCODING(EBCDIC) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.DSNESPRR) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(RR) ENCODING(EBCDIC) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.DSNESPUR) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(UR) ENCODING(EBCDIC) LIB('prefix.SDSNDBRM')
```

   If a BIND PACKAGE command fails because the package already exists, verify that the time and date formats that are returned by the existing packages are satisfactory. If they are, the existing packages can be used without any change to the package list in the SPUFI plans. However, if you need to change the time and date formats that are returned by the existing packages, you must bind new packages with different collection identifiers that have been agreed to by the database server.

   For example, if the collection identifiers are PRIVATCS and PRIVATRR, the commands for doing a remote bind are as follows:

```
BIND PACKAGE (location_name.PRIVATCS) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(CS) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.PRIVATRR) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(RR) LIB('prefix.SDSNDBRM')
```

3. Bind the SPUFI plans to include the packages for all target remote locations, including other DB2 for z/OS servers. To include all locations, use an asterisk as the location name of the collection ID. For example, use the following commands:

```
BIND PLAN(DSNESPCS) -
     PKLIST(*.DSNESPCS.DSNESM68, -
            *.DSNTIAP.DSNTIAP) -
     ISO(CS) CURRENTDATA(YES) ENCODING(EBCDIC) ACTION(REPLACE)
BIND PLAN(DSNESPRR) -
     PKLIST(*.DSNESPRR.DSNESM68, -
            *.DSNTIAP.DSNTIAP) -
     ISOLATION(RR) ENCODING(EBCDIC) ACTION(REPLACE)
BIND PLAN(DSNESPUR) -
     PKLIST(*.DSNESPUR.DSNESM68, -
            *.DSNTIAP.DSNTIAP) -
     ISOLATION(UR) ENCODING(EBCDIC) ACTION(REPLACE)
```

   The SPUFI plans at the DB2 system must be rebound because the location name parameter (which is usually optional) must be explicitly specified for the

remote access functions to construct the correct package name. (SPUFI does not use the SQL statement SET CURRENT PACKAGESET.) The location name entry in the package list must precede any pattern-matching character entry. For example, the package list for the DSNESPCS plan is as follows:

```
PKLIST(location_name.PRIVATCS.DSNESM68, -
       *.DSNESPCS.DSNESM68, -
      *.DSNTIAP.DSNTIAP)
```

The package list for the DSNESPRR plan is as follows:

```
PKLIST(location_name.PRIVATRR.DSNESM68, -
       *.DSNESPRR.DSNESM68, -
      *.DSNTIAP.DSNTIAP)
```

The package list for the DSNESPUR plan is as follows:

```
PKLIST(location_name.PRIVATUR.DSNESM68, -
       *.DSNESPUR.DSNESM68, -
      *.DSNTIAP.DSNTIAP)
```

GUPI

Making SPUFI work with different terminal CCSIDs:

In some cases, a terminal CCSID might be different from the SPUFI CCSID. If the terminal CCSID cannot be changed to the SPUFI CCSID, consider creating additional SPUFI packages and plans that specify the terminal CCSID encoding.

About this task

You can then use the SPUFI default panels to use the plans with the special CCSID encoding instead of the DB2-supplied plans (DSNESPCS, DSNESPRR, and DSNESPUR).

Procedure

To make SPUFI work with different terminal CCSIDs, use the following example:

Suppose that when you install DB2, you create the SPUFI plans and packages using ENCODING(EBCDIC), and the default subsystem CCSID for EBCDIC data, as specified on installation panel DSNTIPF in the EBCDIC CCSID field. Most of the people in your organization use this CCSID as their terminal CCSID, so they do not encounter errors. However, a large group of DB2 application programmers code in the C language and prefer CCSID 1047 as their terminal CCSID. You can prevent this group from corrupting DB2 data and receiving errors by completing the following tasks:

1. Bind SPUFI packages and plans specifically for users who require a terminal CCSID setting of 1047.

   For example, your commands might look like the following commands:

   GUPI

   ```
   BIND PACKAGE(TIAP1047) MEMBER(DSNTIAP) -
        ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
        ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   BIND PLAN(SPCS1047) -
        PKLIST(SPCS1047.DSNESM68 -
   ```

```
            TIAP1047,DSNTIAP) -
      ISOLATION(CS) ENCODING(1047) ACTION(REPLACE)
BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
      ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
      LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPRR1047) -
      PKLIST(SPRR1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
      ISOLATION(RR) ENCODING(1047) ACTION(REPLACE)
BIND PACKAGE(SPUR1047) MEMBER(DSNESM68) -
      ACTION(REPLACE) ISOLATION(UR) ENCODING(1047) -
      LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPUR1047) -
      PKLIST(SPUR1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
      ISOLATION(UR) ENCODING(1047) ACTION(REPLACE)
```

GUPI

2. Grant access to the new packages and plans to the target user group.
3. Instruct users who use terminal CCSID 1047 to specify the following settings in the SPUFI default panels:
   - In panel DSNESP02, specify YES for CHANGE PLAN NAMES.
   - In panel DSNESP07, specify SPCS1047 in the CS ISOLATION PLAN field, SPRR1047 in the RR ISOLATION PLAN field, and SPUR1047 in the UR ISOLATION PLAN field.

   The programmers who require CCSID 1047 can now use SPUFI without receiving an error message.

Migrating DB2

The full DB2 migration process comprises three progressive catalog levels: conversion mode (from Version 8 or Version 9.1), enabling-new-function mode (from Version 8 or Version 9.1), and new-function mode.

If you need to revert from enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode, you might also use conversion mode* (from Version 8 or Version 9.1) or enabling-new-function mode* (from Version 8 or Version 9.1).

When you migrate to Version 10, you cannot use new DB2 facilities until the subsystem is in new-function mode. When the subsystem is in conversion mode (from Version 8 or Version 9.1) or enabling-new-function mode (from Version 8 or Version 9.1), you cannot use the new Version 10 facilities.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Migrating your DB2 subsystem to conversion mode (from Version 8 or Version 9.1)

The first step in migrating your DB2 subsystem is to migrate to Version 10 conversion mode from Version 8 or to Version 10 conversion mode from Version 9.1.

About this task

Attention: If you do not follow the documented procedures, unpredictable results might occur after migration to conversion mode (from Version 8 or Version 9.1).

Before you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), you must have a maintenance level on Version 8 or Version 9.1 that contains the fallback SPE, and you must start the subsystem that you are migrating at least once with the fallback SPE applied. If you do not start your Version 8 or Version 9.1 subsystem at least once with the fallback SPE applied, your Version 10 migration process will fail.

From V8 If you are migrating from Version 8, ensure that the BSDS has been converted to the new format. To convert to the new format, you can use the DSNJCNVB utility. From V8

When you start DB2, the code level of the starting DB2 is compared to the code level that is required by the current DB2 catalog. If the starting DB2 has a code-level mismatch with the catalog, DB2 does not start and a message is issued.

Refer to the DB2 Program Directory for keyword specifications that are used for Preventive Service Planning (PSP) and for information about the service levels of any coupling facilities. You can use the z/OS DISPLAY CF command to display the service level of coupling facilities. Check Information/Access or the ServiceLink facility of IBMLink for PSP information before you migrate. Also check monthly for current information about DB2.

Restriction: After migration to conversion mode (from Version 8 or Version 9.1), the administrative task scheduler is disabled until you run job DSNTIJRT.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Chapter 3, "Loading the DB2 libraries," on page 207

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

"Running the installation CLIST" on page 219

Migration step 1: Actions to complete before migration

Before you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), you must prepare your system for migration. Preparations include making adjustments for release compatibilities, running premigration queries, and completing other premigration tasks.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Run premigration queries (DSNTIJPM):

Before you migrate to DB2 Version 10, run premigration queries on the catalog of the Version 8 or Version 9.1 subsystem that you are migrating.

Procedure

To run premigration queries:

Run job DSNTIJPM.

Results

Job DSNTIJPM provides reports that identify the following conditions:

- From V8 For migration from Version 8, existence of stored procedures that specify DSNWZPR as the external module name. In Version 10, the DSNWZPR alias has been removed and you must alter stored procedures that specify DSNWZPR as the external module name to use the external module DSNWZP instead. From V8

- Existence of the sample database from the previous release. This sample database is required to verify migration to Version 10 conversion mode (from Version 8 or Version 9.1).
- Existence of user-defined DB2 catalog indexes that reside on user-managed storage. Before you migrate to enabling-new-function mode (from Version 8 or Version 9.1), you need to define a shadow data set for each of these indexes. DB2 uses the shadow data set during job DSNTIJEN when it runs REORG to change the catalog table spaces to DB2 Version 10 new-function mode.
- Existence of user-defined DB2 catalog indexes on the DB2 catalog that reside on DB2-managed storage. Before you migrate to enabling-new-function mode (from Version 8 or Version 9.1), you need to review the current space allocations of the data sets for these indexes, and increase the space for any that are approaching capacity.
- Existence of stored procedures that are defined to run in the DB2 stored procedures address space, which is unsupported in Version 10. Before you begin migration, review any such procedures and either drop them or convert them to run in a WLM-established address space.
- Existence of packages and plans that were bound prior to DB2 Version 7. You need to check the timestamp of each of these identified packages or plans to determine whether it was last bound prior to DB2 Version 6. Such a package or plan must be rebound for use on DB2 Version 10. If the ABIND subsystem parameter is set to YES or COEXIST, DB2 will automatically rebind the package or plan after migration. Otherwise, you need to rebind it manually to prevent SQLCODE -908 errors.
- Existence of authorization IDs and roles with the EXECUTE privilege or the EXECUTE WITH GRANT OPTION privilege for one or more DB2 database metadata stored procedures. These stored procedures are re-created during migration to Version 10 conversion mode (from Version 8 or Version 9.1). After they are re-created, you need to grant privileges for the new versions of the stored procedures to the identified authorization IDs and roles.
- Existence of incomplete table definitions. You need to complete or drop all such tables before you begin migration because their behavior will be different in Version 10.
- Existence of incomplete column definitions. You need to complete or drop all such columns or drop the tables before you begin migration because their behavior will be different in Version 10.
- Existence of schemas that have any EBCDIC EXPLAIN tables.
- Existence of EXPLAIN tables that are not in the current release format.
- Existence of DB2 EXPLAIN tables named DSN_PTASK_TABLE that have one or more column names containing a variant EBCDIC character such as the number (#) symbol (code point x'7b' in code page 37). Such tables need to be corrected before you begin migration because they are not supported in Version 10.
- Existence of plans that contain DBRMs, meaning that they were created using the MEMBER option of the BIND PLAN command. Such plans are not supported in Version 10, and will be disallowed in a future release. To convert these plans, use the REBIND PLAN statement with the COLLID option. You also need to update the BIND statements that are used to create these plans to use packages.
- Existence of plans that were bound with the ACQUIRE(ALLOCATE) option. Support for ACQUIRE(ALLOCATE) is removed in Version 10, and the syntax may be disallowed in a future release. Use REBIND PLAN with ACQUIRE(USE) to change this option in existing plans.

- Existence of packages that were bound with parallelism. Such queries will be incrementally rebound each time they are executed on DB2 Version 10. After you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), you can monitor incremental rebinds on the DB2 accounting and statistic reports and issue explicit rebinds for packages reported on IFCID 360 to avoid the cost of incremental rebinds at execution time.
- Existence of packages and plans that were bound with the DBPROTOCOL(PRIVATE) BIND option. You can convert these packages and plans to DRDA protocol by issuing the REBIND command against the objects that are indicated in the query output. Or, you can run the private protocol to DRDA protocol (REXX) tool (DSNTP2DP) to obtain the commands to convert the packages and plans to DRDA protocol.
- Existence of obsolete objects that can be dropped after migration to new-function mode.
- Existence of package names and collection IDs that contain special characters that might cause package-not-found errors. The errors might occur when those packages are accessed from remote client drivers that support the sending of DRDA character type parameter data in UTF-8 Unicode. Those packages need to be bound with the ACTION(REPLACE) option from the remote client drivers. The following sample shows the output from a successful run of the report application.

```
READY
  DSNTPMQ V91B SYSADM
DSNTPMQ entry:
  Subsystem ID ..............................................  V91B
  Authorization ID ......................................... SYSADM
  .
  .
  .
  Report 25 completed
READY
END
.
.
.
=====================================================================
= The following collection IDs and package names were bound by a       =
= remote client system. After migrating to DB2 V10, when a remote      =
= client system upgrades to a version that supports the flow of        =
= Unicode (UTF-8) DRDA character type data, these collection IDs and   =
= package names will not be found by DB2 V10. Therefore, the remote    =
= remote client system needs to bind and replace these packages after  =
= it upgrades to a version that supports the flow of Unicode (UTF-8)   =
= DRDA character type data to a DB2 V10 system.                        =
=====================================================================
COLLID               NAME
-------------------- ----------
COLL@2               PROG1
BIGC#LL@1            PROG2
COLL123              PROG@@
```

- Existence of authorization IDs and roles that have the EXECUTE or EXECUTE WITH GRANT privilege for the DSNLEUSR stored procedure. This stored procedure is dropped and re-created during migration to new-function mode. After migration to new-function mode, these privileges must be granted again to these authorization IDs and roles.
- Existence of catalog tables that will be affected by the disabling of DATA CAPTURE CHANGES during migration to Version 10 conversion mode (from Version 8 or Version 9.1).

- Existence of catalog tables that will be affected by the disabling of DATA CAPTURE CHANGES during enabling-new-function mode (from Version 8 or Version 9.1) processing.
- Existence of packages that will be affected by the additional authorization checking for UPDATE and DELETE statements.
- AMI-based MQ functions in the DB2MQ1C, DB2MQ2C, DB2MQ1N, and DB2MQ2N schemas that are no longer supported. Before migrating to Version 10, identify applications that use these functions and modify them to use the MQI-based replacement functions.
- MQSeries XML routines that are no longer supported. Before migrating to Version 10, identify and modify applications that use these functions.
- Existence of more than one row in the SYSIBM.SYSTABAUTH catalog table for a table owner, table name, and timestamp combination. Only one row should exist for each table owner, table name, and timestamp combination.
- From V9.1 For migration from Version 9.1, existence of all packages that reference the DSN_XMLVALIDATE user-defined function. After migration to conversion mode from Version 9.1, these packages should be rebound to reference the DSN_XMLVALIDATE built-in function instead. From V9.1

Related concepts:

"WLM management of stored procedures" on page 839

Related tasks:

From V8 "Change stored procedures that specify DSNWZPR as the external module name for migration from Version 8" on page 579 From V8

"Converting EXPLAIN tables (before migration)" on page 64

From V9.1 Moving from SYSFUN.DSN_XMLVALIDATE to SYSIBM.DSN_XMLVALIDATE (DB2 Programming for XML) From V9.1

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Related reference:

"The private to DRDA protocol REXX tool (DSNTP2DP)" on page 879

Make adjustments for release incompatibilities:

Some changes in Version 10 might affect your DB2 operations after migrating.

Procedure

To prepare for release incompatibilities:

Plan for and make adjustments according to the incompatibilities that are listed in "Changes that might affect your migration from Version 8" on page 65 or "Changes that might affect your migration from Version 9.1" on page 114.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Define the SMS environment for the DB2 catalog and directory data sets (DSNTIJSS):

In DB2 Version 10, data sets that are defined for the DB2 catalog and directory are managed by DB2 and must be SMS-managed.

About this task

You are not required to convert existing DB2 data sets to an SMS environment before migrating to Version 10. These data sets can indefinitely remain non-SMS-managed, but they will be converted to SMS management when the related table space is reorganized.

The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. To define a data class with this attribute, specify EXT in the DATA SET NAME TYPE field of the DATA SET CLASS DEFINE panel of ISMF. Then, ensure that the automatic class selection (ACS) routine associates the DB2 catalog and directory data sets with this data class.

Procedure

To create the stand-alone SMS environment for the DB2 catalog and directory:

1. Customize job DSNTIJSS according to the directions in the job prolog.
2. Run job DSNTIJSS.
3. To activate the SMS environment, use this z/OS command, where *scds-name* is the name of the SMS source control data set that was specified by DSNTIJSS:

   ```
   SETSMS SCDS(scds-name)
   ```

   Attention: This command will deactivate any existing SMS environment that is defined from another SCDS.
4. Run the SMS CONVERTV tool with the TEST option to verify that all data sets on target volumes can be placed under SMS management.

   Example of running CONVERTV with the TEST option:

   ```
   //CONVRTV EXEC PGM=ADRDSSU,REGION=5M
   //SYSPRINT DD SYSOUT=*
   //DD1      DD UNIT=SYSALLDA,VOL=SER=targvol1,DISP=OLD
   //DD2      DD UNIT=SYSALLDA,VOL=SER=targvol2,DISP=OLD
   //SYSIN    DD *
     CONVERTV DDNAME(DD1) SMS TEST CATALOG
     CONVERTV DDNAME(DD2) SMS TEST CATALOG
   /*
   ```
5. When CONVERTV TEST reports that all data sets on the target volumes can be placed under SMS management, run CONVERTV without the TEST option to convert the data sets. If any data sets cannot be converted, do not run

CONVERTV without the TEST option. You must either create additional ACS routines to manage these data sets or move them to different (non-SMS) volumes.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Save critical access paths:

Sometimes changes between releases of DB2 cause unwanted access path changes.

Procedure

PSPI To save your critical access paths:

1. Consult with your performance analysts to determine which queries are especially critical and ensure that you have a PLAN_TABLE that contains the good access path. You can save static SQL statements in the access path repository by specifying a basic or extended plan management policy.
2. You should run EXPLAIN on your queries before migrating, but because EXPLAIN requires a rebind, your access paths might change. Therefore, extract the needed queries and then run EXPLAIN on them under a different application or program name. This action protects the existing application while the access path information is obtained.
3. After the access paths for the extracted queries are validated, update the APPLNAME or PROGNAME columns of the PLAN_TABLE to the correct name. PSPI

Related concepts:

Plan management polices (DB2 Performance)

Related tasks:

Capturing EXPLAIN information (DB2 Performance)

Saving access path information for static SQL statements (DB2 Performance)

Influencing access path selection by using optimization hints (DB2 Performance)

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Related reference:

EXPLAIN tables (DB2 Performance)

EXPLAIN (DB2 SQL)

Examine all new and changed values for DB2I panels:

During a migration, the DB2I default panels DSNEOP01 and DSNEOP02 are not initialized with the values that are specified during the installation CLIST process. The DB2I panel variables in the ISPF profile from the previous release are used on the current release. Any customized DSNEPROF members are migrated from your previous release to Version 10.

Procedure

To ensure that your customized values are still valid:

Examine any new or changed DB2I default panel values.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Ensure that sample objects from your previous release are available:

If you no longer have the IVP jobs from your previous release, you need to run the previous release's installation CLIST to regenerate them.

Procedure

To ensure that sample objects from your previous release are available:

If you dropped your previous release's sample database (by running job DSNTEJ0), run the previous release's IVP jobs through phase 3 before you start the migration to conversion mode (from Version 8 or Version 9.1).
If you do not have the previous release's jobs available during migration, you will not have a DB2-supported IVP to verify a successful migration to conversion mode (from Version 8 or Version 9.1).

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

Make a backup copy of the BSDS for migration from Version 8:

Because the BSDS is converted in job DSNTIJUZ if you did not convert it in DB2 Version 8, you should make a backup copy of the BSDS before migrating to Version 10 conversion mode from Version 8.

About this task

In Version 10, the BSDS must be in the format that supports up to 10,000 data sets per copy for archive logs and 93 data sets per copy for active logs.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

Change stored procedures that specify DSNWZPR as the external module name for migration from Version 8:

In DB2 Version 8, DSNWZPR was an alias of module DSNWZP. In Version 10, the DSNWZPR alias has been removed.

Procedure

To prepare your Version 8 stored procedures:

Change stored procedures that specify DSNWZPR as the external module name to specify DSNWZP.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

Ensure that no utility jobs are running:

In Version 10, you can restart or terminate a utility only on the same release on which it was started.

About this task

Any outstanding utilities prior to Version 10 cannot be restarted or terminated after you have migrated to Version 10 conversion mode (from Version 8 or Version 9.1).

Immediately before you begin the migration process, ensure that you do not have outstanding utility jobs.

Procedure

To ensure that you do not have outstanding utility jobs:

Issue the DISPLAY UTILITY(*) command.

Related tasks:

From V8 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 14 From V8

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Premigration checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 24 From V9.1

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

-DISPLAY UTILITY (DB2) (DB2 Commands)

Migration step 2: Run the link checker on your DB2 table spaces (optional)

The link checker utility, DSN1CHKR, verifies the integrity of the DB2 directory and catalog table spaces on the system that you are migrating.

Before you begin

Recommendation: Run DSN1COPY and DSN1CHKR with the catalog and the directory table spaces stopped, or with DB2 stopped.

Procedure

To verify the integrity of your DB2 directory and catalog table spaces:

1. GUPI Issue the STOP DATABASE command on catalog and directory table spaces that contain links or hashes. These table spaces include:
   - DSNDB06.SYSDBASE
   - DSNDB06.SYSDBAUT
   - DSNDB06.SYSGROUP
   - DSNDB06.SYSPLAN
   - DSNDB06.SYSVIEWS
   - DSNDB01.DBD01

   GUPI
2. Run DSN1CHKR on these table spaces. DSN1CHKR scans the specified table space for broken links, hash chains, and orphans (records that are not part of any link or chain).
3. Run the DSN1COPY utility with the CHECK option on all catalog table spaces to ensure that the table space pages are physically correct and that the catalog table spaces are clustered.

   When you run this utility on segmented table spaces, you might receive message DSN1985I. The segmented table spaces in the catalog and directory are: DSNDB06.SYSPKAGE, DSNDB06.SYSSTR, DSNDB06.SYSSTATS, DSNDB06.SYSDDF, DSNDB06.SYSOBJ, DSNDB01.SYSUTILX, and DSNDB01.SPT01. You can ignore this message.
4. Run the CHECK INDEX utility.
5. You should run the following query on the catalog tables of the Version 8 or Version 9.1 system that you are migrating. If the query returns any rows, the

identified STOGROUPs have both specific and non-specific volume IDs. Table spaces in databases that use these STOGROUPs cannot be image copied or recovered until ALTER STOGROUP is used to remove volumes so that the STOGROUP has either specific or non-specific volume IDs.

This query is commented out in the Version 10 member DSNTESQ of *prefix*.SDSNSAMP.

GUPI

```
SELECT * FROM SYSIBM.SYSVOLUMES V1
         WHERE VOLID ¬= '*' AND
               EXISTS (SELECT * FROM SYSIBM.SYSVOLUMES V2
                                WHERE V1.SGNAME = V2.SGNAME AND
                                      V2.VOLID = '*')
```

GUPI

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

CHECK INDEX (DB2 Utilities)

DSN1CHKR (DB2 Utilities)

DSN1COPY (DB2 Utilities)

Related information:

DSN1985I (DB2 Messages)

Migration step 3: Determine which packages are invalid after migration (optional)

Migrating to Version 10 conversion mode (from Version 8 or Version 9.1) renders some packages invalid.

Procedure

To identify which packages are invalid after migration to Version 10:

Run the following query on the Version 8 or Version 9.1 subsystem that you are migrating. This query is commented out in the Version 10 member DSNTESQ of *prefix*.SDSNSAMP. You can run this query from SPUFI or from a dynamic SQL program like DSNTEP2.

PSPI

```
SELECT DISTINCT COLLID, NAME, VERSION
  FROM SYSIBM.SYSPACKDEP, SYSIBM.SYSPACKAGE
  WHERE BNAME IN
('SYSPACKAGE'   ,'SYSPLAN'        ,'SYSTABLEPART'   ,'SYSROUTINES' ,
 'DSNOFX01'     ,'DSNOFX02'       ,'DSNOFX07'       ,'DSNOPX01'    ,
 'DSNATX01'     ,'DSNADH01'       ,'DSNADX01'       ,'DSNAGH01'    ,
 'DSNAGX01'     ,'DSNOAX01'       ,'DSNSKX01'       ,'DSNSKX02'    ,
 'DSNKAX01'     ,'DSNKAX02'       ,'DSNAPH01'       ,'DSNAPX01'    ,
 'DSNWCX02'     ,'DSNWCX03'       ,'DSNAUH01'       ,'DSNAUX02'
 )
    AND LOCATION = ' '
```

```
    AND BQUALIFIER = 'SYSIBM'
    AND BTYPE IN ('I','T')
    AND COLLID = DCOLLID
    AND NAME = DNAME
    AND CONTOKEN = DCONTOKEN
  ORDER BY COLLID, NAME, VERSION;
```

PSPI

After migration, you can explicitly rebind these packages or let DB2 rebind them automatically.

Related concepts:

Changes that invalidate packages (DB2 Application programming and SQL)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 4: Check for consistency between catalog tables (optional)

The DSNTESQ queries check the logical correctness of the DB2 catalog.

Before you begin

Before you run these queries, you should have completed migration step 2 to run the DSN1CHKR utility, the CHECK INDEX utility, and the DSN1COPY utility with the CHECK option.

Procedure

To check for consistency between catalog tables:

Run the queries that are not commented out in member DSNTESQ of the *prefix*.SDSNSAMP library. You can execute the SQL statements in DSNTESQ from SPUFI or from a dynamic SQL program like DSNTEP2.
You can run the queries on the actual catalog tables or on "mirror" copies of the catalog tables. If you run the queries on the copies, use the comment lines in member DSNTESQ for guidance. By running queries on copies of the catalog table, you reduce contention on the catalog.

Related tasks:

"Migration step 2: Run the link checker on your DB2 table spaces (optional)" on page 580

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 5: Take image copies of the directory and catalog: DSNTIJIC

For backup purposes, create a copy of the catalog and directory of the system that you are migrating.

About this task

Attention: If you do not create this copy, DB2 can start. However, if errors in the catalog or directory require you to fall back to your previous version, you risk losing some of your tables and data.

Procedure

To create an image copy of the directory and catalog:

1. Examine job DSNTIJIC for the following items, and modify as needed:
   - The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also used as the unit name for the default archive log data sets.
   - Expiration date or retention period. You can add a retention period or an expiration date to the job.
   - The user on the JOB statement. Ensure that the user is authorized. If DB2 is started under ACCESS(MAINT), this must be the same user that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1.
2. Complete one of the following tasks:
   - From V8 If you are migrating from Version 8, run the Version 8 DSNTIJIC job. From V8
   - From V9.1 If you are migrating from Version 9.1, run the Version 9.1 DSNTIJIC job. From V9.1

   Job DSNTIJIC contains a list of all the DB2 directory and catalog table spaces. When you run DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. The copied table spaces allow you to recover the DB2 catalog and directory in case of a failure.
3. If job DSNTIJIC fails or abends, look for problems with the tape that is set up for image copy. If you do not find a problem, examine the log for problems. For example, look for incorrect size or I/O errors.

What to do next

After migration, periodically run the Version 10 job DSNTIJIC against the Version 10 directory and catalog, perhaps daily or weekly. This action reduces the amount

of time that is required for recovering the DB2 directory or catalog. The copied data and log data sets are needed for recovery.

If you are remigrating, you need to complete one of the following actions:

- Change the names of the data sets in which the new image copies are to reside. (Migration image copies use the current data set names.)
- Run the MODIFY utility to remove the migration image copies. If you select this option, ensure that you are familiar with the MODIFY utility.

If DSNTIJIC has been modified to copy table spaces to disk instead of tape, the job is limited to two disk volumes. To change the number of disk volumes, the job needs to be modified again, using volume serial numbers instead of VOL=REF=*.*jobstep*.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

MODIFY RECOVERY (DB2 Utilities)

Migration step 6: Connect DB2 to TSO

Access to TSO is required to support the interactive component of DB2 (DB2I). Access to TSO also allows batch applications to access DB2 when those batch programs are executed under the TSO terminal monitor program (TMP).

Procedure

To attach DB2 to TSO:

1. Make DB2 load modules available to TSO and batch users.
2. Make DB2 CLISTs available to TSO and batch users.
3. Make panels, messages, and load modules available to ISPF and TSO.

What to do next

Save your TSO logon procedures and JCL from your Version 8 or Version 9.1 subsystem in case you need to fall back from DB2 Version 10.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Make DB2 load modules available to TSO and batch users:

To connect DB2 to TSO, you must first make DB2 load modules available to TSO and batch users. If *prefix*.SDSNEXIT and *prefix*.SDSNLOAD are not included in your LNKLST*xx*, you can skip this step.

Procedure

To make DB2 load modules available to TSO and batch users:

If you have not included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, add JOBLIB or STEPLIB statements to your logon procedures and JCL to ensure that you access the Version 10 load modules. Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in your JOBLIB or STEPLIB concatenations.

What to do next

You can attach to multiple releases of DB2 with your existing TSO or CAF logon procedures without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you must update those procedures and jobs to point to the latest level of DB2 load libraries.

Make DB2 CLISTs available to TSO and batch users: DSNTIJVC:

Job DSNTIJVC merges the tailored CLISTs from *prefix*.NEW.SDSNTEMP with unchanged CLISTs from *prefix*.SDSNCLST, and it places all CLISTs in *prefix*.NEW.SDSNCLST.

About this task

Tailoring changes can modify these CLISTs: DSNEMC01, DSNH, DSNU, and DSNHC. The DSNTINST CLIST reads these CLISTs from *prefix*.SDSNCLST, edits them, and places them in *prefix*.NEW.SDSNTEMP. You can modify the default values.

The DSNEMC01 CLIST uses the values that are specified on installation panel DSNTIPF and stores the results in the ISPF profile member DSNEPROF. You can migrate customized DSNEPROF members from Version 8 to Version 10 conversion mode from Version 8 or from Version 9.1 to Version 10 conversion mode from Version 9.1. However, you need to examine any new or changed default panel values to ensure that your customized values are still valid.

DSNTIJVC also converts the DB2 CLISTs from a fixed-block record format to a variable-block format, with a record length of 84 and a block size of 3120. A CLIST that has been converted from fixed-block format to variable-block format cannot be used as input to the DSNTINST CLIST; use the unedited version of the SDSNCLST data set, as created by SMP/E.

Procedure

To make the CLISTs available to TSO and batch users:

1. If you use fixed-block CLIST libraries, modify the DSNTIJVC job as follows:
   - Change the SYSIN DD to DUMMY.
   - Change the allocation of *prefix*.SDSNCLST to match the data control block (DCB) attributes of your other CLIST libraries.
2. Run job DSNTIJVC.
3. Either concatenate *prefix*.NEW.SDSNCLST with your existing CLIST libraries or copy *prefix*.NEW.SDSNCLST into an existing CLIST library.

What to do next

If you need to rerun job DSNTIJVC, first delete data set *prefix*.NEW.SDSNCLST, which is created by this job.

When corrective service is applied to a CLIST, SMP/E changes only the *prefix*.SDSNCLST data set. You need to redo any record format changes and reapply any special tailoring that is required. You also need to move the CLIST to *prefix*.NEW.SDSNCLST. Corrective service (program temporary fixes) for these CLISTs is sent with ++HOLD statements, calling your attention to the possibility of additional work.

Related concepts:

"Installation jobs that the CLIST tailors" on page 497

Make DB2I panels, messages, and load modules available to ISPF and TSO:

During migration, to connect the DB2 Interactive facility (DB2I) to TSO, you must make panels, messages, and load modules available to ISPF and TSO.

About this task

DB2I uses the ISPF PROFILE and SHARED variable pools for most panel variable fields. As a result, you can easily re-enter a panel when panel variables have previously been specified. For the DB2 subcommands that permit lists of plan names, package names, DBRMs, and ENABLE and DISABLE statements, DB2I provides ISPF to contain all the user-specified variables for these subcommand keywords.

DB2I creates and maintains a set of ISPF tables in a user-defined TSO data set that is allocated to a data set with a ddname of DSNETBLS. The DB2I-generated tables in this library are DSNCONNS, DSNDBRMS, and DSNPLPKN. The following table shows the library table member names and their contents.

Table 86. The DB2 ISPF table library

DSNCONNS	ENABLE or DISABLE connection type and connection name variables that are referenced by plan or package name
DSNDBRMS	Subcommand DBRM member and LIBRARY name variables that are referenced by plan name
DSNPLPKN	Package list variables that are referenced by package name
DSNPATHS	Schema name variables that are referenced by plan or package name and that are to be included in the PATH keyword

DB2I uses ISPF table services to maintain individual ISPF tables within the DSNETBLS data set. For performance reasons, ISPF keeps this table library in an open state once an individual table has been updated. Attempts to close this data set by using the TSO FREE command results in error message IKJ56861I.

Procedure

To make panels, messages, and load modules available to ISPF and TSO:

1. If you added the DB2I libraries as part of the step to prepare for running the DB2 installation CLIST, you can skip this step. Otherwise, concatenate the DB2 ISPF libraries with the ISPPLIB, ISPTLIB, ISPSLIB, and ISPMLIB DD statements in your logon procedures. You must also concatenate the DB2 ISPF libraries

with these DD statements in any of your CLISTs where the libraries might be allocated. These libraries are *prefix*.SDSNSPFP, *prefix*.SDSNSPFM, *prefix*.SDSNSPFS, and *prefix*.SDSNPFPE or *prefix*.SDSNPFP. If you use English DB2I panels, concatenate SDSNPFPE to ISPPLIB. If you use Kanji DB2I panels, concatenate SDSNPFPK instead of SDSNPFPE.

```
PROFILE WTP MSGID
ALLOCATE DDNAME(SYSEXEC) DSN('prefix.SDSNCLST') SHR REUSE
ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
    'ISP.SISPMENU') SHR REUSE
ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' 'prefix.SDSNPFPE' +
    'ISP.SISPPENU') SHR REUSE
ALLOCATE DDNAME(ISPSLIB) DSN('prefix.SDSNSPFS' +
    'ISP.SISPSLIB' 'ISP.SISPSENU') SHR REUSE
```

2. Create the DSNETBLS data set.

   When allocating this data set, you should assign the following DCB attributes, where *n* is any integer:

   ```
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(n*LRECL)
   ```

   The following example shows how to set up an ALLOCATE statement to create the data set:

   ```
   ALLOC DA(DSNSPFT) NEW SP(1 1) TR DIR(10) +
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(800)
   F(DSNETBLS) REUSE
   ```

   The following example shows how to allocate an existing data set to the data set with the DSNETBLS ddname:

   ```
   ALLOC DA(DSNSPFT)      F(DSNETBLS) REUSE
   ```

3. If you want to run the ISPF/CAF sample application that is provided with DB2, ensure that the data set *prefix*.RUNLIB.LOAD is included in the logon procedures or in the ISPLLIB concatenation. If you chose IBMCOB for your LANGUAGE DEFAULT on panel DSNTIPF, you should also include the data set *prefix*.CEE.SCEERUN.

Related concepts:

"Dynamic SQL and the ISPF/CAF application" on page 766

Related tasks:

"Make DB2 load modules available to TSO and batch users" on page 584

Related reference:

z/OS ISPF Messages and Codes

Migration step 7: Connect DB2 to IMS (optional)

Connecting DB2 to IMS is an optional part of migration. Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

1. Make DB2 load modules available to IMS.
2. Define DB2 to IMS.
3. Define new programs and transactions to IMS.
4. Prepare IMS applications for DB2.
5. Depending on your site, you might also need to:
   - Define DB2 plans for IMS applications.
   - Generate a user language interface.

Related tasks:

"Connecting the IMS attachment facility" on page 869

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 8: Connect CICS to DB2 (optional)

Connecting DB2 to CICS requires that you regenerate several CICS tables with additional entries.

About this task

A macro is supplied with CICS to define the connection between CICS and DB2 by using a resource control table (RCT).

Ensure that you coordinate the attachment facility connection with your CICS support group.

Procedure

To connect the CICS attachment facility:

1. Recalculate space requirements for the CICS attachment facility.
2. Define your CICS attachment facility parameters using the RCT.
3. Update the CICS system tables.
4. Update the CICS initialization JCL.
5. Coordinate DB2 and CICS security if necessary.
6. Prepare new CICS applications for DB2 if necessary.

Related tasks:

Defining the CICS DB2 connection

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 9: Stop DB2 Version 8 or Version 9.1 activity

Before making DB2 Version 10 conversion mode (from Version 8 or Version 9.1) operational, ensure that all work is stopped on the Version 8 or Version 9.1 subsystem that you are migrating.

Before you begin

If you do not stop work on the Version 8 or Version 9.1 subsystem or data sharing member that you are migrating, fallback procedures might fail.

Procedure

To stop work on the system that you are migrating:

1. Issue the following command, where *-DSN1* is the subsystem command prefix that is defined for DB2. You must have system administrator (SYSADM) or system operator (SYSOPR) authority to issue this command.

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI The QUIESCE keyword allows DB2 to complete the processing of programs that are currently executing. This might require some processing time.
2. Issue the following command to allow only the system administrators and system operators that were defined during installation to access DB2:

   GUPI

   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```

   GUPI If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. For example, if the bootstrap data set (BSDS) does not match the subsystem parameter values, ensure that the correct jobs were run for DSNTIJUZ. Ensure that you started DB2 with the correct subsystem parameter option.
3. Ensure that all work is complete.
   - Make sure no units of recovery remain. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```

     GUPI Then use -DSN1 RECOVER INDOUBT for any indoubt threads.
   - All utilities must be restarted or terminated on the version on which they were started. If you do not use data sharing, ensure that all utilities are completed or terminated on your Version 8 or Version 9.1 subsystem. To ensure that no utility work remains, issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY UTILITY(*)
     ```

     GUPI

     After you have determined the utilities that are running, you can let them complete processing or you can terminate the utility. To stop all utilities, issue the following command:

     ```
     -DSN1 TERM UTILITY(*)
     ```
   - Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. To determine existing restrictions, issue the following commands:

     GUPI

     ```
     -DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
     -DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
     ```

     GUPI

You must have system administrator or system operator authority (specified on installation panel DSNTIPP1) to issue this command.

Recover any table spaces and index spaces that have write error ranges or deferred restart states by using the RECOVER utility.

4. To stop DB2, issue the following command:

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI

   A user with system administrator (SYSADM) or system operator (SYSOPR) authority must enter this command.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

-DISPLAY DATABASE (DB2) (DB2 Commands)

-DISPLAY THREAD (DB2) (DB2 Commands)

-DISPLAY UTILITY (DB2) (DB2 Commands)

-STOP DB2 (DB2) (DB2 Commands)

-TERM UTILITY (DB2) (DB2 Commands)

Migration step 10: Back up your DB2 Version 8 or Version 9.1 volumes (optional)

After you have stopped Version 8 or Version 9.1 activity, you should back up the Version 8 or Version 9.1 subsystem that you are migrating. This backup provides a recovery base in case one becomes necessary.

Procedure

To back up your subsystem:

Take dumps of the DB2 subsystem data sets. You also can take dumps of the SMP/E data sets and the DB2 distribution and target libraries.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 11: Define DB2 initialization parameters: DSNTIJUZ

Job DSNTIJUZ generates the DB2 data-only subsystem parameter module DSNZP*xxx*, the application defaults load module (*dsnhdecp*), and the data-only load module DSNHMCID.

About this task

The subsystem parameter module consists of the expansion of six macros that contain the DB2 execution-time parameters that you selected by using the ISPF panels. The names of these macros are DSN6ARVP, DSN6FAC, DSN6GRP, DSN6LOGP, DSN6SPRM, and DSN6SYSP.

Procedure

To define the DB2 initialization parameters:

1. Save your Version 8 or Version 9.1 subsystem parameter module so that it is available in case you need to fall back.
2. Run job DSNTIJUZ.

Results

The DSNTINST CLIST performs calculations by using the values that you specified for some of the parameter values that you entered on the panels. These calculations appear in the macro descriptions.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

DSNTIJUZ actions:

Job DSNTIJUZ defines the DB2 data-only subsystem parameter module DSNZP*xxx*, the application defaults load module, and the data-only load module DSNHMCID.

Job DSNTIJUZ also completes the following actions:

- From V8 For migration from Version 8, runs the DSNJCNVB module to convert the BSDS if you did not convert the BSDS in your previous release. The updated format supports up to 10,000 data sets per copy for archive logs and 93 data sets per copy for active logs. From V8
- Assembles and link-edits the application defaults module, *dsnhdecp*, into the *prefix*.SDSNEXIT library.
- Assembles and link-edits the subsystem parameter module, DSNZP*xxx*, into the *prefix*.SDSNEXIT library.
- Assembles and link-edits the DSNHMCID data-only module, which is needed for message conversion by DB2 applications and utilities. DSNHMCID is link-edited into both the *prefix*.SDSNEXIT and *prefix*.SDSNLOAD libraries.
- Uses SMP/E in step DSNTIMQ to read in the edited version of DSNTIJUZ and set up GENASM entries for the DB2 application defaults macro and system parameter (DSN6*xxxx*) macro. This action causes service to any of these macros to trigger SMP/E to initiate a reassembly of the DB2 system parameter module.

  This job step is optional and commented out. If you want to run it, you need to uncomment it and run it once to feed the appropriate includes and library names to SMP/E.

If DSNHMCID cannot exist in *prefix*.SDSNLOAD, take one of the following actions:

- Include *prefix*.SDSNEXIT ahead of *prefix*.SDSNLOAD in the system link list.
- Include *prefix*.SDSNEXIT ahead of *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statements for all DB2 applications, address space start-up procedures, TSO log-on procedures, CICS tasks, and IMS tasks that use DB2.

Additional steps for DSNTIJUZ are as follows:

1. If you added a STEPLIB DD statement to the DB2 start procedures ahead of *prefix*.SDSNEXIT and *prefix*.SDSNLOAD, you can move the SYSLMOD output to that library.
2. If you changed the prefix for the DB2 distribution libraries, edit DSNTIJUZ to correct the data set names.
3. If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBAA10, edit DSNTIJUZ so that the SMP/E temporary data set (SMPTLIB) is included in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. This action ensures that member DSNARIB is linked with the application defaults load module. SMPTLIB is *hlq*.HDBAA10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

   ```
   SET   BOUNDARY (GLOBAL).
   LIST  DDDEF ( SMPTLIB ).
   ```

   Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD. The linkage editor issues a return code of 8, along with message IEW0342 for the following CSECTs:

DSNFSYSP	DSNJARVP	DSNJLOGP	DSNTSPRM
DSNVDIR1	DSNZMSTR	DSN3DIR1	

Related reference:

"Subsystem parameters that are not on installation panels" on page 253

Add a second BSDS:

If the subsystem that you are migrating has only one BSDS, you should add another BSDS to DB2 before you migrate.

About this task

Recommendation: Add a second BSDS because having two BSDSs makes recovery much easier in most situations. In cases that normally require recovery and restart, a second BSDS allows you to continue working. Also, the required storage is small, and the data set is relatively inactive.

If you do not add another BSDS, manually add TWOBSDS=NO in the DSN6LOGP macro in job DSNTIJUZ.

Procedure

To add a second BSDS:

1. Change your subsystem parameter to TWOBSDS=YES by using job DSNTIJUZ.
2. Define a second BSDS; use the VSAM BSDS definition in job DSNTIJIN as an example.
3. Add a //BSDS2 DD statement to the DSN1MSTR DB2 startup procedure.

4. Execute the RECOVER BSDS command to establish dual BSDS.

Results

You might receive message GIM65001W when you run steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when you runstep DSNTIMQ. You can ignore these messages.

If DSNTIJUZ fails or abends, correct the problem and rerun the job, using the same subsystem parameter name.

Related reference:

-RECOVER BSDS (DB2) (DB2 Commands)

Related information:

Using Access Method Services (DFSMS Access Method Services for Catalogs)

Migration step 12: Establish subsystem security (optional)

DB2 lets you control access to data within DB2. DB2 also works together with outside security systems, such as RACF, that control access to the DB2 subsystem.

About this task

Because your Version 10 subsystem reuses the data objects from the Version 8 or Version 9.1 subsystem that you are migrating, you have probably already supplied the protection that those objects need. However, you probably want to protect the new (Version 10) DB2 data objects.

Procedure

To establish subsystem security:

1. Run job DSNTIJRA. Job DSNTIJRA defines the administrative task scheduler started task module to RACF program control and defines the administrative task scheduler as a trusted context in RACF. DSNTIJRA is intended for the site security administrator, and can be run any time prior to installing or migrating to Version 10.
2. If you have a DB2 data sharing environment, customize and run DSNTIJRA for each member of the group.

Related concepts:

Getting started with DB2 security (Managing Security)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 13: Define DB2 Version 10 to z/OS: DSNTIJMV

Job DSNTIJMV completes some of the steps that are required to identify DB2 to z/OS, including updating members of SYS1.PARMLIB and SYS1.PROCLIB.

Before you begin

Ensure that you completed the task in "Make DB2 load modules available to TSO and batch users" on page 584.

About this task

DSNTIJMV renames your Version 8 or Version 9.1 procedures so that they do not conflict with Version 10 procedures, which replace your old procedures. The job also creates Language Environment options data sets for the DB2 core WLM environment for Java routines.

Procedure

To define DB2 Version 10 to z/OS:

1. Run job DSNTIJMV. If job DSNTIJMV runs successfully, it produces return codes of 0. Because a rename can fail without setting the return code, verify all renames.
2. Review the content of the Language Environment options data set and adjust the following environment variables as needed. The data set is created by DSNTIJMV and must be allocated by the JAVAENV DD of the address space procedure for the DB2 core WLM environment for Java routines.
   - _CEE_ENVFILE, if present, needs to be set to the name and path of the environment variable file that was created by job step DSNTIJR.
   - JVMPROPS needs to be set to the name and path of the JVM startup options file that was created by job step DSNTIJS.
   - DB2_BASE and JCC_HOME need to be set to the names of the HFS directories that you selected for the DB2 for z/OS and JCC product classes.
3. From V9.1 If you are migrating from Version 9.1, review the content of the environment variable file that was created by job step DSNTIJR. Adjust the following information as needed:
   - CLASSPATH needs to specify the path for the Java class that contains the JAR files for z/OS XML System Services XML schema registration (gxljapi.jar and gxljosrglmpl.jar).
   - LIBPATH needs to specify the path for the native library for z/OS XML System Services (libGXLNVBAT.so).
   - STEPLIB needs to specify the name of the DB2 for z/OS runtime library.

   From V9.1

What to do next

Because different sites have different requirements for identifying DB2 to z/OS, DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes in SYS1.PARMLIB and SYS1.PROCLIB are incomplete. You might have additional procedures of your own to rename, or you might have to provide procedures for both releases, using alias names to indicate the current release. If you changed the DB2 subsystem name, the names of the DB2 address space startup procedures also change. If you made any changes to your Version 8 or Version 9.1 procedures (such as data set names), make similar changes to the Version 10 procedures. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB or by editing DSNTIJMV.

Recommendation: For SYS1.PROCLIB, submit the procedure-update section of DSNTIJMV as necessary. However, before you make the updates, read the information about DSNTIJMV, and examine DSNTIJMV to study the updates that it makes. Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV job.

Before starting DB2, check the private area sizes in the SYS1.PROCLIB update section to ensure that you have enough user private area.

Also, examine the size of the private area on the DB2 startup procedures. If necessary, modify them to satisfy the requirements for EDM pool size, buffers, numbers of open data sets, and the amount of available private address space.

From V9.1 If you are migrating from Version 9.1, the installation CLIST removes the _CEE_ENVFILE environment variable from the Language Environment options data set. This action allows the data set to maintain compatibility with the Version 9.1 runtime environment for the SYSPROC.XSR_COMPLETE stored procedure. After you migrate to new-function mode, you must add the _CEE_ENVFILE environment variable as part of enabling the Version 10 runtime environment, as explained in "Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode" on page 661 From V9.1

Related concepts:

Runtime environment for Java routines (Application Programming for Java)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

DSNTIJMV actions:

Job DSNTIJMV updates SYS1.PARMLIB and SYS1.PROCLIB to help identify DB2 to z/OS. It also creates Language Environment options data sets for the DB2 core WLM environment for Java routines.

Specifically, job DSNTIJMV completes the following actions:

1. Updates the following SYS1.PARMLIB members:
   - IEFSSN*xx*

     This member contains an entry for every z/OS subsystem. Unless you change the DB2 subsystem name or the DB2 command prefix, you do not need to change this member. If you change the subsystem name or the command prefix, either change the current member or create a new member.

     Place the primary system's record (JES2 or JES3) record as the first line in an IEFSSN*xx* member. However, if you use SMS, place the SMS line before the primary system. There might also be other products that change position during system initialization. The DB2 line should come after SMS, the JES subsystem, and other vendor products.
   - IEAAPF*xx* or PROG*xx*

Job DSNTIJMV updates IEAAPF*xx* to include the DB2 program libraries (*prefix*.SDSNEXIT, *prefix*.SDSNLOAD, *prefix*.SDXRRESL, and *prefix*.SDSNLINK) as libraries that are authorized by using the authorized program facility (APF).

All libraries that are concatenated with *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in STEPLIB and JOBLIB statements must be APF-authorized.

- LNKLST*xx*

  Job DSNTIJMV updates this member to include the DB2 load module library, *prefix*.SDSNLINK, in the LNKLST*xx*. If you moved the modules from *prefix*.SDSNLINK into another library, edit DSNTIJMV to include that library in the LNKLST*xx*. If you have combined *prefix*.SDSNLINK and *prefix*.SDSNLOAD into one library, edit DSNTIJMV to include the combined library in the LNKLST*xx*.

You can do additional editing for the SYS1.PARMLIB updates. If you are editing DSNTIJMV rather than making the changes directly, you have a choice: You can either include your additional entries for the SYS1.PARMLIB members (IEAAPF*xx* and LNKLST*xx*) at the end of the existing list of entries, or you can place them earlier in the list.

If you include these entries at the end of the existing SYS1.PARMLIB list, ensure that commas (the continuation character) delimit each entry except the last entry.

ECSA size is another SYS1.PARMLIB change to consider at this time. You specify ECSA size in the CSA parameter of the IEASYS00 parameter. Ensure that you have specified an adequate size for this subsystem.

The **IOP parameter** is another SYS1.PARMLIB change to consider at this time. DB2 can schedule synchronous read-write I/Os and prefetch read I/Os under the application address space's I/O scheduling priority. To enable this type of I/O scheduling:

- Use the IOP parameter to set the I/O priority for the address space of a performance group. The IOP parameter is in the IEAIPSxx member of SYS1.PARMLIB.
- Enable z/OS I/O priority scheduling by specifying IOQ=PRTY in the IEAIPSxx member of SYS1.PARMLIB.

2. Renames your Version 8 or Version 9.1 procedures so that they are not replaced by DB2 for z/OS Version 10 procedures.
3. Updates SYS1.PROCLIB to include the following Version 10 procedures, where *ssnm* is the subsystem name:
   - System services address space startup procedure (*ssnm*MSTR)
   - Database services address space startup procedure (*ssnm*DBM1)
   - Distributed data facility address space startup procedure (*ssnm*DIST)
   - Address space startup procedures for the DB2 core WLM environments (*ssnm*WLM1, *ssnm*WLMC, *ssnm*WLMD, *ssnm*WLMG, *ssnm*WLMJ, *ssnm*WLMM, *ssnm*WLMP, *ssnm*WLMR, *ssnm*WLMU, *ssnm*WLMW, and *ssnm*WLMX)
   - WLM sample procedure for stored procedures
   - IRLM address space startup procedure (IRLMPROC)
   - Program preparation procedures
   - Utilities procedure (DSNUPROC)

If you specified a suffix on panel DSNTIPA1, that suffix is appended to data sets &USER..DBRMLIB.DATA.*suffix*, &USER..RUNLIB.LOAD.*suffix*, and &USER..SRCLIB.DATA.*suffix*. To override these data set names, edit the updates to SYS1.PROCLIB.

The STEPLIB concatenation of the *ssnm*DBM1 address space procedure includes a commented-out DD for the IBM Language Environment data set SCEERUN. If your system does not include SCEERUN in the system link list, you must uncomment this DD. Also, if your system link list does not include the Language Environment data set SCEERUN2, you must add it to the STEPLIB concatenation of your *ssnm*DBM1 address space procedure.

4. Creates Language Environment options data sets for the DB2 core WLM environment for Java routines. The data sets are created by job step DSNTIJJ.
5. Defines an environment variable file for the DB2 core WLM environment for Java routines. This step is completed by job step DSNTIJR.
6. Defines a sample JVM startup options file for the DB2 core WLM environment for Java routines. This step is completed by job step DSNTIJS.

Related concepts:

How DB2 assigns I/O priorities (DB2 Performance)

"Language Environment options data sets for the DB2 core Java WLM environment" on page 513

Runtime environment for Java routines (Application Programming for Java)

Migration step 14: Define system data sets: DSNTIJIN

When job DSNTIJIN is tailored for migration, it defines the following non-VSAM data sets: *prefix*.SRCLIB.DATA, *prefix*.RUNLIB.LOAD, and *prefix*.DBRMLIB.DATA.

About this task

From V8 If you are migrating from DB2 Version 8, the installation CLIST adds job step DSNTASKL to job DSNTIJIN. This job step checks for the administrative task scheduler task list data set and defines it if it is not present. From V8

When job DSNTIJIN is tailored for migration, it does not define data sets for new DB2 catalog and directory table spaces and indexes. Catalog and directory data sets are automatically defined by DB2 when you run job DSNTIJTC to tailor the catalog and job DSNTIJEN to enable new-function mode.

When migrating to DB2 Version 10, do not use DSNTIJIN or any other job to define data sets for the DB2 catalog and directory.

Procedure

To define *prefix*.SRCLIB.DATA, *prefix*.RUNLIB.LOAD, and *prefix*.DBRMLIB.DATA:

Run job DSNTIJIN.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 15: Define user authorization exit routines: DSNTIJEX (optional)

You can use job DSNTIJEX to define user authorization exit routines. If you did not run job DSNTIJEX in the release from which you are migrating, you can skip this step.

About this task

Job DSNTIJEX builds sample authorization exit routines from the source code in *prefix*.SDSNSAMP. These sample authorization exit routines are DSN3@SGN and DSN3@ATH, and the user version of the access control authorization exit routine DSNX@XAC. DSNTIJEX then places the exit routines in the *prefix*.SDSNEXIT library.

Job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. You can modify the access control authorization exit routine DSNX@XAC and use DSNTIJEX to assemble and link-edit it. This exit routine allows you to bypass some or most of DB2 authorization checking and to specify your own authorization checking. The DB2 CLIST tailors the JCL in DSNTIJEX to meet the requirements of your site.

The sample authorization exit routines are not the same as the default authorization exit routines that are supplied by DB2. By implementing the sample authorization exit routines, you can provide group names as secondary authorization IDs.

You have the following options regarding exit routines:
- To use the default authorizations, skip job DSNTIJEX.
- To use the sample authorization exit routines, run job DSNTIJEX.
- To use your own authorization exit routines, modify job DSNTIJEX to reference the correct library, and then run it.

Procedure

If you choose to run job DSNTIJEX:

1. Make the following optional adjustments to DSNTIJEX if they apply to your site:
   - DSNXSXAC is a copy of the default access control authorization exit routine that you can modify. This exit routine allows you to bypass some or most of DB2 authorization checking and to specify your own authorization checking. If you do not change the exit routine, you should delete this step.
   - DSNACICS is a stored procedure that invokes user exit routine DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. If you do not need to modify the caller's parameter values, you can use the default DSNACICX exit routine. However, if you need to modify the caller's parameter values, you need to perform the following tasks:

a. Write a user exit routine in assembler, COBOL, C, or PL/I
b. Assemble or compile the source code
c. Link-edit the object code into the DB2 exit routine library

Installation job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX. You can use this step as a model for your program preparation job.

- If you will use the RACF/DB2 external security module (DSNXRXAC) as your DB2 access control authorization exit, modify job DSNTIJEX to refer to DSNXRXAC instead of DSNXSXAC.

2. Run job DSNTIJEX.

   If job DSNTIJEX runs successfully, it produces a return code of 0 or 4.

   If job DSNTIJEX fails or abends, correct the problem, and rerun the job.

Related concepts:

Access control authorization exit routine (Managing Security)

DB2 data access control (Managing Security)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related information:

Exit routines (DB2 Administration Guide)

Migration step 16: IPL z/OS

When you migrate to DB2 Version 10 conversion mode (from Version 8 or Version 9.1), you should complete the z/OS initial program load (IPL) before you start the DB2 subsystem.

Before you begin

Alternatively, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands to define the subsystem to z/OS until the next IPL.

You should complete the IPL before or during migration, but IPLs are not necessary for fallback or remigration. Before you begin the IPL for z/OS, ensure that job DSNTIJMV was run.

Job DSNTIJMV makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member
- Creates new load module libraries in the LNKLST*xx* member

About this task

The load module library SDSNLINK contains the early code. If all of the required maintenance has been applied to your system, the early code is upward compatible with DB2 Version 10. Ensure that the early code pre-conditioning PTFs have been installed on your system before you migrate. The Version 10 early code is

downward compatible with Version 8 or Version 9.1. The Version 10 early code is incompatible with Version 7.

Procedure

To IPL z/OS:

If you are at the appropriate service level for Version 8 or Version 9.1, you can plan ahead, do PARMLIB updates (which are necessary at least to update the APF authorization list), and complete the IPL for z/OS whenever convenient, before you begin your migration. Otherwise, IPL z/OS before you start DB2 Version 10.

Results

After you complete the IPL for z/OS, message DSN3100I is displayed on the z/OS console, stating that DB2 is ready for the START command.

Related tasks:

"Migration step 13: Define DB2 Version 10 to z/OS: DSNTIJMV" on page 593

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

SETPROG Command

SETSSI Command

Migration step 17: Start DB2 Version 10

During migration to DB2 Version 10, start the DB2 subsystem after you run job DSNTIJMV and IPL z/OS.

Procedure

To start DB2 Version 10:

1. Start the IRLM. If you have not requested that DB2 automatically start the IRLM, you should start it before you start DB2. Use the following command, where *irlmproc* is the name that you assigned to the IRLM startup procedure:

   ```
   START irlmproc
   ```

   The *irlmproc* is the value that you specified for the PROC NAME option on installation panel DSNTIPI.

   If you specified YES for the AUTO START option on installation panel DSNTIPI, DB2 starts the IRLM automatically.
2. Start DB2 from the z/OS console with the following command, where -DSN1 is the subsystem command prefix that you defined for DB2, and *DSNZPxxx* is the name of the DB2 initialization parameter module:

   ```
   -DSN1 START DB2 PARM(DSNZPxxx)
   ```

   If you omit the PARM parameter, and you specified a value in the PARAMETER MODULE field on installation panel DSNTIPO3, that value is used as the parameter module name. If you did not specify a parameter module on panel DSNTIPO3, DB2 uses the default, DSNZPARM.

If you have not run job DSNTIJRT, and the administrative task scheduler starts, the administrative task scheduler issues error message DSNA679I to report SQLCODE -805 until you run job DSNTIJRT. The administrative task scheduler starts if you specified an address space procedure name in the ADMIN SCHEDULER field on installation panel DSNTIPX.

If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, possibly *ssnm*DIST, possibly *ssnm*ADMT, and *irlmproc*, where *ssnm* is the DB2 subsystem name, *ssnm*ADMT is the DB2 administrative task scheduler procedure name, and *irlmproc* is the IRLM procedure name.

If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

```
DSNR002I   RESTART COMPLETED
DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
```

In the next migration step, you migrate the DB2 catalog. Before the catalog is migrated, some catalog or directory table spaces are restricted.

The following messages might occur during startup of DB2 because the catalog and directory table spaces are restricted:

- DSNT501I with reason code 00C900A6
- DSNL700I with reason code 00C900A6 (if DDF is auto-started)
- Abend 04E with reason code 00E70014 (during DDL registration)

These restrictions can cause failure of processes that are started as part of normal DB2 startup, such as collection of real-time statistics, until catalog tailoring is completed. Possible actions to restart such processes are explained in the discussion of job DSNTIJTC.

You can determine existing restrictions by issuing this command after you start DB2:

```
-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
```

The preceding command might also generate message DSNT501I with reason code 00C900A6.

If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. A common cause of startup failure is that the BSDS does not match the subsystem parameter values; ensure that the correct job was run for DSNTIJUZ. Also, check that the subsystem parameter member that you specified (or used by default) when you started DB2 is the one that the DSNTIJUZ job built. Check the JCL for the DB2 startup procedure.

3. If you want to use the TSO SUBMIT command to do housekeeping and migration verification, start TSO (if it is not already started).
4. If you did not enable primary user IDs to issue DB2 commands from the z/OS console or TSO SDSF when you first installed the subsystem, take one of the following actions:
   - Grant SYSOPR authority to all primary user IDs that issue DB2 commands from the z/OS console or TSO SDSF. Issue the following command:

     ```
     GRANT SYSOPR TO userid
     ```

   - Define RACF classes to authorize DB2 commands. Use the following statements:

     ```
     SETR CLASSACT(DSNADM)
     RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
     SETR RACLIST(DSNADM) REFRESH
     PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
     SETR RACLIST(DSNADM) REFRESH
     ```

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 18: Tailor DB2 Version 10 catalog: DSNTIJTC

Job DSNTIJTC invokes the CATMAINT utility to migrate the catalog of your previous DB2 release to the Version 10 catalog. To execute DSNTIJTC, you must have installation SYSADM authority.

Before you begin

Starting in DB2 Version 10, data sets for the DB2 catalog and directory are managed by DB2 and must be SMS-managed. The SMS environment that you use for DB2 catalog and directory data sets must be established before you begin installation or migration to Version 10. The SMS environment must include a data class for allocating the data sets in extended format and using extended addressability. You are not required to convert existing DB2 catalog and directory data sets to the SMS environment before migrating to Version 10. These data sets will be converted the next time the related table space is reorganized.

In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

Procedure

To complete the tailoring of the DB2 catalog:

1. Run job DSNTIJTC on your Version 10 subsystem.

   A status message, DSNU777I, is issued at several points to indicate migration progress. Diagnostic error messages are issued when CATMAINT processing fails. If a problem is found during the SQL processing phase of migration, message DSNU778I is issued. If non-supported functions are encountered, message DSNU776I is issued. All of these messages are written to the SYSPRINT data set.

   If you receive abend reason code 00C200EF, the catalog and directory data sets are not in a compliant SMS environment.
2. If job DSNTIJTC fails, save the output and verify that you are at the correct maintenance level.
   a. If you are not at the correct maintenance level, install the appropriate maintenance.
   b. If you are at the correct maintenance level, correct the problem,
   c. Run job DSNTIJTC again.
3. If job DSNTIJTC fails again, return to your previous DB2 release. Because CATMAINT failures roll back all Version 10 changes, the catalog and directory are in the format of your previous DB2 release. Altered indexes are not rolled back. Determine if any index is in a pending status by using the CHECK INDEX utility. To return to your previous DB2 release:
   a. Rename procedures to use the libraries of your previous release.
   b. Reconnect TSO, IMS, and CICS to the libraries of your previous release.

4. If your previous release is damaged, you need to complete a point-in-time recovery to recover the system. To complete a point-in-time recovery:
   a. Restore your previous version's catalog and directory from image copies.
   b. Restore the BSDSs from archive logs that were made prior to migration.
   c. Flush the SCA (for data sharing environments only).
   d. Recover the catalog and directory indexes.

   When you remigrate to Version 10, run the DSNTIJTC job again.

Results

DSNTIJTC contains one step that creates new catalog and directory objects, adds columns to existing catalog tables, and creates and updates indexes on the catalog tables to accommodate new Version 10 objects. All IBM-supplied indexes are created or updated sequentially during the execution of DSNTIJTC.

When you start DB2 Version 10 for the first time, you might receive message DSNT501I with reason code 00C900A6. You can ignore this message. Running job DSNTIJTC corrects the cause of this message.

After CATMAINT has run successfully and module DSNICUBC issues message DSNT501I, you can restart your real-time statistics collection by stopping and starting the DSNRTSDB database. In a data sharing group, migrate all members to Version 10 before stopping and starting the DSNRTSDB database.

Related concepts:

"Falling back" on page 638

"Work file database storage requirements" on page 35

Recovery of data to a prior point in time (DB2 Administration Guide)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

CATMAINT (DB2 Utilities)

Related information:

DSNU776I (DB2 Messages)

DSNU777I (DB2 Messages)

DSNU778I (DB2 Messages)

Migration step 19: Ensure that the catalog has no problems (optional)

After you tailor your DB2 10 for z/OS catalog, you should check the integrity of your catalog and directory.

Procedure

To check the integrity of your catalog and directory, complete these steps in any order:

- Run CHECK INDEX on all the indexes in the catalog and directory by using job DSNTIJCX.
- Run the link checker (DSN1CHKR) to ensure that no existing links are broken.
- Run the queries in member DSNTESQ of *prefix*.SDSNSAMP. Because SPUFI is not yet bound, you cannot use SPUFI to run these queries.

  From V8 If you are migrating from Version 8, you can use the Version 8 DSNTEP2 program to run the queries. From V8

  From V9.1 If you are migrating from Version 9.1, you can use the Version 9.1 DSNTEP2 program to run the queries. From V9.1
- Run the DSN1COPY utility with the CHECK option on the catalog table spaces.

Related tasks:

"Migration step 2: Run the link checker on your DB2 table spaces (optional)" on page 580

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

DSN1COPY (DB2 Utilities)

Migration step 20: Rebuild indexes: DSNTIJRI (optional)

During migration to Version 10, some indexes are put in the advisory rebuild-pending state when columns are added to the index.

From V8 If you are migrating from DB2 Version 8, the following indexes might be affected:

- DSNADX01
- DSNAGX01
- DSNAPX01
- DSNATX01
- DSNAUX02
- DSNKAX01
- DSNKAX02
- DSNOAX01
- DSNOFX01
- DSNOFX02
- DSNOFX07
- DSNSKX01
- DSNWCX02
- DSNWCX03
- DSNXRX01

From V8

From V9.1 If you are migrating from DB2 Version 9.1, the following index might be affected:

- DSNXRX01

From V9.1

To rebuild these indexes, run job DSNTIJRI.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 21: Enable change data capture

During migration to Version 10 conversion mode (from Version 8 or Version 9.1), DB2 disables change data capture on the SYSROUTINES catalog table. If you want to use change data capture on this table, you must re-enable it.

Procedure

To re-enable change data capture:

Issue the following statement for each table for which you want to re-enable change data capture:

GUPI

```
ALTER TABLE SYSIBM.tablename DATA CAPTURE CHANGES;
```

Where *tablename* is the name of the catalog table. GUPI

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related reference:

ALTER TABLE (DB2 SQL)

Migration step 22: Prepare dynamic SQL program: DSNTIJTM

Job DSNTIJTM assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically.

Procedure

To prepare the dynamic SQL program:

1. Determine whether a BIND PLAN statement is needed for DB2 REXX Language Support. DSNTIJTM enables DB2 REXX Language Support for use by DSNTWFG, a REXX program called by DSNTIJTM to create table spaces in the work file database. For migration, the CLIST removes the BIND PLAN statement for DB2 REXX Language Support because it is assumed that the REXX plan was created in a previous release. If the REXX plan was not created in a previous release, you need to add a command like the following after the bind statements for the DB2 REXX Language Support package:

```
 BIND PLAN(DSNREXX) -
PKLIST(*.DSNREXX.DSNREXX, -
*.DSNREXUR.DSNREXX, -
*.DSNREXCS.DSNREXX, -
*.DSNREXRS.DSNREXX, -
*.DSNREXRR.DSNREXX) -
ACT(REP) ISO(CS) SQLRULES(DB2) -
ENCODING(EBCDIC)
```

2. Review and possibly modify or delete the GRANT statements in job DSNTIJTM. This job grants PUBLIC access to a number of DB2-supplied objects. Consider granting execute access on the plan to PUBLIC.
3. Run job DSNTIJTM.

Results

In DB2 versions prior to Version 9.1, DSNTIJTM created work file table spaces only for the initial installation. In Version 9.1 and later, DSNTIJTM optionally creates additional table spaces in the work file database.

If you do not have an existing 32 KB work file table space, be sure to create at least one during migration.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 23: Bind SPUFI and DCLGEN and user-maintained database activity: DSNTIJSG

Use job DSNTIJSG to rebind packages for SPUFI, DCLGEN, and the EXEC SQL utility.

About this task

In migration mode, DSNTIJSG completes the following actions:

- Rebinds packages at the new release level for SPUFI, DCLGEN, and the EXEC SQL utility.

  From V8 If you are migrating from Version 8, packages for SPUFI are not bound by this job. From V8
- Creates a new resource limit facility table, DSNRLMTnn, and a new resource limit facility index, DSNMRLnn. By default, the step that creates this new table and index is commented out in the JCL. To have DB2 create DSNRLMTnn and DSNMRLnn, uncomment the step.
- From V8 If you are migrating from DB2 Version 8, DSNTIJF, the last step of DSNTIJSG, frees an obsolete plan and package named DSNHYCRD. You can expect this step to end with return code 12 if the DSNHYCRD package or plan is not present. From V8

Procedure

To rebind your packages:

1. If you bound special SPUFI packages and plans in Version 8 or Version 9.1, you need to bind those packages again in Version 10. You do not need to bind those plans again. For example, to update special SPUFI packages that were created for use by SPUFI users who require a TSO terminal CCSID of 1047, issue the following commands:

   GUPI

   ```
   BIND PACKAGE(TIAP1047) MEMBER(DSNTIAP) -
        ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
        ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
        ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   ```

   GUPI

2. From V8 Beginning in Version 9, SPUFI provides an option to select data with a cursor isolation level of Uncommitted Read. To add a special package and plan with ISO(UR) for SPUFI users who require a TSO terminal of CCSID 1047, issue the following commands:

   GUPI

   ```
   BIND PACKAGE(SPUR1047) MEMBER(DSNESM68) -
        ACTION(REPLACE) ISOLATION(UR) ENCODING(1047) -
        LIBRARY('prefix.SDSNDBRM')
   BIND PLAN(SPUR1047) -
        PKLIST(*.SPUR1047.DSNESM68, -
               *.TIAP1047.DSNTIAP) -
        ISOLATION(UR) ENCODING(1047) ACTION(REPLACE)
   ```

   GUPI From V8

3. If you are using a data sharing environment, you must ensure that the resource limit facility (RLF) is inactive on all members in the data sharing group before running DSNTIJSG. To do this, issue the STOP RLIMIT command for each member.
4. Ensure that all your subsystems have the same maintenance level for your DSNUGSQL DBRM.
5. Run job DSNTIJSG, If DSNTIJSG runs successfully, it produces a return code of 0. However, job step DSNTIJG can produce a return code of 4 if the specified access has already been granted on one or more objects.

   From V8 Also, if you are migrating from DB2 Version 8, job step DSNTIJF can end with return code 8 if the package and plan for DSNHYCRD does not exist.

   From V8

   DB2 issues a bind warning for each plan. Expect the following messages:

   ```
   DSNE932I  WARNING, ONLY IBM-SUPPLIED PLAN NAMES SHOULD BEGIN WITH DSN
   DSNE932I  WARNING, ONLY IBM-SUPPLIED PACKAGE-IDS SHOULD BEGIN WITH DSN
   DSNE932I  WARNING, ONLY IBM-SUPPLIED COLLECTION-IDS SHOULD BEGIN WITH DSN
   ```

6. If job DSNTIJSG fails or abends, ensure that the user that is specified on the JOB statement is authorized. Use the name that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1. (The RESTART parameter on the JOB statement can be useful.) Correct any other problems, and run job DSNTIJSG again.

7. If you encounter resource shortages, review the parameters in job DSNTIJUZ and make any necessary modifications. Then, stop DB2, rerun DSNTIJUZ, start DB2, and rerun DSNTIJSG from the last successful step.

What to do next

Recommendation: Alter your DB2 Version 10 buffer pools that have frequent page reads or frequent page writes to use PGFIX YES if you have sufficient real storage available for these buffer pools. Fixing the buffer pages in real storage once and keeping them fixed avoids the processing time that DB2 needs to fix and free pages each time there is an I/O. In some cases, this processing time can be as much as 10% for I/O-intensive workloads. To use this option, issue the following command:

GUPI

```
ALTER BPOOL(bpname) VPSIZE(vpsize) PGFIX(YES)
```

GUPI

Where *bpname* is the name of the buffer pool and *vpsize* is the size of the virtual pool.

Related tasks:

"Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode" on page 661

"Special packages and plans for SPUFI" on page 568

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 24: Convert EXPLAIN tables to the current format and encoding type

When you migrate to DB2 Version 10, you should convert your EXPLAIN tables to the current (Version 10) format and Unicode encoding.

A table that is not in the current format has fewer columns than are available in the current release, or it has columns that have a different length or data type than expected in the current release. Unicode EXPLAIN tables are preferred because the DB2 catalog tables are encoded in Unicode.

Before you begin

Recommendation: Before you begin migration to Version 10, convert all EXPLAIN tables to the Version 8 or Version 9 format and Unicode encoding. For more information, see "Converting EXPLAIN tables (before migration)" on page 64.

Before you convert the EXPLAIN tables to Version 10 format, your DB2 Version 10 subsystem must be in conversion mode from Version 8 or conversion mode from Version 9.1.

About this task

Starting in DB2 Version 10 conversion mode (from both Version 8 and Version 9.1), EXPLAIN tables must be in Version 8 or later format and preferably encoded in Unicode. DB2 returns SQLCODE -20008 reason code 2 for statements or commands that invoke EXPLAIN processing if the EXPLAIN tables are in a pre-Version 8 format. Statements or commands that invoke EXPLAIN processing return SQLCODE +20520 reason code 2 if an EXPLAIN table is in Version 8 or Version 9.1 format, regardless of the encoding type. After you convert your EXPLAIN tables to Version 10 format, you must also convert the tables to Unicode encoding. If an EXPLAIN table is in Version 10 format and encoded in EBCDIC, SQLCODE -878 is returned.

Procedure

To convert EXPLAIN tables:

1. If premigration job DSNTIJPM report 15 identifies any EXPLAIN tables that are not in Version 10 format, customize job DSNTIJXA by providing the creator ID of the tables to migrate to the Version 10 format, a DB2 subsystem name, and an authorization ID. To migrate all EXPLAIN tables, specify an asterisk as the creator ID.

   From V8 The Version 10 format of the DSN_STATEMENT_CACHE_TABLE EXPLAIN table is not supported in conversion mode from Version 8. If you run DSNTIJXA in conversion mode from Version 8, this table is converted to Version 9.1 format. From V8

   Also, if you ran installation job DSNTIJSG, applied the PTF for APAR PM28500, and then ran DSNTIJXA, the DSN_PROFILE_TABLE and DSN_PROFILE_HISTORY tables contain a downlevel IPADDR column name. This occurs because renaming of columns is not supported in conversion mode from Version 8.

   To convert these tables to Version 10 format, you must run DSNTIJXA again in new-function mode. You can also choose to skip running DSNTIJXA in conversion mode from Version 8 and just run it once in new-function mode. If you choose to run DSNTIJXA once in new-function mode, stop this procedure here.
2. Run job DSNTIJXA. Job DSNTIJXA calls REXX exec DSNTXTA, which alters to the Version 10 format all EXPLAIN tables or EXPLAIN tables that belong to the specified creator ID.

   Exception: From V8 In conversion mode from Version 8, DSNTIJXA cannot convert DSN_QUERY_TABLE and DSN_PTASK_TABLE from the Version 8 format. Use job DSNTIJXB to convert these tables instead. From V8
3. If premigration job DSNTIJPM report 14 identifies any EXPLAIN tables that use EBCDIC encoding, convert them to Unicode by completing the following steps.
   a. Customize job DSNTIJXB with the following values:
      - A DB2 subsystem name
      - An authorization ID
      - The schema name of the EXPLAIN tables to convert
      - The name of the target database
      - The name of the target 4 KB page table space
      - The name of the target 8 KB page table space

- The name of the target 16 KB page table space
- The name of the target 8 KB page LOB table space
- The name of the target 32 KB page LOB table space

b. Run job DSNTIJXB. Job DSNTIJXB calls REXX exec DSNTXTB, which generates DB2 cross-loader control statements and equivalent SQL statements that can be used to copy all EBCDIC EXPLAIN tables under a specified creator ID to Unicode equivalents. All EXPLAIN tables that belong to the specified creator ID must be in the current release format except for DSN_QUERY_TABLE, which can be in Version 8 format.

c. If you have purchased the DB2 Utilities Suite, run job DSNTIJXC to process the control statements that were generated by DSNTIJXB. Otherwise, use the equivalent SQL statements that were generated by DSNTIJXB to convert the tables to Unicode.

d. After you convert an EXPLAIN table from EBCDIC encoding to Unicode encoding, check for joins to that table. When you convert EXPLAIN tables to Unicode, applications that join with EXPLAIN tables might have different results because of the CCSID conversion. For more information, see Objects with different CCSIDs in the same SQL statement (DB2 Internationalization Guide).

e. Repeat steps a, b, c, and d for each creator that has at least one EBCDIC EXPLAIN table.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 25: Configure DB2 for running stored procedures and user-defined functions

Before you can use user-defined functions or the following types of stored procedures, you need to perform some set-up tasks:

- External stored procedures
- External SQL procedures
- Native SQL procedures that satisfy at least one of the following conditions:
  - The native SQL procedure calls at least one external stored procedure, external SQL procedure, or user-defined function.
  - The native SQL procedure is defined with ALLOW DEBUG MODE or DISALLOW DEBUG MODE.
- DB2-supplied stored procedures

All routines, except native SQL procedures, must run in an environment that is established by the z/OS Workload Manager (WLM). A set of DB2-supplied routines are installed by job DSNTIJRT and validated by job DSNTIJRV as part of the migration process. You can also define your own routines or use routines that are provided by a non-IBM vendor. Before you can use any stored procedures and user-defined functions, however, you need to set up WLM environments for these routines. If you plan to use certain DB2-supplied stored procedures or user-defined functions, you need to perform some additional steps.

Related concepts:

Stored procedures (DB2 Application programming and SQL)

User-defined functions (DB2 SQL)

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Tailoring DB2 subsystem parameters for stored procedures during migration (optional):

Several subsystem parameters should be tailored to your DB2 environment before you use stored procedures. These parameters were set when you installed or migrated DB2. If you are satisfied with those values, you do not need to complete this step.

About this task

The subsystem parameters for stored procedures are:

- MAX_NUM_CUR
- MAX_ST_PROC
- STORMXAB
- STORTIME

Procedure

To modify the values of these subsystem parameters, complete one of the following tasks:

- Run the installation CLIST again in INSTALL or MIGRATE mode. On the installation panels, leave most of the existing values and change the following fields on their respective panels:
  1. On panel DSNTIPA1, specify the input member that contains field values for your current installation.
  2. On panel DSNTIPT, specify different names for the TEMP CLIST LIBRARY and SAMPLE LIBRARY fields to avoid overwriting your original libraries.
  3. On panel DSNTIPX, add or change the values of the MAX ABEND COUNT, TIMEOUT VALUE, MAX OPEN CURSORS, and MAX STORED PROCS fields as appropriate.
  4. On panel DSNTIPY, specify a name for the REMOTE LOCATION field. DB2 uses this name for the stored procedure sample applications.
- Edit job DSNTIJUZ to add or change values for the subsystem parameters for routines.

  This method has the following disadvantages:
  - DB2 does not use these new values for any future migrations because your DSNTID*xx* member and DSNTIJUZ parameters are not saved for future input.
  - DB2 does not generate the sample jobs for stored procedure and user-defined functions because you are not running the DSNTINST CLIST.

Related reference:

"Main panel: DSNTIPA1" on page 265
"Data set names panel 1: DSNTIPT" on page 295
"Routine parameters panel: DSNTIPX" on page 453
"Job editing panel: DSNTIPY" on page 488
"Directory of subsystem parameters and application default values" on page 242

Setting up Language Environment for stored procedures during migration:

Language Environment (LE) is a component of the z/OS operating system that establishes a common runtime environment for all applications, regardless of programming language. DB2 uses Language Environment to provide a runtime environment for external stored procedures and external SQL procedures.

About this task

Language Environment provides a set of essential runtime services that are available through a set of interfaces that are consistent across programming languages. These services include routines for runtime message handling, condition handling, and storage management.

Language Environment provides the following advantages for DB2 stored procedures:

- Language Environment hides the differences between programming languages.
- Language Environment provides the ability to make a stored procedure resident in the stored procedure address spaces.
- Language Environment supports a large number of runtime options, including those options that are needed to use tools to debug your stored procedures.

Procedure

To set up Language Environment for stored procedures:

1. Customize Language Environment by completing the instructions in z/OS Language Environment Customization.
2. Specify the Language Environment runtime library in the JCL startup procedure for each WLM-managed address space for stored procedures. This library must be the only available runtime library. Do not reference any other language runtime libraries within the system link list or within the JOBLIB or STEPLIB statements. If other language runtime libraries are defined in the system link list, use JOBLIB or STEPLIB statement overrides to exclude them.

   Multiple languages can share the same Language Environment runtime library. Therefore, you do not need to specify the language-specific libraries in the JCL startup procedure of each stored procedure address space.

   Restriction: Do not use compilers (or associated runtime libraries) that are no longer supported by IBM.

What to do next

After you set up Language Environment for stored procedures, you can override the Language Environment runtime options for individual stored procedures. You can do this by using the RUN OPTIONS clause in the CREATE PROCEDURE statement or ALTER PROCEDURE statement. The default values for Language

Environment runtime options are defined by z/OS. You might have overridden some of these default values when you customized Language Environment.

Related concepts:

External stored procedures (DB2 Application programming and SQL)

SQL procedures (DB2 Application programming and SQL)

Language Environment setup (DB2 9 for z/OS Stored Procedures: Through the CALL and beyond)

Related reference:

CREATE PROCEDURE (external) (DB2 SQL)

CREATE PROCEDURE (SQL - external) (DB2 SQL)

Customizing Language Environment run-time options (z/OS Language Environment Customization)

Implementing RRS for stored procedures during migration:

To use some DB2 stored procedures, you must implement Resource Recovery Services (RRS). RRS is a sync point manager program that coordinates two-phase commit processing. DB2 requires that RRS be active because WLM-managed stored procedure address spaces use the RRS attachment facility (RRSAF).

About this task

RRS records events for protected resources in log streams that are managed by a z/OS component called the system logger. The following table identifies the log streams that RRS uses, where *gname* can be your Sysplex name or any name in a non-Sysplex environment:

Table 87. Log streams that RRS uses

Log stream type	Log stream name	Content
Main unit-of-recovery log state stream	ATR.*gname*.MAIN.UR	The state of active units of recover (URs). RRS periodically moves this information into the RRS delayed UR state log when UR completion is delayed.
Delayed unit-of-recovery log state stream	ATR.*gname*.DELAYED.UR	The state of active URs, when UR completion is delayed.
Resource manager data log stream	ATR.*gname*.RM.DATA	Information about the resource managers that use RRS services.
Restart log stream	ATR.*gname*.RESTART	Information, which is needed during restart, about incomplete URs. This information enables a functioning RRS instance to assume remaining incomplete work from an RRS instance that failed.
Archive log stream	ATR.*gname*.ARCHIVE	Information about completed URs. This log stream is optional, but recommended.

All of these log streams are required except for the archive log stream. RRS does not start if it is not able to connect to the required log streams.

In a Sysplex environment, these log streams are shared by the systems of the Sysplex.

Procedure

To implement RRS for stored procedures:

1. Define the RRS log streams that are to be written, either on disk or in the coupling facility by completing the following actions:
   a. If you are using the coupling facility, add definitions for the RRS structures in the CFRM policy.

      When you are use coupling facility log streams, each log stream needs to be mapped to a coupling facility structure.

      Example: The following example shows JCL that updates the CFRM policy to add the RRS structures:

```
//DEFCFRM1 JOB MSGCLASS=X,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN DD *
    DATA TYPE(CFRM) REPORT(YES)
    DEFINE POLICY NAME(CFRM18) REPLACE(YES)
    CF NAME(CF01)
        TYPE(009672)
        MFG(IBM)
        PLANT(02)
        SEQUENCE(000000040104)
        PARTITION(1)
        CPCID(00)
        DUMPSPACE(2048)
        CF NAME(CF02)
        TYPE(009672)
        MFG(IBM)
        PLANT(02)
................
................
................
STRUCTURE NAME(RRS_ARCHIVE_1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF1,CF2)
               REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_RMDATA_1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF1,CF2)
               REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_MAINUR_1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF1,CF2)
               REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_DELAYEDUR_1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF1,CF2)
               REBUILDPERCENT(5)
        STRUCTURE NAME(RRS_RESTART_1)
```

```
                INITSIZE(8000)
                SIZE(16000)
                PREFLIST(CF1,CF2)
                REBUILDPERCENT(5)
```

b. Define the log streams in the system logger policy by using the IXCMIAPU utility. This utility program is provided in the SYS1.MIGLIB system library.

   If you are using the coupling facility, also define the structures in the system logger policy. You can map each log stream to a single structure or you can map log streams of like data types to the same structure.

   Example of defining coupling facility log streams: The following example JCL shows definitions for coupling facility structures and RRS log streams. In this case, each RRS log stream is mapped to a structure.

```
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DATA TYPE(LOGR) REPORT(YES)

  DEFINE STRUCTURE NAME(RRS_ARCHIVE_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(262)

  DEFINE STRUCTURE NAME(RRS_RMDATA_1) LOGSNUM(1)
         MAXBUFSIZE(1024) AVGBUFSIZE(252)

  DEFINE STRUCTURE NAME(RRS_MAINUR_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

  DEFINE STRUCTURE NAME(RRS_DELAYEDUR_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

  DEFINE STRUCTURE NAME(RRS_RESTART_1) LOGSNUM(1)
         MAXBUFSIZE(64000) AVGBUFSIZE(158)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.ARCHIVE) STRUCTNAME(RRS_ARCHIVE_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.RM.DATA) STRUCTNAME(RRS_RMDATA_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.MAIN.UR) STRUCTNAME(RRS_MAINUR_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.DELAYED.UR) STRUCTNAME(RRS_DELAYEDUR_1)
LS_DATACLAS(SHARE33)
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)

DEFINE LOGSTREAM
NAME(ATR.SANDBOX.RESTART) STRUCTNAME(RRS_RESTART_1)
LS_DATACLAS(SHARE33)
```

```
HLQ(LOGR) MODEL(NO) LS_SIZE(1024)
LOWOFFLOAD(0) HIGHOFFLOAD(80) STG_DUPLEX(NO)
RETPD(15) AUTODELETE(YES)
/*
```

Example of defining disk log streams: The following example JCL shows definitions for RRS log streams on disk.

```
/DEFSTREXECPGM=IXCMIAPU
//SYSPRINT DDSYSOUT=A
//SYSINDD *
   DATA TYPE(LOGR)

   DEFINE LOGSTREAM NAME(ATR.PLEX1.MAIN.UR)
      LOWOFFLOAD(60)
      HIGHOFFLOAD(80)
      DASDONLY(YES)
      HLQ(TEST)
      LS_SIZE(1024)
      LS_DATACLAS(VSAMLS)
      STG_SIZE(1024)

   DEFINE LOGSTREAM NAME(ATR.PLEX1.DELAYED.UR)
      LOWOFFLOAD(60)
      HIGHOFFLOAD(80)
      DASDONLY(YES)
      HLQ(TEST)
      LS_SIZE(960)
      LS_DATACLAS(VSAMLS)
      STG_SIZE(960)

  DEFINE LOGSTREAM NAME(ATR.PLEX1.ARCHIVE)
     LOWOFFLOAD(0)
     HIGHOFFLOAD(80)
     DASDONLY(YES)
     HLQ(TEST)
     LS_SIZE(960)
     LS_DATACLAS(VSAMLS)
     AUTODELETE(YES)
     RETPD(2)
     STG_SIZE(2000)

  DEFINE LOGSTREAM NAME(ATR.PLEX1.RM.DATA)
     LOWOFFLOAD(60)
     HIGHOFFLOAD(80)
     DASDONLY(YES)
     HLQ(TEST)
     LS_SIZE(192)
     LS_DATACLAS(VSAMLS)
     STG_SIZE(192)

  DEFINE LOGSTREAM NAME(ATR.PLEX1.RESTART)
     LOWOFFLOAD(60)
     HIGHOFFLOAD(80)
     DASDONLY(YES)
     HLQ(TEST)
     LS_SIZE(960)
     LS_DATACLAS(VSAMLS)
     STG_SIZE(960)
```

c. If you are using coupling facility log streams, compile and link-edit the changed CFRM policy.

d. If you are using coupling facility log streams, activate the updated CFRM policy in your Sysplex by using the following operator command:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

2. Establish the priority for RSS within the WLM policy.

The RRS priority needs to be equal to or higher than the dispatching priority of DB2. You can use the SYSSTC service class for the RRS address space to achieve a higher dispatching priority.

3. Move the ATRRRS procedure that is supplied by IBM from SYS1.SAMPLIB to your SYS1.PROCLIB as member RRS, and make any necessary changes. This JCL procedure is used to start RRS.

   You can use a member name other than RRS, but the first four characters of this name must match the subsystem name that you specify in the IEFSSN*xx* member of SYS1.PARMLIB in the next step.

   The GNAME value in the procedure must match the *gname* that you specified when defining the log streams. The default value for *gname* at RRS startup is the Sysplex name.

   Example: The following example shows a sample procedure to start RRS.

   ```
   //RRS   PROC GNAME='',CTMEM=''
   //RRS   EXEC PGM=ATRIMIKE,REGION=0M,TIME=NOLIMIT,
   //          PARM='GNAME=&GNAME CTMEM=&CTMEM'
   //
   ```

4. Define RRS as a subsystem to z/OS by editing the active IEFSSN*xx* member of SYS1.PARMLIB to include the following entry:

   ```
   SUBSYS SUBNAME(RRS) /* RESOURCE RECOVERY SERVICES */
   ```

   The subsystem name can be RRS or any other name that you choose. However, the first four characters (or fewer characters if the name is shorter) must match the name of the JCL procedure to start RRS.

5. Start RRS by using the following operator command:

   ```
   START RRS
   ```

   If you subsequently need to stop RRS, use one of the following operator commands:

   - `SETRRS CANCEL`
   - `SETRRS SHUTDOWN`

   If neither of these commands stops RRS, use the `FORCE RRS,ARM` command. In this command, RRS is the subsystem name that your installation assigned to RRS in PARMLIB member IEFSSN*xx*.

Related concepts:

RRSAF (DB2 9 for z/OS Stored Procedures: Through the CALL and beyond)

Related reference:

z/OS MVS Programming: Resource Recovery

Systems Programmer's Guide to Resource Recovery Services (RRS)

Setting up a WLM application environment for stored procedures during migration:

All stored procedures and user-defined functions (except native SQL procedures) must run in WLM-managed stored procedure address spaces. You must assign each stored procedure to a WLM application environment, which routes the stored procedure work to the appropriate WLM-managed address space.

About this task

If you use data sharing, see "Special considerations for WLM-managed stored procedures in coexistence" on page 723 for additional guidance about setting up WLM environments.

For DB2-supplied routines, you can use the provided definitions for core WLM environments as a starting point for defining your environments. You can use job DSNTIJRW to define and activate these core environments. For more information on using job DSNTIJRW, see "Setting up the DB2 core WLM environments during installation" on page 547. Job DSNTIJMV adds to SYS1.PROCLIB the startup address space procedures for the core WLM environments.

Recommendation: Use partitioned data set extended (PDSE) for load libraries that contain external modules for stored procedures. Using PDSEs might eliminate your need to stop and start the stored procedures address space due to growth of the load libraries. If a load library grows from additions or replacements, the library might need to be extended.

Procedure

To set up a WLM application environment for one or more stored procedures:

1. Gather requirements for the WLM environment. These requirements include:
   - The name of the DB2 subsystem or data sharing group where the stored procedure or user-defined function is to run.
   - The maximum number of task control blocks (TCBs) that can run concurrently in an address space that is initiated by the WLM environment. This setting, referred to as the NUMTCB, is the maximum number of procedures or functions that should run concurrently in that WLM-managed address space. The appropriate setting depends on the requirements or limitations of the individual stored procedure or user-defined function.
     - For recommended NUMTCB values for DB2-supplied stored procedures, see "DB2-supplied stored procedures and user-defined functions" on page 833.
     - For recommended NUMTCB values for your own stored procedures, see Assigning stored procedures and functions to WLM application environments (DB2 Performance). If a routine allocates a large amount of memory or if the external module for the routine is not re-entrant, the NUMTCB value should be 1. If there is a need for serialization, the NUMTCB value should also be 1.
     - The WLM environment for a REXX stored procedure must have a NUMTCB setting of 1.
   - Requirements for Authorized Program Facility (APF).
     - Stored procedures and user-defined functions that execute privileged instructions (for example, WLM_REFRESH) must run in a WLM environment that has an APF-authorized address space.
     - DB2 REXX stored procedures, such as DSNTPSMP, and other routines that use TSO/E and REXX facilities must run in a WLM environment that does not have an APF-authorized address space.
     - Most stored procedures can run in a WLM environment that either has or does not have an APF-authorized address space. However, running a routine in an APF-authorized environment is recommended only when doing so is necessary.

- Special data sets and DD names that are needed by the stored procedure or user-defined function. For example, all Java stored procedures implicitly require a Language Environment runtime options data set to be allocated to DD name JAVAENV. Other routines might also have an explicit requirement for a work file to be allocated to a particular DD name.

2. Create a JCL startup procedure for the WLM application environment.

   Use *ssnm*WLM (where *ssnm* is the DB2 subsystem ID) as the basis for creating the new startup procedure. *ssnm*WLM is the default name of the sample JCL procedure for starting WLM-established address spaces. If you specified a name in the WLM PROC NAME field on the DSNTIPX installation panel, the sample procedure has that specified name. The sample procedure is placed in your system PROCLIB by installation job DSNTIJMV. For the DSNACICS stored procedure, DSNTIJMV also provides sample procedure *ssnm*CICS, where *ssnm* is the subsystem ID. Tailor your new startup procedure as follows:

   - Change the APPLENV value to the name of the WLM application environment that you plan to define.
   - Change the startup procedure name to an appropriate procedure name.

     Recommendation: Specify the same name for the application environment name and the procedure name. Using the same name simplifies monitoring the WLM environment definitions.
   - Change the value of DB2SSN to your DB2 subsystem name.
   - Ensure that the NUMTCB value is appropriate. If the minimum number of address spaces is specified, the NUMTCB value will be the number of task control blocks that are started in an address space.

     Recommendation: If you want to be able to easily change the NUMTCB value, specify the NUMTCB value here, and do not specify this parameter on the application environment definition panel when you define the WLM application environment. Specifying NUMTCB on the environment definition panel associates this parameter with the WLM application environment definition. Any changes to the JCL startup procedure can be made by refreshing the WLM application environment. However, changes to the WLM application environment require that you reinstall the z/OS service policy at an LPAR or sysplex level. If you specify the NUMTCB parameter for both the WLM application environment and the startup procedure, the value in the WLM application environment overrides the value in the startup procedure.
   - If the environment is one in which the cost of starting new address spaces is very high, you can use the MNSPAS parameter. The MNSPAS parameter specifies the minimum number of address spaces that are to be started and maintained. The value can be 0 through 50. The default value is 0, which means that no minimum number of address spaces is specified. If the number of address spaces is at the specified minimum, cancelling an address space causes WLM to start another one to maintain the minimum. If you refresh a WLM environment, WLM starts a number of new address spaces as needed (or the specified minimum number) and stops the old address spaces.

     If the MNSPAS parameter is specified, the NUMTCB value is the number of task control blocks that are started in an address space. If the MNSPAS parameter is not specified, the NUMTCB value is the maximum number of task control blocks that can be started in an address space.
   - Edit the data set names to match your data set names.

- Ensure that the library in which the stored procedure resides is in the STEPLIB concatenation.
- Ensure that the IBM Language Environment runtime library is specified in the STEPLIB concatenation, unless your system includes the Language Environment runtime library in the system link list.
- If you want to run the stored procedure in a WLM environment that has an APF-authorized address space, ensure that all the libraries in the STEPLIB concatenation are defined to the Authorized Program Facility.
- Add any special DD statements that are required for the stored procedures or user-defined functions that are to run in the WLM environment. If you plan to use stored procedures that use the ODBA interface to access IMS databases, make the following additional changes:
  - Add the data set name of the IMS data set that contains the ODBA callable interface code to the end of the STEPLIB concatenation. The name of this data set is usually IMS.RESLIB.
  - After the STEPLIB DD statement, add a DFSRESLB DD statement that names the IMS data set that contains the ODBA callable interface code.

Example of a startup procedure: The following example shows a startup procedure for a WLM application environment for external SQL, COBOL, C, C++, PL/I, and assembler stored procedures. This startup procedure for external SQL stored procedures needs one unauthorized data set included in STEPLIB.

```
//*******************************************************************
//* JCL FOR RUNNING THE WLM-ESTABLISHED STORED PROCEDURES
//* ADDRESS SPACE
//* RGN     -- THE MVS REGION SIZE FOR THE ADDRESS SPACE.
//* DB2SSN  -- THE DB2 SUBSYSTEM NAME.
//* NUMTCB  -- THE NUMBER OF TCBS USED TO PROCESS
//*            END USER REQUESTS.
//* APPLENV -- THE MVS WLM APPLICATION ENVIRONMENT
//*            SUPPORTED BY THIS JCL PROCEDURE.
//*
//*******************************************************************
//DB9AWLM PROC RGN=0K,APPLENV=WLMENV,DB2SSN=DB9A,NUMTCB=40,MNSPAS=3
//IEFPROC EXEC PGM=DSNX9WLM,REGION=&RGN,TIME=NOLIMIT,
//             PARM='&DB2SSN,&NUMTCB,&APPLENV,&MNSPAS'
//STEPLIB  DD  DISP=SHR,DSN=DB9AU.RUNLIB.LOAD
//         DD  DISP=SHR,DSN=CEE.SCEERUN
//         DD  DISP=SHR,DSN=DB9A9.SDSNEXIT
//         DD  DISP=SHR,DSN=DB9A9.SDSNLOAD
```

3. Define the application environment to WLM by using one of the following methods.
   - Use the WLM administrative application (IWMARIN0) to define a new environment or copy or modify an existing one.
     a. In the first WLM panel, Definition Menu, enter 9 for Application Environments.

        The application environment definition panel is displayed.
     b. On the application environment definition panel, specify the appropriate action code:
        - Create a new WLM application environment.
        - Copy an existing WLM application environment.
        - Modify an existing WLM application environment.

        The appropriate panel for creating, copying or modifying the application environment is displayed.

c. If you are creating or modifying a WLM environment, specify the appropriate options for the WLM application environment definition, using the same application environment name and procedure name that you specified for the startup procedure.

d. After you specify the appropriate options, press Enter to create the application environment definition.

- Modify and run sample job DSNTIJRW. As provided, DSNTIJRW creates the core DB2 WLM environments, but you can modify it to define your own WLM environments.

4. Install the WLM service definition that contains information about the application environment by opening the WLM Definition Menu panel and selecting the **Install** function in the **Utilities** menu bar.
5. Activate the WLM service policy from the installed service definition by completing one of the following actions:
   - Issue the VARY WLM command as follows:

     ```
     VARY WLM,POLICY=policy-name
     ```

   - On the WLM Definition Menu panel, choose the **Activate service policy** option from **Utilities** menu bar.
6. Set up security for the WLM application environments and for creating stored procedures by completing the following actions:

   a. Optional: Specify which WLM-established address spaces can run stored procedures. If you do not complete this step, any WLM-established address space can run stored procedures.

   b. Grant access to users to create procedures in specific WLM address spaces.

   c. Grant access to users to create procedures in specific schemas. Use the GRANT statement with the CREATIN option for the appropriate schema.

   d. Grant access to users to create packages for procedures in specific collections. Use the GRANT statement with the CREATE option for the appropriate collection.

      Requirement: For native SQL procedures, the collection name for the package must be the same as the schema name.

   e. Grant access to refresh the WLM environments to the appropriate people.

   For more detail, see Managing authorization for stored procedures (Managing Security).
7. Ensure that the ID that is used to run the JCL startup procedure for the WLM application environment has permission to use RRSAF. Each time one of the DB2 WLM address spaces is started, it uses RRSAF to attach to DB2.

What to do next

When you create or alter a stored procedure, specify the WLM application environment that DB2 is to use in the WLM_ENVIRONMENT option of the CREATE or ALTER PROCEDURE statement. When you specify environments, be aware that:

- Routines that have low memory requirements and that are frequently called should not share an environment with routines that have high memory requirements and are rarely called. Otherwise, storage is wasted.
- Stored procedures can be grouped according to the authorization settings for a WLM environment.

- WLM environments should not be shared among routines that use the same DD name for different purposes.
- WLM environments should not be shared among routines if one routine allocates data sets in the environment and that routine is run only occasionally. If other routines that do not use the data sets use the environment, the data sets are kept artificially allocated, which might interfere with other tasks.

Related concepts:

"WLM-established stored procedures address spaces storage requirements" on page 48

Related tasks:

Setting up and managing Workload Manager (DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond)

Managing authorization for stored procedures (Managing Security)

Refreshing a WLM application environment for stored procedures (DB2 Administration Guide)

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

"DSNTWLMB tool" on page 847

GRANT (collection privileges) (DB2 SQL)

GRANT (schema privileges) (DB2 SQL)

Related information:

Using Operator Commands for Application Environments (z/OS MVS Planning: Workload Management)

Migration step 26: Set up DB2-supplied routines

Installation jobs are provided for the setup of all DB2-supplied routines and supporting objects that are provided with the DB2 base product.

Related tasks:

"Disabling DB2-supplied routines for validation" on page 851

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Setting up the DB2 core WLM environments during migration:

DB2 stored procedures (except native SQL procedures) must run in WLM-managed stored procedure address spaces. You must assign each stored procedure to a WLM application environment, which routes the stored procedure work to the appropriate WLM-managed address space.

About this task

For DB2-supplied routines, you can use the provided definitions for core WLM environments as a starting point for defining your environments. You can also manually define the WLM environments by using the IWMARIN0 application, as described in "Setting up a WLM application environment for stored procedures during installation" on page 543.

Exception: You do not need to complete this configuration task for native SQL procedures unless they call at least one external stored procedure or external SQL procedure. This task also applies to native SQL procedures that call at least one user-defined function or that are defined with ALLOW DEBUG MODE or DISALLOW DEBUG MODE. Otherwise, this task applies to only user-defined functions, external stored procedures, external SQL procedures, and DB2-supplied stored procedures

Procedure

To set up the provided core WLM environments:

Run job DSNTIJRW.
Job DSNTIJRW invokes the DSNTWLMB tool to create the core WLM environments. The CLIST assigns the core WLM environment names in DSNTIJRW according to the values that are specified on panel DSNTIPR1. For detailed information about the environments that are created by DSNTIJRW, see "Core WLM environments for DB2-supplied routines" on page 840.

Related concepts:

"WLM management of stored procedures" on page 839

Related reference:

"DSNTWLMB tool" on page 847

"Parameters for DB2-supplied routines panel: DSNTIPR1" on page 458

Installing DB2-supplied routines during migration:

Two jobs install and validate the installation of DB2-supplied routines. These jobs are configured with the options that you specified on installation panel DSNTIPR1 and panels DSNTIPRA through DSNTIPRP.

Procedure

To install the DB2-supplied routines:

1. Run job DSNTIJRT to install and configure all DB2-supplied routines.
2. After DSNTIJRT successfully completes, run job DSNTIJRV to validate the installation of the routines.
3. From V9.1 If you used the SYSPROC.XSR_COMPLETE stored procedure in Version 9.1, then ensure that the WLM environment that is used for running XSR_COMPLETE is updated to use the Version 10 code paths. The following parameters in the Java Language Environment (JAVAENV) data set for the WLM environment are affected:
   - DB2_BASE
   - JCC_HOME
   - JAVA_HOME

   From V9.1
4. Authorize the appropriate users to use one or more of these stored procedures by issuing the GRANT EXECUTE statement.

   As part of the installation process, job DSNTIJRT defines, binds, and grants access to the stored procedures and user-defined functions. Job DSNTIJRT grants EXECUTE authority to the authorization IDs that are specified for the GRANT EXECUTE fields on installation panels DSNTIPRB through DSNTIPRP.

The default setting for GRANT EXECUTE is PUBLIC, meaning that anyone who can access DB2 can use these stored procedures.

Recommendation: When you run the installation CLIST, use the GRANT EXECUTE fields on panels DSNTIPRB through DSNTIPRP to specify different access lists.

Important: DSNAEXP can execute the EXPLAIN statement on any explainable SQL statement that is valid on the system, and the EXPLAIN output can reveal potentially sensitive information. The privileges to run DSNAEXP should therefore be granted with consideration. For example, you should avoid granting access for DSNAEXP to PUBLIC.

5. Complete additional setup steps for any of the following routines that you plan to use.
   - The stored procedures for XML schema support
   - The administrative task scheduler routines and administrative enablement routines
   - The CICS transaction invocation procedure
   - The DSNAEXP EXPLAIN stored procedure
   - The common SQL API stored procedures
   - The WebSphere MQ user-defined functions
   - The web service user-defined functions

What to do next

After installing the DB2-supplied routines, you can choose to disable any of the routines that are unnecessary or unusable on your subsystem.

Related concepts:

"Configuration control statements for DB2-supplied routines" on page 849

Related tasks:

Granting privileges for executing stored procedures and stored procedure packages (Managing Security)

"Disabling DB2-supplied routines for validation" on page 851

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Job DSNTIJRT:

Job DSNTIJRT installs and configures DB2-supplied routines by executing program DSNTRIN.

DSNTIJRT provides a *configuration control statement* for each DB2-supplied routine and the mode for processing those statements. The configuration control statement specifies the WLM environment, list of authorization IDs that are allowed to execute the routine, and optional package owner for the routine. DSNTRIN reads and validates those statements and uses them to create and customize routines and supporting objects.

Format

DSNTRIN accepts the following parameters:

DB2SSN
Identifies the DB2 subsystem on which to install and configure the DB2-supplied routines.

MODE
During SQL processing, DSNTRIN configures each routine with the WLM environment specified in the configuration control statement for that routine. Processing behavior depends on the DSNTRIN run mode, the DB2 catalog mode, and which objects already exist.

INSTALL
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the routine already exists, it is updated, if necessary, to use the WLM environment specified in the configuration control statement for that routine. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. If a supporting object is used by more than one routine, the object is created when one of those routines is first created.

REINSTALL
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. Existing routines and any related created global temporary tables are first dropped before they are re-created. Related databases are not dropped if they already exist, but they are created if they do not exist. If a supporting object is used by more than one routine, the object is created when one of those routines is first created. Drop processing of routines and related objects that do not exist is bypassed.

REINITIALIZE
Create, bind, and grant access to the DB2-supplied routine according to the WLMENV and GRANTTO parameters of the configuration control statement. If the package for a routine already exists, it is not rebound unless the DBRM has been updated or the package has been invalidated. Create, initialize, and grant access to any related objects required by the routine, such as a database or created global temporary table. Existing routines, existing related created global temporary tables, and existing related databases are first dropped before they are re-created. After each SQL statement, a COMMIT statement is issued. If a supporting object is used by more than one routine, the object is created when one of those routines is first created.

INSTALL-PREVIEW
Simulate INSTALL mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

REINSTALL-PREVIEW
Simulate REINSTALL mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

REINITIALIZE-PREVIEW
Simulate REINITIALIZE mode without updating the system. Generated JCL with embedded DDL and bind statements is outputted to the JCLOUT DD statement.

AUTHID
DSNTRIN uses this value as the CURRENT SQLID for creating and configuring most DB2-supplied routines. DSNTRIN also uses this value as the default OWNER for binding packages for the DB2-supplied routines that do not have the PKGOWNER specified in the configuration control statement.

SECDEFID
DSNTRIN uses this value as the CURRENT SQLID setting when creating and configuring DB2-supplied routines that are defined with the SECURITY DEFINER option.

Input

DSNTRIN allocates the following DD statements for input:

DBRMLIB
Location of the DBRMs for DB2-supplied routines. This is typically the *prefix*.SDSNDBRM target library.

SYSUT1
Internal work file for the DB2 installation bind interface (DSNTRBI).

CFIGIN
Location of the configuration control statements for DB2-supplied routines.

DB2OPT
This required DD can contain the following optional parameters:

STOGROUP
Specifies a storage group to use instead of the default SYSDEFLT storage group.

BP4K Specifies an alternate 4 K buffer pool for base table spaces that are created by DSNTRIN. The default 4 K buffer pool is BP0.

BP8K Specifies an alternate 8 K buffer pool for base table spaces that are created by DSNTRIN. The default 8 K buffer pool is BP8K0.

BP16K
Specifies an alternate 16 K buffer pool for base table spaces that are created by DSNTRIN. The default 16 K buffer pool is BP16K0.

BP32K
Specifies an alternate 32 K buffer pool for base table spaces that are created by DSNTRIN. The default 32 K buffer pool is BP32K.

LOBBP8K
Specifies an alternate 8 K buffer pool for LOB table spaces that are created by DSNTRIN. The default 8 K buffer pool is BP8K0.

LOBBP16K
Specifies an alternate 16 K buffer pool for LOB table spaces that are created by DSNTRIN. The default 16 K buffer pool is BP16K0.

Note: DSNTRIN does not create or activate this storage group or any of these buffer pools.

Output

DSNTRIN allocates the following DD statements for output:

SYSPRINT
Destination for echo of configuration control statements that are read and for messages that are generated during processing. Each processed routine generates at least one status message.

SYSTSPRT
Internal message file for the DB2 installation bind interface (DSNTRBI).

CFIGOUT
Destination for configuration control statements that are read by DSNTRIN. This output is useful for diagnostic purposes when DSNTRIN reports an error in a configuration control statement.

SQLOUT
Destination for DDL statements that are generated and processed by DSNTRIN. These statements are useful for diagnostic purposes.

BINDOUT
Destination for bind statements that are generated and processed by DSNTRIN. These statements are useful for diagnostic purposes.

JCLOUT
Destination for a generated sample job that is generated when one of the PREVIEW modes is specified. The job contains steps that process inline DDL and bind statements that are needed to install and configure missing, downlevel, or incorrectly configured DB2-supplied routines. This DD is required if you specify a PREVIEW mode.

Example of executing DSNTRIN in INSTALL mode:

```
//DSNTRIN  EXEC PGM=DSNTRIN
//         PARM=('DB2SSN(VA1A) MODE(INSTALL) AUTHID(AUTHID) SECDEFI-
           D(ADMF001)')
//DBRMLIB  DD DISP=SHR,DSN=DSN!!0.SDSNDBRM
//SYSUT1   DD UNIT=SYSDA,SPACE=(27930,(10,5)),
//            DCB=(RECFM=FB,LRECL=133)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//CFIGOUT  DD SYSOUT=*
//SQLOUT   DD SYSOUT=*
//BINDOUT  DD SYSOUT=*
//DB2OPT   DD *
 STOGROUP(DB2_ROUTINES_GROUP)
 BP4K(BP0)
 BP8K(BP8K0)
 BP16K(BP16K0)
 BP32K(BP32K)
 LOBBP8K(BP8K0)
 LOBBP16K(BP16K0)
/*
//CFIGIN   DD *
  ...
SYSPROC.DSNWSPM
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)

SYSPROC.DSNWZP
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)
  ...
//*
```

Example of executing DSNTRIN in INSTALL-PREVIEW mode:

```
//DSNTRIN  EXEC PGM=DSNTRIN
//         PARM='DB2SSN(VA1A) MODE(INSTALL-PREVIEW) AUTHID(AUTHID)'
//DBRMLIB  DD DISP=SHR,DSN=DSN!!0.SDSNDBRM
//SYSUT1   DD UNIT=SYSDA,SPACE=(27930,(10,5)),
//            DCB=(RECFM=FB,LRECL=133)
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//CFIGOUT  DD SYSOUT=*
//SQLOUT   DD SYSOUT=*
//BINDOUT  DD SYSOUT=*
//JCLOUT   DD DSN=USER.JCLLIB(DB2INST),DISP=SHR
//DB2OPT   DD *
 STOGROUP(DB2_ROUTINES_GROUP)
 BP4K(BP0)
 BP8K(BP8K0)
 BP16K(BP16K0)
 BP32K(BP32K)
 LOBBP8K(BP8K0)
 LOBBP16K(BP16K0)
/*
//CFIGIN   DD *
  ...
SYSPROC.DSNWSPM
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)

SYSPROC.DSNWZP
WLMENV(DSNWLM_NUMTCB1)
GRANTTO(PUBLIC)
  ...
//*
```

Related concepts:

"Configuration control statements for DB2-supplied routines" on page 849

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Job DSNTIJRV:

Job DSNTIJRV validates the DB2-supplied routines by executing program DSNTRVFY. You should validate these routines after DB2 installation and after each phase of migration.

DSNTRVFY validates a single routine, multiple routines, or all DB2-supplied routines. DSNTRVFY produces a report that identifies routines that passed the validation and possible problem areas for routines that failed.

Format

DSNTRVFY accepts the following required parameters:

DB2SSN(*ssid*)
Identifies which DB2 subsystem to connect to for validating the DB2-supplied routines.

ROUTINE(*name*)
Identifies which DB2-supplied routine needs to be validated. The following values are supported for *name*:

- A routine name in the form of *schema-name.routine-specific-name*.

- The keyword **DD:SYSIN** to specify a list of routines. List the routines in the form of *schema-name.routine-specific-name* under the SYSIN DD statement with one routine per line.
- The keyword **ALL** to specify all DB2-supplied routines.

DSNTRVFY also accepts the following optional parameter:

AUTHID(*authorization ID*)
DSNTRVFY uses this value as the OWNER for BIND PACKAGE and BIND PLAN statements and as the CURRENT SQLID for issued SQL statements. If this value is not specified, the primary authorization ID is used for BIND PACKAGE, BIND PLAN, and issued SQL statements.

Output

DSNTRVFY allocates the following required DD statements for output:

SYSPRINT
Contains messages generated during verification processing. Each processed routine can generate one or more messages.

SYSTSPRT
DSNTRVFY writes bind output to SYSTSPRT if an error occurs during bind processing.

The following JCL example is for validating a single routine:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(SYSPROC.DSNUTILS)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
```

The following JCL example is for validating multiple routines:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(DD:SYSIN)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  SYSPROC.DSNWZP
  SYSPROC.DSNACCOR
  SYSPROC.DSNUTILS
  SYSPROC.DSNUTILU
```

The following JCL example is for validating all routines:

```
//DSNTJVFY EXEC PGM=DSNTRVFY,
//         PARM='DB2SSN(VA1A) ROUTINE(ALL) AUTHID(FVTGRP2)'
//DBRMLIB  DD DSN=DB2A.SDSNDBRM,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(32000,(30,30)),DCB=(RECFM=VB,LRECL=133)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
```

Processing

DSNTRVFY completes the following validation:

- Verifies that the routine is defined to the SYSIBM.SYSROUTINES catalog.

- Verifies that the collection ID defined in the SYSIBM.SYSROUTINES catalog table matches the collection ID used to bind the package.
- Verifies that the routine is currently defined with the DB2-reserved WLM application environment name DSNWLM_NOT_ENABLED, indicating that it is not enabled to run on the current DB2 subsystem. If it is, then a warning message is issued and the validation is bypassed.
- Verifies if the routine can be executed on the current DB2 catalog mode.
- Verifies the APF authorization requirements.
- Verifies if the WLM application environment is defined and available for use.
- Verifies if the NUMTCB setting is valid or within the recommended range.
- Verifies if the required DD statements for some of the routines are defined.
- Verifies if the WLM environment assigned to Java routines is enabled for Java.
- Verifies security coexistence requirements (for example, some routines should run on only their own WLM application environment).
- Verifies that the selected routine can be called successfully. Certain routines such as DSNACICS, DSNAIMS, MQSeries functions, and most debugger routines are restricted to basic validation and are not called.

If the validation is successful, the following message is issued:

```
DSNT027I DSNTRVFY VALIDATION FOR schema.routine-specific-name SUCCESSFUL
```

If the validation fails, messages are issued to explain the reasons for failure. Examine these messages and make necessary adjustments before running the validation job again.

Input

DSNTRVFY requires the following DD statements for input:

DBRMLIB
Identifies the library where the DBRM for the DSNTRVFY program is located. This is typically the *prefix*.SDSNDBRM target library.

SYSIN
Contains the list of DB2 routines. This is required when the validation type is for multiple routines (DD:SYSIN).

SYSUT1
Work data set required for bind processing.

Limitations

The following limitations apply to the DSNTRVFY program:

- The name of the routine in the form *schema-name.routine-specific-name* is limited to 72 characters, the maximum line length in JCL.
- Only Version 10 DB2-supplied routines are supported for validation.
- Some routines, if enabled, receive basic validation to ensure that they are defined correctly and assigned to a valid WLM application environment. However, validation will not extend to calling these routines for one or more of the following reasons:
  - Optional system support such as CICS, IMS, or MQSeries is required.
  - External setup on a remote server or workstation is required.

– A call could introduce or modify data unexpectedly in a production environment.

The DB2-supplied routines that are restricted to basic validation are:

– DB2DEBUG.CREATE_SESSION
– DB2DEBUG.DESTROY_SESSION
– DB2DEBUG.GET_REPORT
– DB2DEBUG.LIST_SESSION
– DB2DEBUG.PUT_COMMAND
– SYSFUN.SYSTS_ENCRYPT
– SYSPROC.ADMIN_COMMAND_UNIX
– SYSPROC.ADMIN_JOB_CANCEL
– SYSPROC.ADMIN_JOB_FETCH
– SYSPROC.ADMIN_JOB_QUERY
– SYSPROC.ADMIN_JOB_SUBMIT
– SYSPROC.DBG_ENDSESSIONMANAGER
– SYSPROC.DBG_INITIALIZECLIENT
– SYSPROC.DBG_RECVCLIENTREPORTS
– SYSPROC.DBG_SENDCLIENTREQUESTS
– SYSPROC.DBG_SENDCLIENTCOMMANDS
– SYSPROC.DBG_TERMINATECLIENT
– SYSPROC.DSNACICS
– SYSPROC.DSNAIMS
– SYSPROC.DSNAIMS2
– SYSPROC.DSNLEUSR
– SYSPROC.SYSTS_ALTER
– SYSPROC.SYSTS_CREATE
– SYSPROC.SYSTS_DROP
– SYSPROC.SYSTS_RESTORE
– SYSPROC.SYSTS_START
– SYSPROC.SYSTS_STOP
– SYSPROC.SYSTS_TAKEOVER
– SYSPROC.SYSTS_UPDATE
– All MQSeries, web services, and SOAP functions (those with schema DB2MQ, DB2MQ1C, DB2MQ2C, or DB2XML)

Migration step 27: Verify views

During migration to Version 10 conversion mode (from Version 8 or Version 9.1), some views might be marked with view regeneration errors. If any views are marked with view regeneration errors, you can regenerate those views again.

Procedure

GUPI To verify views:

1. Issue the following query to determine which views were marked with view regeneration errors during migration:

```
SELECT CREATOR,NAME
FROM SYSIBM.SYSTABLES
WHERE TYPE = 'V'
AND STATUS = 'R'
AND TABLESTATUS = 'V'
```

2. If any views have view regeneration errors, issue the following ALTER VIEW statement:

   ```
   ALTER VIEW view REGENERATE
   ```

   where *view* is the name of the view with regeneration errors.

   You can use the DSNTIAUL program to generate the ALTER VIEW statement as follows:

```
// EXEC PGM=IKJEFT01
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(dsn)
    RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB10) +
      LIB('prefix.RUNLIB.LOAD') PARMS('SQL')
  END
/*
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
  SELECT SUBSTR(
    'ALTER VIEW '
    || STRIP(CREATOR) || '.' || NAME
    || ' REGENERATE ;'
    || '                                        '
    || '                                        '
    ,1,80)
    FROM SYSIBM.SYSTABLES
    WHERE TYPE = 'V'
    AND STATUS = 'R'
    AND TABLESTATUS = 'V'

  ;
/*
//SYSPUNCH DD  DUMMY
//SYSREC00 DD  DISP=(NEW,CATLG,DELETE),
//             DSN=&SYSUID.SQLTIAUL.ALTVIEWS,
//             UNIT=SYSDA,SPACE=(TRK,(750,75),RLSE)
```

GUPI

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 28: Take an image copy of the DB2 Version 10 conversion mode (from Version 8 or Version 9.1) catalog: DSNTIJIC

For backup purposes, create a copy of the DB2 for z/OS Version 10 conversion mode (from Version 8 or Version 9.1) catalog and directory.

Procedure

To create an image copy of the directory and catalog:

Run the Version 10 DSNTIJIC job.
COPY returns message DSNU1530I for each table space that has not yet been defined. These table spaces will be defined as part of enabling-new-function mode (from Version 8 or Version 9.1).

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related information:

DSNU1530I (DB2 Messages)

Migration step 29: Set up the administrative task scheduler

You can use the administrative task scheduler to execute administrative tasks according to a time-based or event-based schedule.

Before you begin

You should have run job DSNTIJRA and job DSNTIJIN before you set up the administrative task scheduler.

About this task

Restriction: After migration to conversion mode (from Version 8 or Version 9.1), the administrative task scheduler is disabled until you run job DSNTIJRT.

The following figure shows an overview of the steps for installing and configuring the administrative task scheduler.

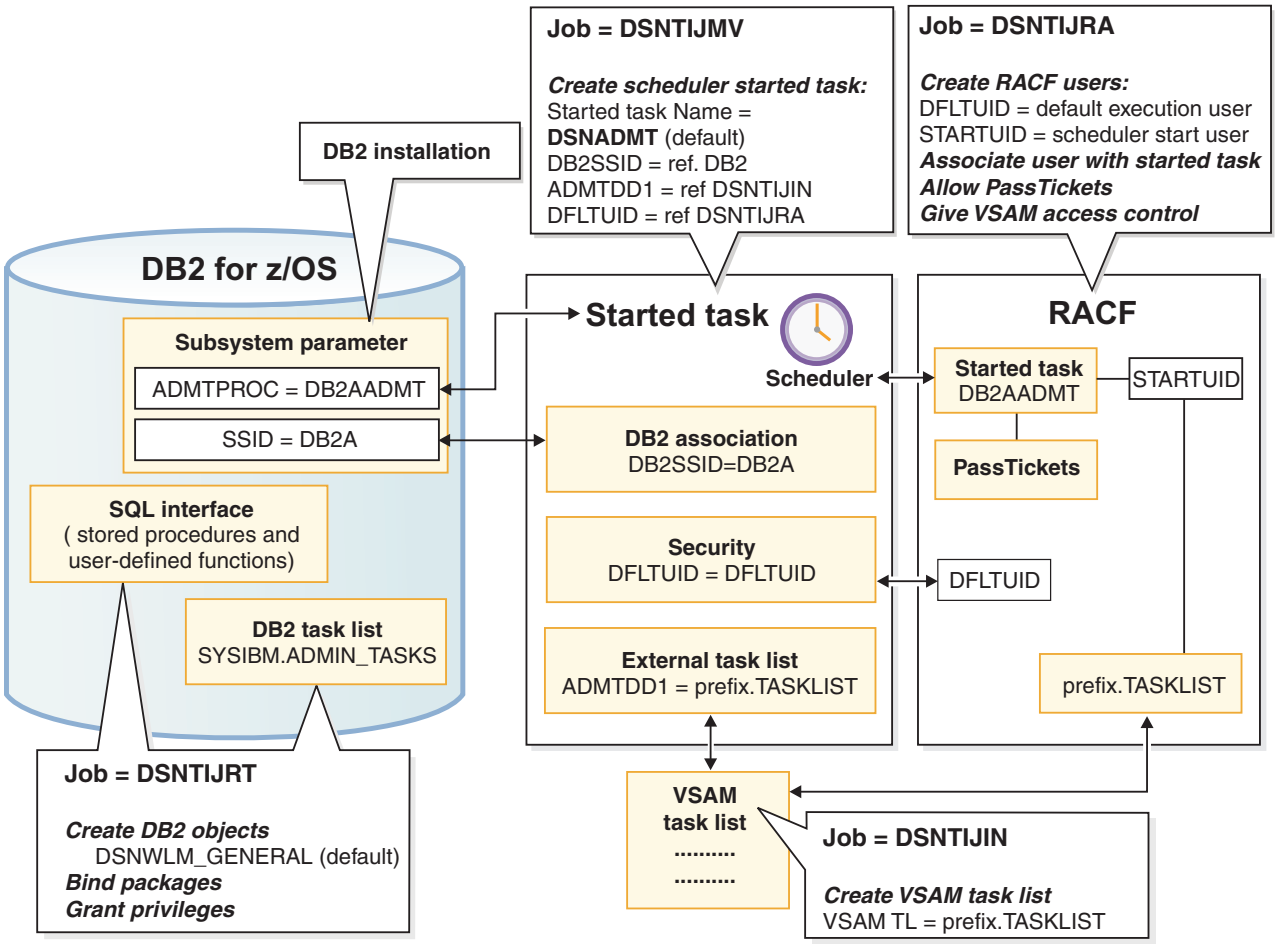


Figure 75. Setting up the administrative task scheduler

Procedure

To set up the administrative task scheduler:

1. Customize the administrative task scheduler address space startup procedure (xxxxADMT) in job DSNTIJMV. The name of the startup procedure needs to match the value of the ADMTPROC subsystem parameter. Before using the startup procedure, locate and review the settings for the following parameters:

   DB2SSID
   The name of this DB2 subsystem.

   DFLTUID
   The default ID that is used by the administrative task scheduler to execute its tasks. DFLTUID must be different than the ID that is used to start this address space.

   TRACE
   Indicates whether to activate tracing for the administrative task scheduler. The value can be ON or the default value of OFF.

   You can specify other parameters, such as ERRFREQ, MAXHIST, MAXTHD, or STOPONDB2STOP in the startup procedure similarly to the three parameters DB2SSID, DFLTUID and TRACE.

   ERRFREQ
   Specifies how frequently message DSNA679I displays on the console,

indicating that one of the redundant, active copies of the task list is not accessible. By default, this message displays on the console once per minute, for example, when DB2 is offline.

MAXHIST
Specifies the number of status entries per task that are stored. This parameter is a positive integer with a default value of 10. When the limit is reached, the oldest status entries are deleted.

MAXTHD
Specifies the maximum number of execution threads for an administrative task scheduler. The default value is 99.

STOPONDB2STOP
Specifies that the administrative task scheduler will terminate when DB2 is stopped.

Also locate and review the ADMTDD1 DD statement. The VSAM data set that is indicated must match the VSAM task list data set that is defined in job DSNTIJIN.

The following example shows how these parameters can be updated.

```
//DSNADMT  PROC LIB='DSN!!0.SDSNLOAD',
//          DB2SSID=DSN,
//          DFLTUID=DFLTUID,
//          TRACE=OFF,
//          MAXTHD=10
//*
//STARTADM EXEC PGM=DSNADMT0,DYNAMNBR=100,REGION=0K,
//          PARM=('DB2SSID=&DB2SSID',
//          ' DFLTUID=&DFLTUID',
//          ' TRACE=&TRACE'
//          ' MAXTHD=&MAXTHD'
//          ' ERRFREQ=1440'
//          ' STOPONDB2STOP')
//STEPLIB  DD DISP=SHR,DSN=&LIB
//ADMTDD1  DD DISP=SHR,DSN=DSNC!!0.TASKLIST
```

2. Ensure that the administrative task scheduler routines were enabled. They were enabled by job DSNTIJRT.
3. Create a job to make image copies of the administrative task scheduler table spaces. Establish a schedule for making image copies. In general, you should back up important databases on a regular basis. The database for the administrative task scheduler is no exception and should be copied on the same frequency as the DB2 catalog and directory. The following example shows how to copy the table spaces in this database to stacked data sets on tape with a retention period of 99 days:

```
//*
//DSNTIC  EXEC PGM=DSNUTILB,PARM='DSN,IMAGCOPY',COND=(4,LT)
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//DSNTRACE DD  SYSOUT=*
//SYSLISTD DD  *
  LISTDEF DSNLDEF
          INCLUDE TABLESPACES DATABASE DSNADMDB
//SYSIN    DD  *

  TEMPLATE DSNTPLT
           DSN(prefix.IMAGCOPY.&DB..;&TS.;)
           DISP (NEW,KEEP,DELETE)
           UNIT TAPE
           STACK YES
```

```
          RETPD 99
     COPY LIST DSNLDEF
          COPYDDN(DSNTPLT)
/*
```

Related concepts:

"Administrative task scheduler address space storage requirements" on page 48

Multi-threading in the administrative task scheduler (DB2 Administration Guide)

Related tasks:

"Additional steps for enabling the administrative task scheduler and administrative enablement routines" on page 854

Scheduling administrative tasks (DB2 Administration Guide)

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Related information:

DSNA677I (DB2 Messages)

Migration step 30: Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system (optional)

Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system by running some of the sample jobs from your previous DB2 release.

About this task

If all of the local DB2 objects from your Version 8 or Version 9.1 subsystem still exist (that is, if you have not run job DSNTEJ0), follow this procedure.

Procedure

To verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system:

1. Change the JOBLIB statements to point to *prefix*.SDSNLOAD.
2. Ensure that the DSN8EAE1 module that you created when you originally ran the Version 8 or Version 9.1 sample jobs is copied to *prefix*.SDSNEXIT. DSN8EAE1 is an EDITPROC that is used by the employee sample table.
3. Edit the Version 8 or Version 9.1 sample jobs before running them. Do **not** run all the sample jobs. Run only the specific jobs and job steps that are listed in the following steps.
4. Test the migration of the IVP phase 2 applications from your previous version:
   a. DSNTEJ2A: Perform all except the first two steps of job DSNTEJ2A. Expect a return code of 4 because table spaces DSN8D910.NEWDEPT and DSN8D910.NEWPHONE are placed in COPY-pending states.
   b. DSNTEJ2C: Issue only the RUN PROGRAM(DSN8BC3) PLAN(DSN8BH81) statement in step PH02CS04.
   c. DSNTEJ2D: Issue only the RUN PROGRAM(DSN8BD3) PLAN(DSN8BD81) statement in step PH02DS03.

d. DSNTEJ2E: Issue only the RUN PROGRAM(DSN8BE3) PLAN(DSN8BE81) statement in step PH02ES04.
e. DSNTEJ2F: Issue only the RUN PROGRAM(DSN8BF3) PLAN(DSN8BF81) statement in step PH02FS03.
f. DSNTEJ2P: Run only step PH02PS05.

5. Test the migration of the IVP phase 3 applications from your previous version:
   a. Do not run job DSNTEJ3C or DSNTEJ3P.
   b. If you want to test the DB2 Version 8 or Version 9.1 ISPF-CAF applications under Version 10, place the Version 8 or Version 9.1 SDSNSPFP panel library ahead of the Version 10 SDSNSPFP panel library in the ISPPLIB concatenation. This placement is necessary so that the plans that are migrated from your previous version can be used. Remove the Version 8 or Version 9.1 SDSNSPFP library from your ISPPLIB concatenation when you are finished testing the Version 8 or Version 9.1 IVP applications under Version 10.

What to do next

Do not run any other Version 8 or Version 9.1 sample jobs.

Related concepts:

"Dynamic SQL and the ISPF/CAF application" on page 766

Related tasks:

"Ensure that sample objects from your previous release are available" on page 578

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Migration step 31: Collect improved statistics (optional)

When migration to Version 10 conversion mode (from Version 8 or Version 9.1) is complete, you should run RUNSTATS before rebinding your application packages to collect improved statistics.

About this task

Running RUNSTATS before rebinding packages or executing dynamic SQL statements isolates the opportunity for Version 10 access path changes to the first rebind only.

Rebinding packages before running RUNSTATS introduces the Version 10 optimizer to statistics from the previous version. Therefore, if you do not run RUNSTATS before rebinding packages, the Version 10 optimizer is not introduced with Version 10 statistics until the next rebind after running RUNSTATS.

Procedure

To collect improved DB2 statistics:

Before rebinding your application packages, use the following guidelines:

- From V8 If you migrated from DB2 Version 8, run your standard RUNSTATS procedures against all table spaces.

This action is recommended before rebinding application packages because DB2 Version 9 introduced improved formulas for index statistics, including CLUSTERRATIOF. From V8

- If your standard Version 8 or Version 9 RUNSTATS procedures do not include the KEYCARD option of RUNSTATS, you should run your standard RUNSTATS procedures against all table spaces.

  This action is recommended because by default, DB2 Version 10 automatically collects the statistics that are collected when the KEYCARD option is specified.

Related tasks:

From V8 "Checklist for migration to DB2 Version 10 conversion mode from Version 8" on page 15 From V8

From V9.1 "Checklist for migration to DB2 Version 10 conversion mode from Version 9.1" on page 25 From V9.1

Falling back

Falling back is the process of returning to DB2 Version 8 or Version 9.1 after migrating your catalog and directory to DB2 Version 10 conversion mode (from Version 8 or Version 9.1).

Important restrictions:

- From V8 If you migrated from a Version 8 subsystem, you can only fall back to Version 8. A subsystem that has started migration to Version 10 and then fallen back to Version 8 cannot subsequently migrate to Version 9.1. From V8
- From V9.1 If you migrated from a Version 9.1 subsystem, you can only fall back to Version 9.1. From V9.1

You can fall back to a previous version only after successfully migrating the catalog to Version 10 conversion mode (from Version 8 or Version 9.1). However, you cannot fall back to a previous version or return to Version 10 conversion mode (from Version 8 or Version 9.1) after the subsystem enters Version 10 enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode.

Fall back if you have a severe error while operating Version 10 conversion mode (from Version 8 or Version 9.1) and you want to return to operation on your previous version. After fallback, the catalog remains a Version 10 catalog.

Remigrating is the process of returning to Version 10 conversion mode (from Version 8 or Version 9.1) after falling back.

Implications of falling back

Before you fall back to your previous DB2 version, consider the implications of doing so. For example, you cannot use the new DB2 Version 10 for z/OS facilities until you are in Version 10 new-function mode.

Data sharing

If any member of a data sharing group needs to falls back, consider the implications for data sharing. For more information about falling back in a data sharing environment, see "Falling back in a data sharing environment" on page 734.

Frozen objects

Falling back does not undo changes that the migration process made to the catalog. DB2 uses the migrated catalog after fallback. Some objects in this catalog that have been affected by Version 10 function might become *frozen* objects after fallback. Frozen objects are unavailable, and they are marked with the release dependency marker O. If an object is marked with a release dependency, it remains marked forever. The release dependency marker is listed in the IBMREQD column of catalog tables. The following table lists the objects that are frozen when you fall back to a previous release.

Table 88. Objects that are frozen after fallback

RELEASE DEPENDENT MARK = O
• Plans, packages, or views that use any new syntax, objects, or bind options • DBRMs that are produced by a precompile in Version 10 with a value of V10 for the NEWFUN option

Plans and packages become frozen objects when they use new SQL syntax, use new BIND options and attributes, or reference frozen objects. When plans and packages become frozen objects, the automatic rebind process is adversely affected.

GUPI After falling back, you can determine if any of your objects are frozen by issuing the following statement:

```
SELECT * FROM catalog table
WHERE IBMREQD = 'O'
```

GUPI

SQL syntax

When you fall back to Version 8 or Version 9.1, you cannot use Version 10 SQL syntax. In Version 10, you can run the DB2 precompiler or DB2 coprocessor on your applications with option NEWFUN(V8) to determine whether they contain syntax that is newer than Version 8. You can run the DB2 precompiler or DB2 coprocessor on your applications with option NEWFUN(V9) to determine whether they contain syntax that is newer than Version 9.1.

Automatic rebind

After fallback, if you specified YES or COEXIST for the ABIND subsystem parameter, plans or packages that are bound in Version 10 are automatically rebound on their first execution in Version 8 or Version 9.1. If you specified NO for the ABIND subsystem parameter, this automatic rebind fails.

The automatic rebind also fails if you try to use plans or packages that are frozen. To make available the plans and packages that are not automatically rebound, change the SQL statements or remove the reference to a frozen object, precompile the application programs, and explicitly bind the plans and packages on Version 8 or Version 9.1.

After fallback, you need to rebind SPUFI. You might need to bind SPUFI with ENCODING(EBCDIC) to avoid an error message.

Administrative task scheduler tasks

After fallback, all tasks that are managed by the administrative task scheduler in DB2 Version 10 are not manageable by the administrative task scheduler in Version 8 or Version 9.1 because of the higher task version. Use ADMIN_TASK_ADD in Version 8 or Version 9.1 to delete and reinsert all tasks. You can use the ADMIN_TASK_LIST table user-defined function to read and store the characteristics of the tasks in a user table before you fall back.

Database metadata stored procedures

In DB2 Version 10, the DB2 database metadata stored procedures have new Unicode load modules.

Table 89. Database metadata stored procedures and corresponding Unicode load modules

Stored procedure name	Unicode load module
SYSIBM.SQLCOLPRIVILEGES	DSNACPRU
SYSIBM.SQLCOLUMNS	DSNACOLU
SYSIBM.SQLFOREIGNKEYS	DSNAFNKU
SYSIBM.SQLFUNCTIONCOLS	DSNAFCOU
SYSIBM.SQLFUNCTIONS	DSNAFUNU
SYSIBM.SQLGETTYPEINFO	DSNATYPU
SYSIBM.SQLPRIMARYKEYS	DSNAPRKU
SYSIBM.SQLPROCEDURECOLS	DSNAPCOU
SYSIBM.SQLPROCEDURES	DSNAPRCU
SYSIBM.SQLSPECIALCOLUMNS	DSNASPCU
SYSIBM.SQLSTATISTICS	DSNASTAU
SYSIBM.SQLTABLEPRIVILEGES	DSNATBPU
SYSIBM.SQLTABLES	DSNATBLU
SYSIBM.SQLUDTS	DSNAUDTU
DSNACLRU DLL (for remote procedure)	DSNACLRU

When you run job DSNTIJRT after migrating to conversion mode, it drops the EBCDIC-encoded versions of these stored procedures, and re-creates them as Unicode-encoded. Therefore, if you fall back after running job DSNTIJRT and you use the DB2 database metadata stored procedures, you need to make the Version 10 load modules available to this DB2 subsystem and other members of the data sharing group.

To make your Version 10 load modules available to this DB2 subsystem and other members of the data sharing group, complete one of the following tasks:

- Copy the Version 10 load modules that are listed above into the Version 8 or Version 9.1 *prefix*.SDSNLOAD libraries.
- Include the Version 10 *prefix*.SDSNLOAD library in the STEPLIB of the WLM startup procedure for the database metadata stored procedures.

Other implications of falling back

Before you fall back to Version 8 or Version 9.1, be aware of the following considerations:

Reactive governing
If you decreased ASUTIME values in the resource limit facility to account for the fact that the resource limit facility does not accumulate ASUTIME across SQL statement boundaries in DB2 Version 10, you need to increase the ASUTIME values to avoid getting -905 errors.

TEMP database
From V8 If you dropped the TEMP database in conversion mode from Version 8, you need to re-create the TEMP database. Otherwise, applications that use the TEMP database fail in Version 8. From V8

Private protocol
In Version 10, if you explicitly rebind a package or plan that was bound with DBPROTOCOL(PRIVATE) in a previous version of DB2, the object is rebound with the DBPROTOCOL(DRDA) bind option. If you fall back from Version 10, the package or plan retains the DBPROTOCOL(DRDA) setting. If you want to run the object with private protocol in the fallback release, you must explicitly rebind it with DBPROTOCOL(PRIVATE).

Subgroup attachment
Subgroup attachment can be used after fallback only if you used the Version 10 early code when you defined the DB2 subsystem to z/OS.

Related concepts:
"Falling back and remigrating in a data sharing environment" on page 734

Automatic rebinding (DB2 Application programming and SQL)

Fallback procedure

Because the structure of the DB2 Version 10 for z/OS catalog is used after you fall back to a previous version, the fallback procedure involves only a few steps.

Before you begin

To fall back, your subsystem must be in Version 10 conversion mode (from Version 8 or Version 9.1).

About this task

From V8 If you migrated from a Version 8 subsystem, you can only fall back to Version 8. A subsystem that has started migration to Version 10 and then fallen back to Version 8 cannot migrate to Version 9.1. From V8

From V9.1 If you migrated from a Version 9.1 subsystem, you can only fall back to Version 9.1. From V9.1

What to do next

You can save your Version 10 TSO logon procedures and JCL for remigration to Version 10.

Fallback step 1: Stop DB2 Version 10 activity:

Before you can fall back, you must stop DB2 Version 10 activity and ensure that no recovery is required on system databases.

Procedure

To stop Version 10 work:

1. Issue the following command:

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI

   The QUIESCE keyword allows DB2 to complete processing of currently executing programs. This activity might require some processing time.
2. Issue the following command:

   GUPI

   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```

   GUPI

   This command allows only the install-defined system administrators and system operators to access DB2.

   If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Check to see that you started DB2 with the correct subsystem parameter load module.
3. Ensure that all work is complete.
   - Ensure that no units of recovery remain. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```

     GUPI Then use RECOVER INDOUBT for any indoubt threads.
   - Ensure that no utility work remains. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY UTILITY(*)
     ```

     GUPI Then, either allow utilities to complete before proceeding, or stop all utility processing with the following command:

     ```
     -DSN1 TERM UTILITY(*)
     ```
   - Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
     -DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
     ```

     GUPI A user with install-defined system administrator or system operator authority also must enter this command.

     Recover any table spaces and index spaces with write error range or deferred restart states.
4. To stop DB2, issue the following command:

GUPI

```
-DSN1 STOP DB2 MODE(QUIESCE)
```

GUPI A user with SYSADM or SYSOPR authority also must enter this command.

If IRLM does not stop automatically when DB2 stops, stop IRLM manually. To stop IRLM, issue the following command, where *irlmproc* is the name you assigned to the IRLM startup procedure:

```
STOP irlmproc
```

Fallback step 2: Reactivate DB2 Version 8 or Version 9.1 code: DSNTIJFV:

After you stop your DB2 Version 10 activity, continue the fallback process by using job DSNTIJFV to reactivate the code of your previous DB2 version.

About this task

Job DSNTIJFV renames procedures to activate Version 8 or Version 9.1 and deactivate DB2 Version 10.

Procedure

To reactivate the code of your previous DB2 version:

1. (Optional) Tailor job DSNTIJFV.

   SYS1.PROCLIB is the default target for JCL procedures. Add statements to rename other procedures, such as your IMS, CICS, TSO logon procedures, and batch procedures. You might also need to rename procedures in your jobs from Version 8 or Version 9.1.

   You might want two sets of procedures, such as DSN1*xxxx* and DSN2*xxxx*, at all times, with an alias for the current release level.
2. Run job DSNTIJFV.
3. Ensure that job DSNTIJFV completed successfully.
   - If DSNTIJFV runs successfully, it produces a return code of 0. Check that all renames executed successfully.
   - If DSNTIJFV fails or abends, rerun only the renames that failed. If some of the procedures already exist, check carefully to ensure that procedures for the two releases are not mixed.

Fallback step 3: Reconnect TSO, IMS, and CICS to DB2 Version 8 or Version 9.1:

After you reactivate your DB2 Version 8 or Version 9.1 code during fallback, reconnect TSO, IMS, and CICS to DB2.

Procedure

To reconnect TSO, IMS, and CICS to DB2:

1. Re-establish your Version 8 or Version 9.1 TSO logon procedures and JCL.
2. For CICS connections, complete one of the following actions:
   - If you overwrote the load module during migration to DB2 Version 10, reassemble the resource control table (RCT) with the Version 8 or Version 9.1 libraries.

- If you did not overwrite the load module, change the STEPLIB statements in DB2 CICS jobs so that they refer to the Version 8 or Version 9.1 libraries.

3. For IMS connections, change the STEPLIB statements in DB2 IMS jobs and address spaces so that they refer to the Version 8 or Version 9.1 libraries.

Fallback step 4: Start DB2 Version 8 or Version 9.1:

During fallback, you can start Version 8 or Version 9.1 after you re-establish your Version 8 or Version 9.1 TSO logon procedures, CICS connections, and IMS connections. You can start Version 8 only if you migrated to Version 10 from Version 8. You can start Version 9.1 only if you migrated from Version 9.1.

Procedure

To start Version 8 or Version 9.1:

1. Start the IRLM. If you have not requested that DB2 automatically start the IRLM, start it before you start DB2. Use the following command, where *irlmproc* is the name that you assigned to the IRLM startup procedure:

   ```
   START irlmproc
   ```

   This is the value that you specified for the PROC NAME option on installation panel DSNTIPI.

   If you specified YES for the AUTO START option on installation panel DSNTIPI, DB2 starts the IRLM automatically.

2. GUPI Start DB2 from the z/OS console by using the following command:

   ```
   -DSN1 START DB2,PARM(DSNZPxxx)
   ```

   In this command, *-DSN1* is the subsystem command prefix that you defined for DB2, and *DSNZPxxx* is the name of the Version 8 or Version 9.1 subsystem parameter module. If you used the default name, DSNZPARM, you can omit the PARM parameter.

   If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, and possibly *ssnm*DIST and *irlmproc*, where *ssnm* is the DB2 subsystem name and *irlmproc* is the IRLM procedure name.

   If DB2 starts successfully, the series of restart messages that you receive concludes with these two messages:

   ```
   DSNR002I   RESTART COMPLETED
   DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
   ```

   GUPI

3. Complete one of the following actions:

Option	Description
If you have done distributed processing with your DB2 Version 10 subsystem	GUPI Check message DSNR005I for the number of indoubt threads after you start DB2. If you find no indoubt threads, continue falling back as if you had not done any distributed processing. If you find indoubt threads, issue the following command: `-DSN1 DISPLAY THREAD(*) TYPE(INDOUBT)` If the number of indoubt threads that are reported in the DSNV408I messages is equal to the number of threads that are reported in the DSNR005I message, continue falling back as if you had not done any distributed processing. If fewer indoubt threads are reported by DSNV408I messages than in message DSNR005I, proceed as follows: 1. Stop your Version 8 or Version 9.1 subsystem. 2. Determine which units of work are incomplete by scanning the DB2 recovery log with the DB2 Version 10 DSN1LOGP utility. Use the SUMMARY option of this utility. 3. Examine the DSN1LOGP output to find all the DSN1162I messages that have a COORDINATOR name in a remote location. Each of these messages identify an indoubt DBAT. Record the LUWID that is displayed in each message. 4. Decide whether to commit or abort each indoubt DBAT. One way to do this is by contacting the COORDINATOR location. If it is another DB2 subsystem, use the DISPLAY THREAD command to help you decide. 5. If you have not already done so during migration, apply the fallback PTF that is supplied with Version 10. 6. Start your Version 8 or Version 9.1 subsystem again. 7. Issue the RECOVER INDOUBT ACTION(*correct decision*) LUWID(*luwid*) command to resolve each indoubt DBAT. GUPI
If you have not done distributed processing with your DB2 for z/OS Version 10 subsystem	Check outstanding restrictions after you start DB2. Identify databases whose uses are restricted by issuing the following command: GUPI `-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT` GUPI You can start some of these databases at this time.

4. If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. A common cause of startup failure is that the BSDS does not match the subsystem parameter values; ensure that the startup procedure is pointing to the correct BSDS and subsystem parameter. Also, check that the subsystem parameter member that you specified (or is used by default) when you started DB2 is the one that job DSNTIJUZ built. Check the JCL for the DB2 startup procedure.
5. Optional: Start TSO. If you want to use the TSO SUBMIT command to do housekeeping and fallback verification, you must start TSO (if it is not already started).

Related tasks:

Starting a table space or index space that has restrictions (DB2 Administration Guide)

Fallback step 5: Modify your DSNTIJIC job:

When you migrate to Version 10 conversion mode (from Version 8 or Version 9.1), new table spaces are added to the DB2 catalog. Therefore, when you fall back from Version 10, you need to modify your Version 8 or Version 9.1 image copy job (DSNTIJIC) to account for these new table spaces.

Procedure

To modify your DSNTIJIC job:

Complete one of the following tasks:

- From V8 If you migrated from Version 8, add the following table spaces to the Version 8 DSNTIJIC job:
  - DSNDB06.SYSCONTX
  - DSNDB06.SYSPLUXA
  - DSNDB06.SYSROLES
  - DSNDB06.SYSRTSTS
  - DSNDB06.SYSTARG
  - DSNDB06.SYSTSASC
  - DSNDB06.SYSTSUNI
  - DSNDB06.SYSXML

  Your Version 8 DSNTIJIC job is ready for use in fallback mode. From V8
- From V9.1 If you migrated from Version 9.1, add the following table spaces to the Version 9.1 DSNTIJIC job:
  - DSNDB06.SYSTSASC
  - DSNDB06.SYSTSUNI

  Your Version 9.1 DSNTIJIC job is ready for use in fallback mode. From V9.1

Fallback step 6: Verify fallback:

After you start DB2 Version 8 or Version 9.1, you must complete some of your own testing to determine if the fallback was successful.

About this task

You cannot run the DB2 Version 10 samples on Version 8 or Version 9.1.

Procedure

To verify fallback:

1. Test the IVP phase 2 applications:
   a. DSNTEJ2A: Run all, except the first two, steps of job DSNTEJ2A. Expect a return code of 4 because table spaces DSN8D910.NEWDEPT and DSN8D910.NEWPHONE are placed in COPY-pending states.

b. DSNTEJ2C: Execute only the RUN PROGRAM(DSN8BC3) PLAN(DSN8BH81) statement in step PH02CS04.
c. DSNTEJ2D: Execute only the RUN PROGRAM(DSN8BD3) PLAN(DSN8BD81) statement in step PH02DS03.
d. DSNTEJ2E: Execute only the RUN PROGRAM(DSN8BE3) PLAN(DSN8BE81) statement in step PH02ES04.
e. DSNTEJ2F: Execute only the RUN PROGRAM(DSN8BF3) PLAN(DSN8BF81) statement in step PH02FS03.
f. DSNTEJ2P: Run only step PH02PS05.

2. Test the IVP phase 3 applications. Do not run job DSNTEJ3C or DSNTEJ3P.
3. Run the sample applications of the DB2 version that you fell back to.
4. Test your own applications.
5. Retry the problem for which you decided to fall back.

Related tasks:

"Migration step 30: Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system (optional)" on page 636

Fallback step 7: Update your Version 9.1 WLM environment for XSR_COMPLETE:

If you use the SYSPROC.XSR_COMPLETE stored procedure in Version 9.1, then after falling back, you must update the Java Language Environment (JAVAENV) file of the WLM environment that you use for running XSR_COMPLETE.

Procedure

To update your Version 9.1 WLM environment for XSR_COMPLETE:

Ensure that the WLM environment that is used for running XSR_COMPLETE is updated to use the Version 9.1 code paths. The following parameters in the JAVAENV data set for the WLM environment are affected:

- DB2_BASE
- JCC_HOME
- JAVA_HOME

Remigrating

Remigrating (returning to DB2 Version 10 after falling back to Version 8 or Version 9.1) is simpler than the initial migration process.

Before you begin

Before you remigrate, refer to "Changes that might affect your migration from Version 8" on page 65 or "Changes that might affect your migration from Version 9.1" on page 114 because many of those considerations apply to remigrations, too. Whether each consideration applies to your subsystem depends on the type of activity that occurred on your Version 8 or Version 9.1 subsystem after fallback.

About this task

A plan or package is automatically rebound in Version 10 when it is executed for the first time after remigration if it was not explicitly bound in Version 8 or Version 9.1. However, if you specified NO for the ABIND subsystem parameter, automatic

binds are disabled. This means that the plan or package from your previous release is the one that runs in Version 10, so the plan or package does not benefit from Version 10 enhancements.

Procedure

To remigrate your subsystem to Version 10:

1. Check the integrity of your Version 8 or Version 9.1 catalog and directory:
   - Run the DSN1COPY utility with the CHECK option on your catalog table spaces.
   - Run the link checker utility, DSN1CHKR, to ensure that no existing links are broken.
   - Run the queries in member DSNTESQ of *prefix*.SDSNSAMP.
2. Optional: Ensure that your Version 8 or Version 9.1 DSNTIJIC job has been modified to copy new Version 10 catalog table spaces, and then use DSNTIJIC to take an image copy of your Version 8 or Version 9.1 catalog. This step is not required, but it is recommended.
3. Stop your Version 8 or Version 9.1 subsystem.
4. Reconnect TSO, IMS, and CICS to DB2 Version 10. Re-establish your Version 10 logon procedures and JCL, as well as your Version 10 CICS and IMS connections.
5. Rebuild Version 10 cataloged procedures. Rename the Version 10 procedures that were renamed by job DSNTIJFV during fallback. If job DSNTIJFV was not run, you need to rerun job DSNTIJMV. Comment out step 1 (DSNTIMP), which defines Version 10 to z/OS, and run the job. (You do not need to define Version 10 to z/OS a second time.)
6. Start DB2 Version 10. Ensure that you are using your Version 10 subsystem parameter load module.
7. Take an image copy of the Version 10 catalog by using the DSNTIJIC job.
8. Verify your DB2 Version 10 system.
9. **From V9.1** If you used the SYSPROC.XSR_COMPLETE stored procedure in Version 9.1, ensure that the WLM environment that is used for running XSR_COMPLETE is updated to use the Version 10 code paths. The following parameters in the Java Language Environment (JAVAENV) data set for the WLM environment are affected:
   - DB2_BASE
   - JCC_HOME
   - JAVA_HOME

   From V9.1

Related tasks:

"Migration step 2: Run the link checker on your DB2 table spaces (optional)" on page 580

"Migration step 5: Take image copies of the directory and catalog: DSNTIJIC" on page 583

"Migration step 30: Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system (optional)" on page 636

Migrating your DB2 subsystem to new-function mode

After you migrate your DB2 subsystem to conversion mode (from Version 8 or Version 9.1), you must complete the migration to Version 10 new-function mode by migrating the catalog.

About this task

The basic steps to migrate from conversion mode (from Version 8 or Version 9.1) to new-function mode are:

1. Migrate from conversion mode (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1).
2. Complete enabling-new-function mode (from Version 8 or Version 9.1) processing.
3. Migrate from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode.

Important: Once you begin enabling-new-function mode (from Version 8 or Version 9.1), you cannot fall back to the previous release. All members of a data sharing group must have successfully migrated to Version 10 conversion mode (from Version 8 or Version 9.1) before you begin the process to enable new-function mode.

A point of consistency needs to be created for the catalog and directory before you migrate to new-function mode. You should use the quiesce utility to establish a point of consistency for the catalog and directory table spaces. You should quiesce DSNDB01.SYSUTILX by itself. Updates to the DB2 catalog and directory should be avoided while the subsystem is in enabling-new-function mode (from both Version 8 and Version 9.1).

Generally, you must complete the following steps to convert the catalog:

1. Run the installation CLIST with the ENFM option. During enabling-new-function mode (from Version 8 or Version 9.1) processing, the CLIST generates installation jobs DSNTIJCS, DSNTIJEN, DSNTIJES, DSNTIJNF, DSNTIJNG, and DSNTIJNH, and customizes the Version 10 IVP jobs.

   From V8 If you migrated from Version 8, the CLIST also generates job DSNTIJNX. From V8
2. Create a copy of the DB2 Version 10 conversion mode (from Version 8 or Version 9.1) catalog and directory for backup purposes.
3. Run installation job DSNTIJEN.

   Important: Job DSNTIJEN converts DB2 to enabling-new-function mode (from Version 8 or Version 9.1), from which you cannot fall back. Do not run DSNTIJEN until you are certain that you will not need to fall back.

After job DSNTIJEN finishes, catalog migration is complete. You can then run job DSNTIJNF to move the subsystem to new-function mode.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Related tasks:

"Migration step 5: Take image copies of the directory and catalog: DSNTIJIC" on page 583

Chapter 4, "Tailoring DB2 jobs to your environment using the installation CLIST," on page 219

Implications of migrating to Version 10 new-function mode

As you prepare to migrate to Version 10 new-function mode, be aware that some changes in enabling-new-function mode (from both Version 8 and Version 9.1) and new-function mode might affect your migration.

The following changes apply to migration to new-function mode.

Archive logs must not be changed to use DFSMS extended format before migration to new-function mode: From V8 DB2 Version 10 supports the use of DFSMS striped or compressed extended format archive logs. If you are migrating from Version 8 and plan to use extended format archive logs, do not make this change until after migration to new-function mode. In a coexistence environment, you might encounter archive log read failures because DB2 Version 8 cannot read DFSMS extended format archive logs. Disaster recovery systems and all DB2 subsystems in a data sharing group must be in Version 10 new-function mode before you use extended format archive logs.

Change data capture cannot be enabled on catalog tables during enabling-new-function mode (from Version 8 or Version 9.1): During enabling-new-function mode (from both Version 8 and Version 9.1) processing, change data capture is disabled on the SYSPACKSTMT catalog table. You should not re-enable change data capture until your DB2 subsystem is in Version 10 new-function mode.

DB2 ignores the LOAD and REORG parameter KEEPDICTIONARY when tables are converted to reordered row format: By default, DB2 ignores the LOAD and REORG parameter KEEPDICTIONARY when tables are converted from basic row format to reordered row format in Version 10 new-function mode. If you want the KEEPDICTIONARY parameter to be honored, you need to set subsystem parameter HONOR_KEEPDICTIONARY to YES.

User-specified application defaults load module is allowed: In DB2 Version 10 new-function mode, you can specify an application defaults module to load instead of DSNHDECP. However, you should always have at least one module that has the name DSNHDECP because some applications might continue to have a dependency on the DSNHDECP name. You should not use this capability to specify your own application defaults load module in production until you have thoroughly tested the capability on a test system. To ensure fallback capability, you cannot modify applications to load an application defaults load module with a name other than DSNHDECP until your subsystem is in new-function mode.

Update to STMT_ID column of SYSIBM.SYSPACKSTMT table: After processing in enabling-new-function mode (from Version 8 or Version 9.1) is

complete, the STMT_ID column of the SYSIBM.SYSPACTSTMT catalog table is populated for existing rows in the table. The SYSIBM.SYSPACTSTMT table is not synchronized with the DB2 directory in SPT01 until you rebind the existing package.

Also, any packages for static SQL statements that existed prior to Version 10 must be rebound for any user who wants to utilize enhanced messages and traces that provide statement identifier information.

Changes to CREATE TABLE and ALTER TABLE statements for XML columns: When a table with XML columns is created by a CREATE TABLE or CREATE TABLE LIKE statement in Version 10 new-function mode, and the table is created in a universal table space, the XML columns and their associated XML tables are created in the XML versioning format. Prior to new-function mode, the XML columns and their associated XML tables are not created in the XML versioning format.

Also, when an ALTER TABLE ADD statement is used to add an XML column to a table in Version 10 new-function mode, and the table is created in a universal table space, the XML column and the associated XML table are created in the XML versioning format if it is the first XML column in the table or if all the XML columns in the table are in the XML versioning format.

SYSDATE interpreted as synonym for CURRENT TIMESTAMP(0): In Version 10 conversion mode (from both Version 8 and Version 9.1), references to SYSDATE as an ordinary identifier result in SQLCODE -4700. In Version 10 new-function mode, references to SYSDATE as an ordinary identifier are interpreted as synonyms for CURRENT TIMESTAMP(0).

Changes to CURRENT TIMESTAMP special register: In Version 10 conversion mode (from both Version 8 and Version 9.1), references to CURRENT TIMESTAMP in a function call result in SQLCODE -4700. In Version 10 new-function mode, references to CURRENT TIMESTAMP(*integer*) are interpreted as references to the new CURRENT TIMESTAMP special register.

Different data types returned for ADD_MONTHS and LAST_DAY functions: Before Version 10 new-function mode, ADD_MONTHS and LAST_DAY return results of data type DATE. After migration to new-function mode, ADD_MONTHS and LAST_DAY return results of same data type as the input expression (for example, TIMESTAMP). However, if the input expression has a string data type, the result data type is DATE.

New data type for labeled duration SECONDS: Before Version 10 new-function mode, labeled duration SECONDS values are expressed as DECIMAL (15,0) numbers. After migration to new-function mode, labeled duration SECONDS values are expressed as a DECIMAL (27,12) number.

Changes to SECOND, TIMESTAMP, and TIMESTAMP_FORMAT built-in functions: If you have implemented SECOND, TIMESTAMP, or TIMESTAMP_FORMAT user-defined functions, after migration to Version 10 new-function mode, built-in functions SECOND, TIMESTAMP, and TIMESTAMP_FORMAT either allow additional arguments or the argument allows additional data types. If the user-defined function name is not qualified and SYSIBM precedes the function's schema in the SQL path, the built-in functions might be invoked instead of the user-defined function.

New LENGTH and SCALE values for TIMESTAMP data type: Before Version 10 new-function mode, TIMESTAMP columns have a length of 10 bytes and a scale of 0. After migration to new-function mode, the length of a TIMESTAMP column can be 7 - 13 bytes, and the scale can be 0 - 12. However, rows that previously existed continue to have a length of 10 and scale of 0, and an assumed timestamp precision of 6 digits for fractional seconds. This incompatibility applies to the following catalogs: SYSCOLUMNS, SYSCOLUMNS_HIST, SYSDATATYPES, SYSKEYTARGETS, SYSKEYTARGETS_ HIST, and SYSPARMS.

Catalog and directory tables moved to partition-by-growth table spaces: Each table in the following table spaces is moved to its own partition-by-growth table space in new-function mode: DSNDB06.SYSOBJ, DSNDB06.SYSDBASE, DSNDB06.SYSVIEWS, DSNDB06.SYSPLAN, DSNDB06.SYSPKAGE, DSNDB06.SYSDBAUT, DSNDB06.SYSGROUP, DSNDB01.DBD01, and DSNDB01.SPT01. Each table space uses row-level locking, and all tables are in reordered-row format.

Some user-defined tables now part of the catalog: In previous versions, SYSIBM.SYSDUMMYA SYSIBM.SYSDUMMYE, and SYSIBM.SYSDUMMYU tables are user-defined and created as part of the DSNTIJSG job. After migration to new-function mode, these tables are part of the DSNDB06 catalog.

Some statements stored in new LOB columns: After migration to new-function mode, SQL statements that were previously stored in text columns of SYSVIEWS.TEXT, SYSTRIGGERS.TEXT, SYSROUTINES_SRC.CREATEMST, and SYSPACKSTMT.TEXT are now stored in new LOB columns.

Links are removed: As of new-function mode, all links in the catalog and directory are removed.

Change after unloading tables with XML and LOB data to a spanned record data set: There might be an incompatible change after migration to new-function mode for existing applications that read the output data set created by the UNLOAD utility. The application might need to be modified to read the data set because the data set record format is VBS.

DDL statements with LOB columns inherit length from LOB_INLINE_LENGTH: After migration to new-function mode, existing applications inherit the inline length value from subsystem parameter LOB_INLINE_LENGTH if they have CREATE TABLE statements with LOB columns or ALTER TABLE statements that add LOB columns. However, if you maintain the default setting of 0 for LOB_INLINE_LENGTH, then LOB columns continue to be created with no inline attribute by default. Also, if the inline length from the subsystem parameter exceeds the page size of an explicitly or implicitly created universal table space, the LOB column does not inherit the subsystem parameter value.

Possible change to volume selection order: Due to improvements in DB2's record retrieval process, the order in which DB2 selects volumes for storage groups in conversion mode (from Version 8 or Version 9.1) might be different from the order in which DB2 selects volumes in new-function mode.

More space required for SPT01 table space: When you convert to new-function mode, a portion of the contents of the SPT01 table space is moved into a LOB table space. Data in the LOB table space cannot be compressed, so the amount of space that the SPT01 table space requires in new-function mode is greater than the amount of space that SPT01 requires in enabling-new-function mode (from Version

8 or Version 9.1). However, by default, DB2 places most of the LOB data for SPT01 in the base table space, which can be compressed. Therefore, the amount of space that is required for the STP01 table space does not increase significantly when you convert to new-function mode. However, if you set the SPT01_INLINE_LENGTH subsystem parameter to a smaller value, the amount of space that is needed for SPT01 increases because more data is in the LOB table space. You should not need to decrease the SPT01_INLINE_LENGTH value unless the size of SPT01 is close to its 64 GB limit.

Migrating from conversion mode (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)

After you complete the migration process from Version 8, DB2 is in conversion mode from Version 8. After you complete the migration process from Version 9.1, DB2 is in conversion mode from Version 9.1. During conversion mode (from Version 8 or Version 9.1), you cannot use any Version 10 new functions.

Procedure

To migrate from conversion mode (from Version 8 or Version 9.1) to Version 10 enabling-new-function mode (from Version 8 or Version 9.1):

1. Run the installation CLIST again with ENFM specified for the INSTALL TYPE field on panel DSNTIPA1.

   The value of the INPUT MEMBER NAME field is the value that you specified as the output member during migration to conversion mode (from Version 8 or Version 9.1). In a data-sharing environment, it is the output member of the first group member to migrate to conversion mode (from Version 8 or Version 9.1). In Version 10 enabling-new-function mode (from both Version 8 and Version 9.1), the CLIST customizes the installation jobs DSNTIJEN, DSNTIJNF, DSNTIJNG, DSNTIJCS, and DSNTIJES, as well as the Version 10 sample jobs.
2. Press ENTER to continue to panel DSNTIPT.
3. On panel DSNTIPT, choose the output SDSNSAMP data set. If you use the same data set name, it is not deleted or reallocated. You can only change the value of SAMPLE LIBRARY on this panel.
4. Press ENTER to continue.

   Panel DSNTIP00 is displayed.

```
DSNTIP00              ENABLE NEW FUNCTION MODE FOR DB2
===>

Enter storage management properties for defining ENFM image copy data sets:
                     DEVICE        DATA CLASS       MGMT CLASS       STOR CLASS
                     -------       ----------       ----------       ----------
 1 IMAGE COPY   ===>          ===>             ===>             ===>

Enter the data set prefix for the ENFM image copy data sets:
 2 PREFIX       ===>



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 76. Enable new-function mode for DB2 panel: DSNTIP00

5. Review the image copy data set properties in the IMAGE COPY and PREFIX fields and update as required. The device type must be 1 - 8 alphanumeric characters. The value in the PREFIX field must follow z/OS standards for naming data sets.
6. Press ENTER to continue.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Related reference:

"Main panel: DSNTIPA1" on page 265

"Data set names panel 1: DSNTIPT" on page 295

CATENFM (DB2 Utilities)

Completing enabling-new-function mode (from Version 8 or Version 9.1) processing

You need to perform several processing actions while your DB2 subsystem is in enabling-new-function mode (from Version 8 or Version 9.1), which takes place between conversion mode (from Version 8 or Version 9.1) and new-function mode. New DB2 functions are not available when your DB2 system is in enabling-new-function mode (from Version 8 or Version 9.1).

About this task

Attention: After you begin enabling-new-function mode (from Version 8 or Version 9.1), you cannot fall back to your previous release. You can revert to Version 10 conversion mode* (from Version 8 or Version 9.1).

Procedure

To complete enabling-new-function mode (from Version 8 or Version 9.1) processing:

1. Take an image copy of the DB2 catalog and directory.

   This step must be completed before running DSNTIJEN for the first time.

2. From V8 If you had a user-managed real-time statistics database in your previous release, make it accessible so that DSNTIJEN can copy it.

   If you do not want the real-time statistics database of your previous release to be copied, delete it because it is not used after completion of DSNTIJEN. If you do not have a user-defined real-time statistics database (DSNRTSDB) defined, DSNTIJEN issues a warning for step ENFM1100 and skips execution of steps ENFM1103 and ENFM1105. From V8

3. Create shadow data sets for user-defined indexes on the DB2 catalog that reside on user-managed storage.

   You must create shadow data sets for user-defined indexes that are on user-managed storage before you run job DSNTIJEN. DB2 uses these shadow data sets in the REORG steps that convert the catalog table spaces to new-function mode. One of the queries in job DSNTIJPM identifies the affected indexes.

4. Run job DSNTIJEN in LOCALSITE mode. If you run DSNTIJEN in REMOTESITE mode, it will fail with an error message.

   Recommendation: Do not manually reorganize a table space or rebuild indexes on tables in that table space before the migration job step (ENFM0*nn*7) for that table space is complete.

   If you need to determine how much of the enabling-new-function mode (from Version 8 or Version 9.1) process has completed, use the DISPLAY GROUP DETAIL command.

   If the job fails at a table space migration step (ENFM0*nn*7) because of an unrecoverable sort program work space shortage, use one of these options:

   - Increase the number of temporary data sets indicated by the SORTNUM option of the REORG TABLESPACE control statement in the failing step.
   - Remove the SORTNUM and SORTDEVT options of the REORG TABLESPACE control statement, and allocate sort work (SORTWK*xx*) files in the failing step.

   From V8 During copying of the user-defined real-time statistics database to the DB2-defined real-time statistics database (job steps ENFM1103 and ENFM1105), DB2 might issue message DSNT501I with reason code 00C90080. You can ignore this message. From V8

5. Take an image copy of the DB2 catalog and directory after job DSNTIJEN has completed processing.

6. Optional: Run job DSNTIJCI to run the CHECK INDEX utility on all the catalog and directory indexes that were modified or added by job DSNTIJEN.

Related reference:

CATENFM (DB2 Utilities)

Job DSNTIJEN:

Job DSNTIJEN converts DB2 to enabling-new-function mode (from Version 8 or Version 9.1) or enabling-new-function mode* (from Version 8 or Version 9.1).

In general, job DSNTIJEN completes the following functions:

- Saves the current RBA or LRSN in the BSDS.
- Changes types and lengths of existing catalog columns.

- Migrates a number of catalog tables from shared table spaces into dedicated table spaces.

From V8 If you migrated from Version 8, DSNTIJEN also copies the RTS from the user table spaces to new table spaces in the catalog. From V8

DSNTIJEN migrates catalog tables from shared table spaces into dedicated table spaces as follows.

Table 90. Shared table spaces that are converted to dedicated table spaces in Version 10

Table space name in Version 8 and Version 9.1	Corresponding table space names in Version 10
SYSDBASE	SYSTSFAU SYSTSCOL SYSTSTSP SYSTSTPT SYSTSTAB SYSTSIXS, SYSTSIXT[1], SYSTSIXR[1] SYSTSIPT SYSTSREL SYSTSFOR SYSTSSYN SYSTSFLD SYSTSTAU SYSTSKEY
SYSDBAUT	SYSTSDBA SYSTSDBU
SYSGROUP	SYSTSSTG SYSTSVOL
SYSOBJ	SYSTSAUX SYSTSCON SYSTSDAT SYSTSDEP SYSTSENV SYSTSKYC SYSTSPRM SYSTSRAU SYSTSROU, SYSPLUXA[1], SYSPLUXB[1] SYSTSSCM SYSTSTBC SYSTSTRG, SYSTSTRT[1]
SYSPKAGE	SYSTSPKL SYSTSPLY SYSTSPKG SYSTSPKS, SYSTSPKX[1], SYSTSPVR[1] SYSTSPKY SYSTSPKD SYSTSPKA
SYSPLAN	SYSTSPLN SYSTSPLA SYSTSDBR SYSTSPLD SYSTSSTM
SYSVIEWS	SYSTSVEW, SYSTSVWT[1], SYSTSVTR[1] SYSTSVWD

Table 90. Shared table spaces that are converted to dedicated table spaces in Version 10 (continued)

Table space name in Version 8 and Version 9.1	Corresponding table space names in Version 10
1. This is an LOB table space.	

DSNTIJEN consists of the following job steps:

ENFM0000
Terminates pending DSNENFM.* utilities.

ENFM000A
Gets a list of table spaces that are in ICOPY and COPY status.

ENFM000B
Image copies table spaces that are identified in job step ENFM000A.

ENFM0001
Updates the catalog for the new release.

ENFM0010
Enabling-new-function mode for SYSUTILX.

ENFM0020
Checks the new-function mode status of SPT01.

ENFM0021
Cleans up data sets for SPT01 conversion.

ENFM0027
Converts SPT01 to new-function mode.

ENFM0028
Cleans up data sets for SPT01 post-conversion.

ENFM0029
Image copies new table spaces from SPT01.

ENFM0030
Checks the new-function mode status of DBD01.

ENFM0031
Cleans up data sets for DBD01 conversion.

ENFM0037
Converts DBD01 to new-function mode.

ENFM0038
Cleans up data sets for DBD01 post-conversion.

ENFM0039
Image copies new table spaces from DBD01.

ENFM0040
Enabling-new-function mode for SYSLGRNX.

ENFM0050
Checks the new-function mode status of SYSDBASE.

ENFM0051
Cleans up data sets for SYSDBASE conversion.

ENFM0057
Converts SYSDBASE to new-function mode.

ENFM0058
Cleans up data sets for SYSDBASE post-conversion.

ENFM0059
Image copies new table spaces from SYSDBASE.

ENFM0060
Checks the new-function mode status of SYSDBAUT.

ENFM0061
Cleans up data sets for SYSDBAUT conversion.

ENFM0067
Converts SYSDBAUT to new-function mode.

ENFM0068
Cleans up data sets for SYSDBAUT post-conversion.

ENFM0069
Image copies new table spaces from SYSDBAUT.

ENFM0070
Checks the new-function mode status of SYSGROUP.

ENFM0071
Cleans up data sets for SYSGROUP conversion.

ENFM0077
Converts SYSGROUP to new-function mode.

ENFM0078
Cleans up data sets for SYSGROUP post-conversion.

ENFM0079
Image copies new table spaces from SYSGROUP.

ENFM0080
Checks the new-function mode status of SYSOBJ.

ENFM0081
Cleans up data sets for SYSOBJ conversion.

ENFM0087
Converts SYSOBJ to new-function mode.

ENFM0088
Cleans up data sets for SYSOBJ post-conversion.

ENFM0089
Image copies new table spaces from SYSOBJ.

ENFM0090
Checks the new-function mode status of SYSPKAGE.

ENFM0091
Cleans up data sets for SYSPKAGE conversion.

ENFM0097
Converts SYSPKAGE to new-function mode.

ENFM0098
Cleans up data sets for SYSPKAGE post-conversion.

ENFM0099
Image copies new table spaces from SYSPKAGE.

ENFM0100
Checks the new-function mode status of SYSPLAN.

ENFM0101
Cleans up data sets for SYSPLAN conversion.

ENFM0107
Converts SYSPLAN to new-function mode.

ENFM0108
Cleans up data sets for SYSPLAN post-conversion.

ENFM0109
Image copies new table spaces from SYSPLAN.

ENFM0110
Checks the new-function mode status of SYSVIEWS.

ENFM0111
Cleans up data sets for SYSVIEWS conversion.

ENFM0117
Converts SYSVIEWS to new-function mode.

ENFM0118
Cleans up data sets for SYSVIEWS post-conversion.

ENFM0119
Image copies new table spaces from SYSVIEWS.

ENFM9900
Terminates pending DSNENFM.* utilities.

From V8 If you migrated from Version 8, these additional steps are added to DSNTIJEN when you run the installation CLIST in ENFM mode:

ENFM1100
Starts DSNRTSDB (user-defined RTS database) in read-only mode.

ENFM1103
Loads SYSIBM.SYSTABLESPACESTATS from SYSIBM.TABLESPACESTATS.

ENFM1105
Loads SYSIBM.SYSINDEXSPACESTATS from SYSIBM.INDEXSPACESTATS.

ENFM1200
Switches RTS to catalog tables.

From V8

Job DSNTIJCI:

Job DSNTIJCI is an optional job that runs the CHECK INDEX utility on each of the DB2 catalog and directory indexes that are modified or added by job DSNTIJEN. Job DSNTIJCI can be run at any time after completion of job DSNTIJEN.

Job DSNTIJCI checks all indexes in the following directory table spaces:

- DSNDB01.DBD01
- DSNDB01.SPT01
- DSNDB01.SYSDBDXA
- DSNDB01.SYSLGRNX

- DSNDB01.SYSSPUXA
- DSNDB01.SYSSPUXB
- DSNDB01.SYSUTILX

Job DSNTIJCI also checks all indexes in the following catalog table spaces:

- DSNDB06.SYSPLUXA
- DSNDB06.SYSPLUXB
- DSNDB06.SYSTSAUX
- DSNDB06.SYSTSCOL
- DSNDB06.SYSTSCON
- DSNDB06.SYSTSDAT
- DSNDB06.SYSTSDBA
- DSNDB06.SYSTSDBR
- DSNDB06.SYSTSDBU
- DSNDB06.SYSTSDEP
- DSNDB06.SYSTSENV
- DSNDB06.SYSTSFAU
- DSNDB06.SYSTSFLD
- DSNDB06.SYSTSFOR
- DSNDB06.SYSTSIPT
- DSNDB06.SYSTSIXR
- DSNDB06.SYSTSIXS
- DSNDB06.SYSTSIXT
- DSNDB06.SYSTSKEY
- DSNDB06.SYSTSKYC
- DSNDB06.SYSTSPKA
- DSNDB06.SYSTSPKD
- DSNDB06.SYSTSPKG
- DSNDB06.SYSTSPKL
- DSNDB06.SYSTSPKS
- DSNDB06.SYSTSPKX
- DSNDB06.SYSTSPKY
- DSNDB06.SYSTSPLA
- DSNDB06.SYSTSPLD
- DSNDB06.SYSTSPLN
- DSNDB06.SYSTSPLY
- DSNDB06.SYSTSPRM
- DSNDB06.SYSTSPVR
- DSNDB06.SYSTSRAU
- DSNDB06.SYSTSREL
- DSNDB06.SYSTSROU
- DSNDB06.SYSTSSCM
- DSNDB06.SYSTSSTG
- DSNDB06.SYSTSSTM
- DSNDB06.SYSTSSYN

- DSNDB06.SYSTSTAB
- DSNDB06.SYSTSTAU
- DSNDB06.SYSTSTBC
- DSNDB06.SYSTSTPT
- DSNDB06.SYSTSTRG
- DSNDB06.SYSTSTRT
- DSNDB06.SYSTSTSP
- DSNDB06.SYSTSVEW
- DSNDB06.SYSTSVOL
- DSNDB06.SYSTSVTR
- DSNDB06.SYSTSVWD
- DSNDB06.SYSTSVWT

Job DSNTIJCI consists of the following job steps:

DSNCI1
Checks indexes for the DB2 catalog (DSNDB06).

DSNCI2
Checks indexes for DB2 skeleton cursor and skeleton packages table spaces.

DSNCI3
Checks indexes for DB2 database descriptor table spaces.

DSNCI4
Checks indexes for DB2 system utilities table spaces.

DSNCI5
Checks indexes for DB2 log range table spaces.

Related concepts:

"Job DSNTIJEN" on page 655

Related reference:

CHECK INDEX (DB2 Utilities)

Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode

New-function mode can be activated any time after enabling-new-function mode (from Version 8 or Version 9.1) processing is complete.

About this task

When the DB2 subsystem is in new-function mode, all Version 10 function is available for use. From new-function mode, you cannot fall back to your previous version (Version 8 or Version 9.1) or return to conversion mode from Version 8, enabling-new-function mode from Version 8, conversion mode from Version 9.1, or enabling-new-function mode from Version 9.1. A DB2 subsystem in Version 10 enabling-new-function mode (from Version 8 or Version 9.1) or Version 10 new-function mode cannot coexist with a Version 8 or Version 9.1 subsystem.

Procedure

To migrate from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode:

1. Run job DSNTIJNF. New-function mode begins after job DSNTIJNF completes successfully.
2. Optional: Check for consistency between catalog tables by running the queries in member DSNTESQ of *prefix*.SDSNSAMP.
3. Optional: Stop DB2 Version 10.
4. After job DSNTIJNF completes successfully, run job DSNTIJNG. Job DSNTIJNG modifies the application defaults load module, *dsnhdecp*, to allow the DB2 precompiler to accept Version 10 new-function SQL statements by default.
5. If you stopped DB2 before you ran job DSNTIJNG, start DB2 Version 10.
6. From V8 If you migrated from Version 8, run job DSNTIJNX to bind the packages for SPUFI to enable XML support in new-function mode. From V8
7. Run job DSNTIJRT. Job DSNTIJRT executes DSNTRIN, which installs and customizes all DB2-supplied routines and supporting objects that were deferred until new-function mode. This job was customized when the CLIST was run in MIGRATE mode.
8. Run job DSNTIJRV. Job DSNTIJRV validates all DB2-supplied routines and augments existing install verification procedures (IVPs) for specific DB2-supplied routines. It also provides a report that identifies successful validations, warnings, and failures. This job was customized when the CLIST was run in MIGRATE mode.
9. From V9.1 If you used the SYSPROC.XSR_COMPLETE stored procedure during conversion mode from Version 9.1, then update the WLM environment that you use for running XSR_COMPLETE to use Version 10 class paths:
   a. Edit the Java Language Environment (JAVAENV) data set for the WLM environment by adding the following parameter to the environment variable (ENVAR) list. Change the file and path names to the name of the environment variable that was created when you ran job DSNTIJMV, job step DSNTIJR:
      ```
      "_CEE_ENVFILE=/usr/lpp/db2/db2a10_base/classes/!dsn!envfile.txt",
      ```
   b. Refresh the WLM environment that you use for running XSR_COMPLETE.

   From V9.1

What to do next

Recommendations:

- In new-function mode, specify NEWFUN=V10 in your application defaults module. In a data-sharing environment, if you use more than one application defaults module, modify and run the jobs that you use to maintain these modules to specify NEWFUN=V10.
- Alter your DB2 Version 10 buffer pools that have frequent page reads or frequent page writes to use PGFIX YES if you have sufficient real storage available for these buffer pools. Fixing the buffer pages in real storage once and keeping them fixed avoids the processing time that DB2 needs to fix and free pages for each I/O. In some cases, this processing time can be as much as 10% for I/O intensive workloads. To use this option, issue the following command:

  GUPI

  ```
  ALTER BPOOL(bpname) VPSIZE(vpsize) PGFIX(YES)
  ```

Where *bpname* is the name of the buffer pool and *vpsize* is the size of the virtual pool. GUPI

- If you installed or migrated QMF before migrating the DB2 subsystem, you must rerun some QMF jobs to ensure continued proper operation of QMF. For more information, see Tasks to perform when you upgrade DB2 for z/OS after you install QMF.

Related concepts:

"Job DSNTIJRT" on page 549

"Job DSNTIJRV" on page 553

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Related tasks:

Creating the databases and routines (IBM OmniFind Text Search Server Installation, Administration, and Reference)

Job DSNTIJNF:

Job DSNTIJNF activates new-function mode from conversion mode* (from Version 8 or Version 9.1), enabling-new-function mode (from Version 8 or Version 9.1), or enabling-new-function mode* (from Version 8 or Version 9.1).

Job DSNTIJNF contains the following job step:

ENFM9700
Verifies that the DB2 catalog and directory migration is completed.

Related concepts:

"Job DSNTIJNG"

"Job DSNTIJRT" on page 549

"Job DSNTIJRV" on page 553

Job DSNTIJNG:

Job DSNTIJNG must be run during migration to new-function mode to allow the DB2 precompiler to accept Version 10 new-function SQL statements by default.

DSNTIJNG consists of the following job steps:

DSNTIZP
Assembles the application defaults load module, *dsnhdecp*.

DSNTIZQ
Link-edits the application defaults load module, *dsnhdecp*.

DSNTIMQ
Completes SMP/E processing for your application defaults load module. If you want DSNTIJUZ to establish SMP/E GENASM entries in the DB2 system parameter (DSN6*xxxx*) macros and application defaults load module macros, uncomment job step DSNTIMQ.

Related concepts:

"Job DSNTIJNF" on page 663
"Job DSNTIJRT" on page 549
"Job DSNTIJRV" on page 553

Converting EXPLAIN tables for migration from Version 8

From V8

If you migrated from DB2 Version 8, convert remaining pre-Version 10 and EBCDIC-encoded EXPLAIN tables after you migrate to Version 10 new-function mode.

About this task

If you run job DSNTIJXA in conversion mode from Version 8, most of the EXPLAIN tables are converted to Version 10 format. The DSN_STATEMENT_CACHE_TABLE EXPLAIN table is converted to Version 9.1 format because the Version 10 format is not supported in conversion mode from Version 8.

If you ran installation job DSNTIJSG, applied the PTF for APAR PM28500, and then ran DSNTIJXA, the DSN_PROFILE_TABLE and DSN_PROFILE_HISTORY tables contain a downlevel IPADDR column name. This occurs because renaming of columns is not supported in conversion mode from Version 8.

To convert the table to Version 10 format, you must run DSNTIJXA again in Version 10 new-function mode. You can also choose to skip running DSNTIJXA in conversion mode from Version 8 and just run it once in new-function mode to convert all EXPLAIN tables to Version 10 format. If you skip DSNTIJXA in conversion mode from Version 8, you must also convert the EXPLAIN tables to Unicode encoding after you migrate to new-function mode.

Procedure

To convert EXPLAIN tables while the subsystem is in new-function mode:

1. Customize and run job DSNTIJPM to identify EXPLAIN tables that need to be removed or converted. Report 14 identifies EBCDIC-encoded EXPLAIN tables. Report 15 identifies EXPLAIN tables that are not in the Version 10 format.
2. Customize job DSNTIJXA by providing the creator ID of the table or tables to migrate to the Version 10 format, a DB2 subsystem name, and an authorization ID. To migrate all EXPLAIN tables, specify an asterisk as the creator ID.
3. Run job DSNTIJXA. Job DSNTIJXA calls REXX exec DSNTXTA, which migrates to the Version 10 format all EXPLAIN tables or EXPLAIN tables that belong to the specified creator ID.
4. If you did not convert EBCDIC-encoded tables to Unicode in conversion mode from Version 8, complete the following steps to migrate all EBCDIC-encoded EXPLAIN tables to Unicode:
   a. Customize job DSNTIJXB with the following values:
      - A DB2 subsystem name
      - An authorization ID
      - The schema name of the EXPLAIN tables to convert
      - The name of the target database

- The name of the target 4 KB page table space
- The name of the target 8 KB page table space
- The name of the target 16 KB page table space
- The name of the target 8 KB page LOB table space
- The name of the target 32 KB page LOB table space

b. Run job DSNTIJXB. Job DSNTIJXB calls REXX exec DSNTXTB, which generates DB2 cross-loader control statements and equivalent SQL statements that can be used to copy all EBCDIC EXPLAIN tables under a specified creator ID to Unicode equivalents. All EXPLAIN tables that belong to the specified creator ID must be in the current release format.

c. If you have purchased the DB2 Utilities Suite, run job DSNTIJXC to process the control statements that were generated by DSNTIJXB. Otherwise, use the equivalent SQL statements that were generated by DSNTIJXB to convert the tables to Unicode.

d. After you convert an EXPLAIN table from EBCDIC encoding to Unicode encoding, check for joins to that table. When you convert EXPLAIN tables to Unicode, applications that join with EXPLAIN tables might have different results because of the CCSID conversion. For more information, see Objects with different CCSIDs in the same SQL statement (DB2 Internationalization Guide).

e. Repeat steps a, b, c, and d for each creator that has at least one EBCDIC EXPLAIN table.

Reverting to conversion mode* (from Version 8 or Version 9.1) or enabling-new-function mode* (from Version 8 or Version 9.1)

In DB2 Version 10, you can revert to conversion mode* (from Version 8 or Version 9.1) from enabling-new-function mode* (from Version 8 or Version 9.1) or new-function mode. You can also revert to enabling-new-function mode* (from Version 8 or Version 9.1) from new-function mode.

About this task

You can use the reversion process if you encounter problems converting from conversion mode (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode, or to control access to new function after you migrate to new-function mode.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Reverting to conversion mode* (from Version 8 or Version 9.1)

You can revert from DB2 Version 10 enabling-new-function mode (from Version 8 or Version 9.1), enabling-new-function mode* (from Version 8 or Version 9.1), or new-function mode to Version 10 conversion mode* (from Version 8 or Version 9.1). This action might be necessary if you encounter problems or need to control access to new function.

About this task

For example, if you encounter problems after you migrate to enabling-new-function mode (from Version 8 or Version 9.1), you can revert to conversion mode* (from Version 8 or Version 9.1) until the problems are resolved.

Important: You cannot fall back to Version 8 or Version 9.1 from conversion mode* (from Version 8 or Version 9.1).

Procedure

To revert to Version 10 conversion mode* (from Version 8 or Version 9.1):

Run job DSNTIJCS.

Related tasks:

"Moving from conversion mode* (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)" on page 667

"Moving from conversion mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

"Reverting to enabling-new-function mode* (from Version 8 or Version 9.1)"

Reverting to enabling-new-function mode* (from Version 8 or Version 9.1)

If your subsystem is in new-function mode but you decide to postpone having users use new Version 10 functions, you can return to enabling-new-function mode* (from Version 8 or Version 9.1). You can also revert to conversion mode* (from Version 8 or Version 9.1).

About this task

Important: You cannot fall back to Version 8 or Version 9.1 after you have entered enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode.

Procedure

To revert to enabling-new-function mode* (from Version 8 or Version 9.1):

1. Run job DSNTIJES.

   Important: This job sets the DB2 mode to the enabling-new-function mode that corresponds with the version from which you migrated (Version 8 or Version 9.1).
2. In job DSNTIJNG, ensure that the DSNHDECM invocation specifies NEWFUN=V8 or NEWFUN=V9.
3. Run job DSNTIJNG.
4. Verify that DSNHDECM specifies NEWFUN=V8 if you are reverting to enabling-new-function mode* from Version 8 or NEWFUN=V9 if you are reverting to enabling-new-function mode* from Version 9.1.
5. If you are in a data sharing environment and you use more than one application defaults load module, modify and run the jobs that you use to maintain these load modules to specify NEWFUN=V8 or NEWFUN=V9.

Related tasks:

"Migrating from enabling-new-function mode* (from Version 8 or Version 9.1) to new-function mode" on page 668

"Reverting to conversion mode* (from Version 8 or Version 9.1)" on page 665

Returning to enabling-new-function mode (from Version 8 or Version 9.1) or new-function mode

You can return to enabling-new-function mode (from Version 8 or Version 9.1) if you reverted to conversion mode* (from Version 8 or Version 9.1). You can return to new-function mode if you reverted to conversion mode* (from Version 8 or Version 9.1) or enabling-new-function mode* (from Version 8 or Version 9.1).

You can convert to new-function mode only if the DB2 subsystem was previously in enabling-new-function mode (from Version 8 or Version 9.1) and completed enabling-new-function mode (from Version 8 or Version 9.1) processing.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Moving from conversion mode* (from Version 8 or Version 9.1) to enabling-new-function mode (from Version 8 or Version 9.1)

From Version 10 conversion mode* (from Version 8 or Version 9.1), you can resume your migration process by migrating to Version 10 enabling-new-function mode (from Version 8 or Version 9.1).

Procedure

To migrate to enabling-new-function mode (from Version 8 or Version 9.1) from conversion mode* (from Version 8 or Version 9.1):

Run job DSNTIJEN.
Whether DSNTIJEN changes DB2 to enabling-new-function mode* (from Version 8 or Version 9.1) or enabling-new-function mode (from Version 8 or Version 9.1) depends on the state of DB2 before you reverted, as shown in the table below.

Table 91. Overview of steps for converting to DB2 Version 10 enabling-new-function mode (from Version 8 or Version 9.1)

If DB2 reverted to conversion mode* (from Version 8 or Version 9.1) from...	DSNTIJEN will change DB2 to...
new-function mode	enabling-new-function mode* (from Version 8 or Version 9.1)
enabling-new-function mode (from Version 8 or Version 9.1)	enabling-new-function mode (from Version 8 or Version 9.1)[1]
enabling-new-function mode* (from Version 8 or Version 9.1)	enabling-new-function mode* (from Version 8 or Version 9.1)

Note:

1. DSNTIJEN also attempts to complete enabling-new-function mode processing if it has not completed.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Moving from conversion mode* (from Version 8 or Version 9.1) to new-function mode

From Version 10 conversion mode* (from Version 8 or Version 9.1), you can resume your migration process by migrating to Version 10 new-function mode.

Procedure

To migrate to Version 10 new-function mode from Version 10 conversion mode* (from Version 8 or Version 9.1):

Run job DSNTIJNF. If enabling-new-function mode processing did not complete when DB2 was previously in enabling-new-function mode (from Version 8 or Version 9.1), DSNTIJNF issues an error that indicates that you need to run DSNTIJEN.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Migrating from enabling-new-function mode* (from Version 8 or Version 9.1) to new-function mode

After you revert to Version 10 enabling-new-function mode* (from Version 8 or Version 9.1), you can return to Version 10 new-function mode when you are ready.

Procedure

To return to new-function mode:

1. Run job DSNTIJNF.
2. In job DSNTIJNG, ensure that the DSNHDECM invocation specifies NEWFUN=V10.
3. Run job DSNTIJNG.
4. If you are in a data sharing environment and you use more than one application defaults load module, modify and run the jobs that you use to maintain these load modules to specify NEWFUN=V10.

Related concepts:

From V8 "Migration modes for DB2 8 to DB2 10" on page 5 From V8

From V9.1 "Migration modes for DB2 9 to DB2 10" on page 16 From V9.1

Chapter 6. Installing, migrating, and enabling DB2 data sharing

Whether you need to install, migrate, or enable data sharing depends on the type of system that you start with and the type of system that you want to create.

The following table lists the procedures that you need to install, migrate, or enable a data sharing group.

Table 92. Data sharing options

If you have this...	And you want this...	Do this...
No system	Version 10 data sharing	Install a new data sharing group
A Version 10 non-sharing DB2 subsystem	The originating member of a data sharing group	Enable DB2 data sharing
One member in the group	More members in the group	Add new data sharing members
Separate DB2 subsystems	Merged DB2 subsystems into a single group	Merge existing DB2 data into the group
A Version 8 or Version 9.1 data sharing group	A Version 10 data sharing group	Migrate an existing data sharing group to a new release

Related tasks:

Chapter 5, “Installing or migrating DB2,” on page 507

Strategy for creating a data sharing group

When you create a data sharing group, add one member at a time. The first member (the *originating* member) can be created as a new installation, or an existing Version 8 or Version 9.1 member can be migrated to Version 10 and enabled for data sharing.

Enabling DB2 data sharing is a big step. Carefully plan your strategy because after you enable data sharing, it is very difficult to disable it. Disabling should only be considered if your long-term plans are to disable data sharing. Before enabling data sharing, test other major new functions in the release on a single system, and then build and try a test data sharing group. When you are ready to begin a move to production, you must avoid having to fall back to the previous release.

Follow this general strategy for creating a data sharing group:

1. Build and try a test data sharing group. Here is one approach to testing your data sharing group:
   a. Install Version 10 as a single system, and test it with dummy data, or copies of production data, in order to test some of the new functions. Run old applications and begin new application development work.

      In the meantime, prepare the hardware and define the coupling facility structures to enable data sharing.

© Copyright IBM Corp. 1982, 2012

b. Enable data sharing on the test system.

   You have a single-system data sharing group at this point, and this can help you find any initial configuration problems. Make sure that old applications work in this environment.

c. Install additional members in the test group.

   Run applications from different members in the group to fully exercise the group buffer pools and cross-system locking.

2. Move to production. When you are ready to move to production:

   a. Migrate your originating Version 8 or Version 9.1 member to Version 10.

      Recommendation: Wait until your system is stable before enabling data sharing . You can enable data sharing on your existing release and then migrate to Version 10, but it is not recommended, particularly if you plan to migrate soon. If you are in the process of migrating, wait until DB2 enters new-function mode before you move to the data sharing mode.

   b. Verify that the your DB2 system is stabilized in new-function mode. Once the system is stabilized, you can begin using new functions.

   c. Tune applications to contain the level of locking and lock contention rates.

      To reduce the effects of locking contention in a data sharing environment, first control locking costs in a non-data sharing environment. This gives you a baseline from which to perform further tuning after the move to data sharing.

   d. Enable data sharing on the originating member, and run applications on this one-member data sharing group.

   e. Install additional members as needed.

Shared DB2 libraries in a data sharing environment

DB2 target library and distribution library data sets can be shared among all members of a data sharing group. Sharing these libraries reduces the effort of installing and maintaining the different members of the data sharing group. Sharing libraries can also help ensure that members are using the same exit routines. Also, sharing libraries simplifies the tasks of defining a new member to a data sharing group. The member installation process supports sharing of the libraries.

Related concepts:

Database administration for data sharing (DB2 Data Sharing Planning and Administration)

Concurrency and locks in data sharing environments (DB2 Data Sharing Planning and Administration)

Related tasks:

Controlling resource usage (DB2 Performance)

Subsystem parameters and application defaults in a data sharing environment

In a data sharing environment, you must consider the scope of subsystem parameters before setting the values for each member. Also consider the subsystem parameter load module and the application defaults load module that you will specify for each member.

Specifying a member's subsystem parameter load module

Every member of a data sharing group must have its own unique load module for subsystem parameters (sometimes called DSNZPARM in a non-data sharing environment).

About this task

The load module for subsystem parameters is built by job DSNTIJUZ and stored in the *prefix*.SDSNEXIT target library. Every member must use a different name for its parameter load module because the *prefix*.SDSNEXIT target library can be shared among all members of the data sharing group. The installation process requires that you provide the name of the load module for a member.

Recommendation: Name each member's load module using the convention DSNZP*xxx*, where *xxx* includes the number in the member name and the group identifier. For example, DB1A's subsystem parameter load module could be named DSNZP01A.

The subsystem parameter load module name for a member is an optional parameter on the EXEC statement in the JCL procedure that is used to start the *ssnm*MSTR address space. This optional parameter provides support for an operator (or automated operations) not specifying the subsystem parameter load module name when starting a member.

Procedure

To specify the subsystem parameter load module for a member:

Specify the load module name in the EXEC statement of the *ssnm*MSTR JCL procedure in SYS1.PROCLIB as follows:

```
//IEFPROC EXEC PGM=DSNYASCP,PARM='ZPARM(DSNZPxxx)',...
```

Related concepts:

"Subsystem parameter module and application defaults module values" on page 265

The scope and uniqueness of DB2 subsystem parameters

Although various parameters affect the operation of only a single member, some parameters must have the same value on all the members. For example, each catalog alias name must be the same. Other parameter values must be different on each member. For example, each member specifies a different BSDS name.

The values of most parameters do not need to be unique. Recommendations are offered for some of these parameters. In the following tables, the parameter is indicated by the name of the corresponding installation panel field. However, some of the parameters do not reside in the DSNZP*xxx* load module.

Parameter values that must be different on each member

The table below shows the values of the installation panel fields that must be different for every member of a data sharing group. You specify values for these fields when a member is installed, migrated, or updated.

Table 93. Installation panel fields that must have different values on each member of a data sharing group

Installation panel field (parameter) name	Panel ID	Comment
Active Logs: COPY 1 PREFIX	DSNTIPH	Each member writes to its own recovery log.
Active Logs: COPY 2 PREFIX	DSNTIPH	Each member writes to its own recovery log.
Archive Logs: COPY 1 PREFIX (ARCPFX1 parameter)	DSNTIPH	Each member writes to its own recovery log.
Archive Logs: COPY 2 PREFIX (ARCPFX2 parameter)	DSNTIPH	Each member writes to its own recovery log.
Bootstrap Data Sets : COPY 1 NAME	DSNTIPH	Each member has its own BSDS.
Bootstrap Data Sets : COPY 2 NAME	DSNTIPH	Each member has its own BSDS.
COMMAND PREFIX	DSNTIPM	The command prefix used to route commands to this member.
DB2 NETWORK LUNAME	DSNTIPR	Even if you do not use distributed database, this identifier is required to ensure that logical unit of work IDs (LUWIDs) are unique across the members of a data sharing group.
MEMBER IDENTIFIER	DSNTIPJ	The unique identifier for this IRLM.
MEMBER NAME	DSNTIPK	The name for this member.
PARAMETER MODULE	DSNTIPO3	The name of the parameter load module for this member.
PROC NAME	DSNTIPI	The name of the IRLM procedure that z/OS invokes if IRLMAUT=YES.
RESYNC PORT	DSNTIP5	The port used for resynchronization of two-phase commit processes when using TCP/IP network protocols.
SUBSYSTEM NAME	DSNTIPM	The DB2 subsystem identifier for this member.
SUBSYSTEM SEQUENCE	DSNTIPM	The DB2 subsystem sequence number for this member.
SUBSYSTEM NAME (IRLM)	DSNTIPI	The name of the IRLM subsystem associated with a particular member. This name must be unique within the Parallel Sysplex.
WORK FILE DB	DSNTIPK	The name of the work file database for this member.

Parameter values that must be the same on every member

The table below shows the installation panel fields that must have the same values for every member of the data sharing group.

Table 94. Installation panel fields that must have the same values for every member of a data sharing group

Installation panel field (parameter) name	Panel ID	Comment
CATALOG ALIAS (CATALOG parameter)	DSNTIPA2	The DB2 catalog alias name.
DEL CF STRUCTS (DEL_CFSTRUCTS_ON_RESTART parameter)	DSNTIPK	Specifies whether DB2 is to attempt to delete coupling facility structures during restart.
COMPRESS_SPT01 parameter in macro DSN6SPRM	N/A	Specifies whether the SPT01 directory space is to be compressed.
DBADM CREATE AUTH (DBACRVW parameter)	DSNTIPP1	Specifies whether an authorization ID with DBADM authority can create a view for another authorization ID on that database's tables. Or if the authorization ID can create an alias for itself or another authorization ID for a table in that database.
DB2 LOCATION NAME	DSNTIPR	The location name for the entire data sharing group. This name is required.
DESCRIBE FOR STATIC (DESCSTAT parameter)	DSNTIP4	Specifies whether DB2 is to build a DESCRIBE SQLDA when binding static SQL statements
DIRECTORY AND CATALOG DATA (CATDDACL, CATDMGCL, and CATDSTCL parameters)	DSNTIPA2	Specifies the SMS classes that are used to define VSAM data sets for indexes on the DB2 catalog and directory. DATA CLASS, MGMT CLASS, and STOR CLASS settings must be the same on all members.
DIRECTORY AND CATALOG INDEXES (CATXDACL, CATXMGCL, and CATXSTCL parameters)	DSNTIPA2	Specifies the SMS classes that are used to define VSAM data sets for the DB2 catalog and directory. DATA CLASS, MGMT CLASS, and STOR CLASS settings must be the same on all members.
DRDA PORT	DSNTIP5	The DRDA port number for the entire data sharing group. This name is required if using TCP/IP network connections.
EVALUATE UNCOMMITTED (EVALUNC parameter)	DSNTIP8	Specifies whether predicate evaluation can occur on uncommitted data of other transactions. The option applies only to stage 1 predicate processing that uses table access for queries with isolation level RS or CS. Table access includes table space scan, index-to-data access, and RID-list processing.
EXTENDED SECURITY (EXTSEC parameter)	DSNTIPR	Determines the content of error codes returned to a client when a connection request fails because of security errors. Also allows users to change their passwords if their host passwords expire.
GROUP ATTACH	DSNTIPK	The group attachment name that allows programs to generically attach to any member of the group. The default is the member name of the first installed member of the group.
GROUP NAME (GRPNAME parameter)	DSNTIPK	The DB2 group name.

Table 94. Installation panel fields that must have the same values for every member of a data sharing group (continued)

Installation panel field (parameter) name	Panel ID	Comment
INDEX SPACE ALLOCATION (IXQTY parameter)	DSNTIP7	Specifies the amount of space in KB for primary and secondary space allocation for DB2-defined data sets for index spaces that are being created without the USING clause.
IRLM XCF GROUP NAME	DSNTIPJ	The IRLM group name.
INSTALL DD CONTROL SUPT (RGFINSTL parameter and related parameters)	DSNTIPZ	All members of the data sharing group must use the same set of data definition control registration tables, or unpredictable results can occur.
MINIMUM DIVIDE SCALE (DECDIV3 parameter)	DSNTIP4	Whether to retain at least three digits to the right of the decimal point after any decimal division.
PAD INDEXES BY DEFAULT (PADIX parameter)	DSNTIPE	Specifies whether indexes are created as PADDED or NOT PADDED by default.
REAL TIME STATS (STATSINT parameter)	DSNTIPO	Specifies the time, in minutes, that DB2 waits before attempting to write out page set statistics to the real-time statistics tables.
RETVLCFK parameter in macro DSN6SPRM	N/A	Specifies whether the VARCHAR column is to be retrieved from a padded index.
SECURE PORT	DSNTIP5	TCP/IP port number for secure DRDA clients, which, if specified, needs to be the same for all members of the group.
SITE TYPE (SITETYP parameter)	DSNTIPO	All members on the remote site, after they need to become the local site, must have the same value (LOCALSITE).
SKIP UNCOMM INSERTS (SKIPUNCI parameter)	DSNTIP8	Specifies whether cursors bound with read stability or cursor stability will ignore uncommitted inserts made by other transactions.
STATCLUS parameter in macro DSN6SPRM	N/A	The type of clustering statistics collected by the RUNSTATS utility.
STATISTICS HISTORY (STATHIST parameter)	DSNTIP61	Which inserts and updates are recorded in catalog history tables.
STATISTICS ROLLUP (STATROLL parameter)	DSNTIP61	Specifies whether the RUNSTATS utility is to aggregate the partition-level statistics, even though some parts may not contain data.

Table 94. Installation panel fields that must have the same values for every member of a data sharing group (continued)

Installation panel field (parameter) name	Panel ID	Comment
SYSTEM ADMIN 1 (SYSADM parameter) SYSTEM ADMIN 2 (SYSADM2 parameter) SYSTEM OPERATOR 1 (SYSOPR1 parameter) SYSTEM OPERATOR 2 (SYSOPR2 parameter) SECURITY ADMIN 1 (SECADM1 parameter) SEC ADMIN 1 TYPE (SECADM1_TYPE parameter) SECURITY ADMIN 2 (SECADM2 parameter) SEC ADMIN 2 TYPE (SECADM2_TYPE parameter) SEPARATE SECURITY (SEPARATE_SECURITY parameter) REVOKE DEP PRIV (REVOKE_DEP_PRIVILEGES parameter)	DSNTIPP1	Installation SYSADM and SYSOPR authorities and security authorities.
TABLE SPACE ALLOCATION (TSQTY parameter)	DSNTIP7	Specifies the amount of space in KB for primary and secondary space allocation for DB2-defined data sets for table spaces that are being created without the USING clause.
TCP/IP ALREADY VERIFIED (TCPALVER parameter)	DSNTIP5	Whether incoming TCP/IP requests are accepted by DB2 without a password or RACF PassTicket. This option must be the same on all members or requesters will have inconsistent results.
TRACKER SITE (TRKRSITE parameter)	DSNTIPO	Whether DB2 is a tracker site for disaster recovery services.
UNICODE IFCIDS (UIFCIDS parameter)	DSNTIPN	Specifies whether output from IFC records should include Unicode information.
VARY DS CONTROL INTERVAL (DSVCI parameter)	DSNTIP7	Specifies whether DB2 is to use a VSAM CONTROL INTERVAL that matches the page size for data set allocation.

Other recommendations

The table below shows the installation panel fields that can have the same or different values for members of a data sharing group.

Table 95. Recommended values for installation panel fields

Installation panel field (parameter) name	Panel ID	Comment
DEALLOC PERIOD (DEALLCT parameter)	DSNTIPA	The length of time during which an archive read tape unit is allowed to remain unused before it is deallocated. **Recommendation:** Do not archive to tape. If you must, however, you should specify 0 for this parameter unless you intend to run all RECOVER jobs from the same member. Specifying a deallocation delay means that the tape is not available to any other member until the deallocation time expires.
DEFAULT 4-KB BUFFER POOL FOR USER DATA (TBSBPOOL parameter) DEFAULT 8-KB BUFFER POOL FOR USER DATA (TBSBP8K parameter) DEFAULT 16-KB BUFFER POOL FOR USER DATA (TBSBP16K parameter) DEFAULT 32-KB BUFFER POOL FOR USER DATA (TBSBP32K parameter) DEFAULT BUFFER POOL FOR USER LOB DATA (TBSBPLOB parameter) DEFAULT BUFFER POOL FOR USER XML DATA (TBSBPXML parameter)	DSNTIP1	The default buffer pool for table spaces that are created without a specified default. For consistent results when creating objects on different members, make this default the same on all members.
DEFAULT BUFFER POOL FOR USER INDEXES (IDXBPOOL parameter)	DSNTIP1	The default buffer pool for indexes that are created without a specified default. For consistent results when creating objects on different members, make this default the same on all members.
DEVICE TYPE 1 (UNIT parameter) DEVICE TYPE 2 (UNIT2 parameter)	DSNTIPA	The device type or unit name for storing archive logs. **Recommendation:** Archive the first copy of the log to disk.
DISABLE_EDMRTS parameter	none	Whether the environmental descriptor manager (EDM) is to collect real-time statistics about the last use of packages. The value of this subsystem parameter should be the same on all members.
EDM LIMIT BELOW THE BAR (EDMPOOL parameter)	DSNTIPC	If you are migrating from Version 8 or Version 9, change the default value of 0 to the EDMPOOL value that you used in Version 8 or Version 9.
IMMEDIATE WRITE (IMMEDWRI parameter)	DSNTIP8	Determines the IMMEDWRITE bind option that is to be used for plans and packages that are run on the member. **NO** The IMMEDWRITE bind option that is specified for the plan or package is used. The default is NO. **YES** The IMMEDWRITE(YES) option is used for all plans and packages. This option might impact performance. The IMMEDWRITE bind option controls when DB2 writes updated group buffer pool dependent pages to the coupling facility (or to disk for GBPCACHE NONE|SYSTEM objects). DB2 can either write the GBP-dependent pages immediately, before phase 1 commit, or at phase 1 commit. Use the bind option to avoid “row not found” conditions for transactions that have ordered dependencies.

Table 95. Recommended values for installation panel fields (continued)

Installation panel field (parameter) name	Panel ID	Comment
READ COPY2 ARCHIVE (ARC2FRST parameter)	DSNTIPO	Whether the second copy of the archive log should be read first for restart and recovery. **Recommendation:** This option should have the same value (either YES or NO) for all members of the group. Otherwise, the parameter, as it is set on the member that owns the archive log data set, determines which copy is used.
READ TAPE UNITS (MAXRTU parameter)	DSNTIPA	The maximum number of dedicated tape units that can be allocated to read archive log tape volumes concurrently. **Recommendation:** Do not archive to tape. If you must, however, it is vital that you have enough tape units allocated to the member doing the recovery to merge the archive logs from **all** members in the group that have updated the object being recovered. Thus, if there are eight members in the group, make sure you specify at least eight on this panel for each member.
RECORDING MAX (MAXARCH parameter)	DSNTIPA	The maximum number of archive logs to be recorded in the BSDS. **Recommendation:** All members should use the maximum value of 1000, making it easier to transfer a workload from one member to another.
RETAINED LOCK TIMEOUT (RETLWAIT parameter)	DSNTIPI	A multiplier to apply to the timeout value for a connection when that connection is waiting for an incompatible lock held by another failed member. Locks that are held by failed members are called *retained* locks. **Recommendation:** If you have automatic restart, or some other restart automation that quickly restarts failed members, choose a non-zero value for this parameter.
START IRLM CTRACE	DSNTIPI (or TRACE=YES on IRLM startup procedure)	Whether diagnostic traces are activated when IRLM is started. **Recommendation:** Diagnostic traces should be activated for all members of the group. The negligible performance overhead is outweighed by the benefit of easier problem resolution. Only those systems that have reached the limits of processor capacity should consider not automatically activating these traces.
SUBGRP ATTACH	DSNTIPK	The name of the subgroup attachment that is to be used for this subsystem.

Related concepts:

"Order-dependent transactions" on page 200

The impact of archiving logs in a data sharing group (DB2 Data Sharing Planning and Administration)

Related reference:

"Directory of subsystem parameters and application default values" on page 242

"Subsystem parameters that are not on installation panels" on page 253

BIND and REBIND options (DB2 Commands)

Updating subsystem parameters for a member

No group-wide subsystem parameters can be updated; however, you can update the subsystem parameters for an individual member.

Procedure

To update subsystem parameters for a member of a data sharing group:

Specify the following installation option on the DSNTIPA1 installation panel:

```
INSTALL TYPE ===> UPDATE
```

This option updates parameters for the member.

Related concepts:

"The scope and uniqueness of DB2 subsystem parameters" on page 671

Related reference:

"Main panel: DSNTIPA1" on page 265

Application defaults parameters

An application defaults load module, *dsnhdecp*, is generated as part of installing or migrating DB2. In a data sharing environment, this module is generated only for the first member that is installed or migrated.

If you provide a separate *dsnhdecp* module for each member, use the same *dsnhdecp* settings (except for the SSID parameter) for all members of the data sharing group. If you provide a separate *dsnhdecp* module for each member, then after DB2 enters new-function mode, identify the jobs that you use to maintain those *dsnhdecp* modules. You must modify those jobs to specify NEWFUN=V10, and then run those jobs.

Related concepts:

"Subsystem parameter module and application defaults module values" on page 265

Installing a new data sharing group

Although you can install and immediately enable DB2 data sharing on a new Version 10 member, you should do so only in low-risk situations. The recommended approach is to migrate to, or install, a Version 10 member, use it for a while, and then enable data sharing.

About this task

If you decide to install and immediately enable data sharing on a new Version 10 member, that member becomes the *originating member* of the data sharing group. This member's DB2 catalog is used as the DB2 catalog for the data sharing group.

Procedure

To install a new data sharing group:

1. On installation panel DSNTIPA1, specify:

   ```
   INSTALL TYPE ===> INSTALL
   DATA SHARING ===> YES
   ```

2. On installation panel DSNTIP0A, specify 1 to indicate the group data sharing function.
3. On installation panel DSNTIPK, specify:

   ```
   GROUP NAME   ===> group name
   MEMBER NAME  ===> originating member name
   ```

   Verify that the *originating member name* is unique within the z/OS Parallel Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the group name and member name you specify here.
4. Complete the installation panels and specify all of the parameters.
5. Complete the installation of DB2 by following the installation steps in "Installing DB2" on page 507.
6. Run the installation verification procedures (IVP).

Related concepts:

"Subsystem parameters and application defaults in a data sharing environment" on page 670

"The scope and uniqueness of DB2 subsystem parameters" on page 671

"Data sharing naming conventions" on page 153

Related tasks:

"Enabling DB2 data sharing"

"Invoking the CLIST" on page 224

Related reference:

"DATA SHARING field (DSHARE subsystem parameter)" on page 268

Enabling DB2 data sharing

The enabling process allows an existing DB2 subsystem to be the originating member of a data sharing group; it does not allow you to change the subsystem name.

About this task

To enable data sharing:

Related concepts:

"The scope and uniqueness of DB2 subsystem parameters" on page 671

Related tasks:

"Installing DB2" on page 507

"Renaming a member by stopping DB2" on page 741

" Migration step 9: Stop DB2 Version 8 or Version 9.1 activity" on page 588

Running the CLIST for enabling data sharing

The process of enabling data sharing involves running some of the installation jobs. Before you run the installation jobs, tailor them to your environment by running the installation CLIST.

About this task

The enabling CLIST tailors these jobs:

DSNTEJ0	DSNTEJ3M	DSNTEJ6W	DSNTEJ78
DSNTEJ1	DSNTEJ3P	DSNTEJ6Z	DSNTESA
DSNTEJ1L	DSNTEJ4C	DSNTEJ61	DSNTESC
DSNTEJ1P	DSNTEJ4P	DSNTEJ62	DSNTESD
DSNTEJ1S	DSNTEJ5A	DSNTEJ63	DSNTESE
DSNTEJ1T	DSNTEJ5C	DSNTEJ64	DSNTIJDE
DSNTEJ1U	DSNTEJ5P	DSNTEJ65	DSNTIJFT
DSNTEJ2A	DSNTEJ6	DSNTEJ66	DSNTIJGF
DSNTEJ2C	DSNTEJ6D	DSNTEJ67	DSNTIJIN
DSNTEJ2D	DSNTEJ6P	DSNTEJ7	DSNTIJMV
DSNTEJ2E	DSNTEJ6R	DSNTEJ71	DSNTIJTM
DSNTEJ2F	DSNTEJ6S	DSNTEJ73	DSNTIJUZ
DSNTEJ2P	DSNTEJ6T	DSNTEJ75	DSNTIJVC
DSNTEJ2U	DSNTEJ6U	DSNTEJ76	
DSNTJE3C	DSNTEJ6V	DSNTEJ77	

The enabling CLIST also edits the DB2I CLISTs.

Procedure

To tailor the installation jobs for enabling data sharing:

1. Invoke the installation CLIST.
2. On installation panel DSNTIPA1, specify:

   ```
   INSTALL TYPE ===> INSTALL
   DATA SHARING ===> YES
   ```

3. On installation panel DSNTIP0A, specify option 3 to enable data sharing.
4. On installation panel DSNTIPK, specify:

   ```
   GROUP NAME  ===> group name
   MEMBER NAME ===> originating member name
   ```

   Verify that the *originating member name* is unique within the z/OS Parallel Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the group name and member name that you specify here.
5. Complete the remaining installation panels, specifying parameters according to the DB2 subsystem parameters guidelines.

Related concepts:

"Subsystem parameters and application defaults in a data sharing environment" on page 670

Related tasks:

"Invoking the CLIST" on page 224

Stopping DB2 activity

Before running any installation jobs to enable data sharing, ensure that all work is stopped on the subsystem that you chose to be the originating member.

Procedure

To stop work on DB2:

1. Issue the following command, where *-DSN1* is the subsystem command prefix that is defined for DB2. You must have system administrator (SYSADM) or system operator (SYSOPR) authority to issue this command.

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI The QUIESCE keyword allows DB2 to complete the processing of programs that are currently executing. This might require some processing time.
2. Issue the following command to allow only the system administrators and system operators that were defined during installation to access DB2:

   GUPI

   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```

   GUPI If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. For example, if the bootstrap data set (BSDS) does not match the subsystem parameter values, ensure that the correct jobs were run for DSNTIJUZ. Ensure that you started DB2 with the correct subsystem parameter option.
3. Ensure that all work is complete.
   - Make sure no units of recovery remain. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```

     GUPI Then use -DSN1 RECOVER INDOUBT for any indoubt threads.
   - Ensure that all utilities are completed or terminated. To ensure that no utility work remains, issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY UTILITY(*)
     ```

     GUPI

     After you have determined the utilities that are running, you can let them complete processing or you can terminate the utility. To stop all utilities, issue the following command:

```
-DSN1 TERM UTILITY(*)
```

- Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. To determine existing restrictions, issue the following commands:

  GUPI

  ```
  -DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
  -DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
  ```

  GUPI

  You must have system administrator or system operator authority (specified on installation panel DSNTIPP1) to issue this command.

  Recover any table spaces and index spaces that have write error ranges or deferred restart states by using the RECOVER utility.

4. To stop DB2, issue the following command:

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI

   A user with system administrator (SYSADM) or system operator (SYSOPR) authority must enter this command.

Related reference:

-STOP DB2 (DB2) (DB2 Commands)

-DISPLAY THREAD (DB2) (DB2 Commands)

-DISPLAY UTILITY (DB2) (DB2 Commands)

-TERM UTILITY (DB2) (DB2 Commands)

-DISPLAY DATABASE (DB2) (DB2 Commands)

Identifying DB2 to z/OS

Job DSNTIJMV completes some of the steps that are required to identify DB2 to z/OS. These steps include updating members of SYS1.PARMLIB and SYS1.PROCLIB and creating Language Environment options data sets for the DB2 core WLM environment for Java routines.

About this task

Do not overwrite the z/OS-supplied entries for DB2 and IRLM in the program properties tables (PPT) in SYS1.PARMLIB(SCHED*xx*). The operating system supplies default PPT values for all DB2 modules that require them. You should not change these values.

Procedure

To define DB2 to z/OS:

1. Run job DSNTIJMV. If job DSNTIJMV runs successfully, it produces return codes of 0.
2. Review the content of the Language Environment options data set and adjust the following environment variables as needed. The data set is created by

DSNTIJMV and must be allocated by the JAVAENV DD of the address space procedure for the DB2 core WLM environment for Java routines.

- _CEE_ENVFILE, if present, needs to be set to the name and path of the environment variable file that was created by job step DSNTIJR.
- JVMPROPS needs to be set to the name and path of the JVM startup options file that was created by job step DSNTIJS.
- DB2_BASE and JCC_HOME need to be set to the names of the HFS directories that you selected for the DB2 for z/OS and JCC product classes.

What to do next

z/OS requirements: Each DB2 and each IRLM that you define to z/OS in the IEFSSN*xx* PARMLIB member requires a z/OS system linkage index (LX). The default number of these indexes that z/OS reserves is 165. If you place all of your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 165 LXs; otherwise, your subsystems might not start. If you need more than 165 LXs, use the NSYSLX option on the z/OS IEASYS*xx* PARMLIB member to increase this number.

For later diagnosis of IRLM problems, also ensure that the following requirements are met:

- Specify the IRLM dump formatting modules DXRRLM50, DXRRLM55, and DXRRLS55 in a BLSCUSER member in a data set that is in the system's PARMLIB concatenation.
- Copy the DXRRL186, DXRRLFTB, DXRRLM50, DXRRLM55, and DXRRLS55 modules from the IRLM SDXRRESL data set to a data set in the system's link list. Or, add the SDXRRESL data set to the system's link list in a PROG*xx* member of a data set in the system's PARMLIB concatenation. If you use more than one level of IRLM, you will need to use a STEPLIB concatenation when using IPCS to process dumps that are taken for the levels whose modules are not in the link list.

Additional changes to SYS1.PARMLIB and SYS1.PROCLIB: Because different sites have different requirements for identifying DB2 to z/OS, job DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes to SYS1.PARMLIB and SYS1.PROCLIB might be incomplete. You might have additional procedures of your own to rename. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB, or by editing DSNTIJMV.

Recommendation: Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV step. For SYS1.PROCLIB, submit the procedure-update section of job DSNTIJMV. Before you make the updates, read the following information and examine job DSNTIJMV to study the updates that it makes. Then use an editor such as ISPF/PDF to make the updates to SYS1.PARMLIB.

Related concepts:

Runtime environment for Java routines (Application Programming for Java)

Altering the current active log data sets

When job DSNTIJIN is tailored for enabling a DB2 subsystem to become the originating member of a new data sharing group, it does not define any data sets. It only alters the member's active log data sets to use SHAREOPTIONS (2 3).

Procedure

To alter the active log data sets:

Run job DSNTIJIN.
If DSNTIJIN runs successfully, it produces return codes of 0. Check any VSAM messages carefully.

Related concepts:

Allocating and extending data sets (DB2 Performance)

Defining DB2 initialization parameters

Job DSNTIJUZ generates the DB2 data-only load module DSNHMCID, the application defaults load module (*dsnhdecp*), and the subsystem parameter module. The application defaults load module is the name that was specified for the APPL DEFAULTS field on installation panel DSNTIPO3. The subsystem parameter module is the name that was specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

Procedure

To define DB2 initialization parameters:

1. If you added a STEPLIB statement to the DB2 start procedures, modify the SYSLMOD steps to point to the library in the STEPLIB statement instead of to the library in *prefix*.SDSNEXIT.
2. If the DB2 distribution library prefix is different from the target library prefix, edit DSNTIJUZ to correct the data set name for *prefix*.ADSNLOAD.
3. If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBAA10, edit DSNTIJUZ to include the SMP/E temporary data set (SMPTLIB) in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. SMPTLIB is *hlq*.HDBAA10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

   ```
   SET   BOUNDARY (GLOBAL).
   LIST  DDDEF ( SMPTLIB ).
   ```

4. Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD.
5. Run job DSNTIJUZ.

   If you want DSNTIJUZ to establish SMP/E GENASM entries in the DB2 system parameter (DSN6*xxxx*) and application defaults module macros, you need to uncomment job step DSNTIMQ. You might receive message GIM65001 when you run steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when you run step DSNTIMQ. You can ignore these messages.
6. If job DSNTIJUZ fails or abends, correct the problem and rerun the job.

Related reference:

"PARAMETER MODULE field" on page 358

"APPL DEFAULTS field" on page 359

Recording DB2 data to SMF (optional)

When you enable DB2 data sharing, you can specify whether DB2 statistical, accounting, and audit trace data is to be collected.

Procedure

To specify the data that is to be collected:

1. Choose from the following options:

Option	Description
To have DB2 collect statistical information	Accept the default (YES class 1) for the SMF STATISTICS option on installation panel DSNTIPN. To collect statistical information for deadlock or timeout, specify class 3. To collect information about DDF error conditions, specify class 4.
To have DB2 collect accounting information	Accept the default (1) or specify * (all classes) for the SMF ACCOUNTING option on installation panel DSNTIPN.
To have DB2 collect auditing information	Specify * (all classes) for the AUDIT TRACE option on installation panel DSNTIPN.

   In all cases, DB2 invokes a trace, passing the data it collects to the System Management Facility (SMF) of z/OS.

   DB2 also passes performance data to SMF whenever an accounting, statistics, or audit trace is successfully started or stopped. DB2 can also record other performance data. After you complete the installation process, you can use commands to have DB2 record performance data for over 230 different subsystem events.

2. If, during installation, you requested that DB2 pass accounting and statistics data to SMF, update the SMFPRM*xx* member of SYS1.PARMLIB as follows:
   - Specify the ACTIVE parameter.
   - Specify STC,NODETAIL for the SUBSYS parameter.
   - Specify the proper TYPE subparameter of the SYS and SUBSYS parameters.

What to do next

During DB2 execution, you can use the SMF SET or SS command to alter the SMF parameters. For example, you can record the statistics trace class 1 IFCIDs 0001, 0002, and 0202 (SMF record type 100); accounting trace class 1 IFCIDs 0003 and 0239 (SMF record type 101); and all other DB2 trace records (SMF record type 102) to SMF. To record this information, issue the following command:

```
SYS(TYPE(100:102))
```

For DB2 to pass data to SMF, you must allocate an adequate supply of SMF buffers. The default buffer settings are probably insufficient.

You can specify SMF buffering on the VSAM BUFSP parameter of the Access Method Services DEFINE CLUSTER statement. Do not use the default settings if DB2 data is sent to SMF. Specify CISZ(4096) and BUFSP(81920) on the DEFINE CLUSTER statement for each SMF VSAM data set. These values for CISZ and BUFSP are the minimum requirement for DB2. You might need higher values for CISZ and BUFSP, depending on the requirements of all your z/OS subsystems.

You can also code an IEFU84 SMF exit routine to process the records that are produced.

Related tasks:

Controlling traces (DB2 Administration Guide)

Related reference:

-START TRACE (DB2) (DB2 Commands)

z/OS MVS Initialization and Tuning Guide

Related information:

z/OS MVS Initialization and Tuning Reference

Connecting DB2 to TSO

If you are enabling data sharing, connect DB2 to TSO after you define the initialization parameters. This step is necessary only if you specified a group attachment name.

About this task

Access to TSO is required to support the interactive component of DB2 (DB2I). Access to TSO also allows batch applications to access DB2 when those batch programs are executed under the TSO terminal monitor program (TMP).

Procedure

To attach DB2 to TSO:

1. Make DB2 load modules available to TSO and batch users.
2. Make DB2 CLISTs available to TSO and batch users.
3. Make PL/I options available (if applicable).
4. Make panels, messages, and load modules available to ISPF and TSO.
5. Connect the DB2I panels to the ISPF Main Panel.

Related tasks:

"Make DB2 load modules available to TSO and batch users" on page 521

"Make DB2 CLISTs available to TSO and batch users (DSNTIJVC)" on page 522

"Ensure that PL/I options are available" on page 523

"Make DB2I panels, messages, and load modules available to ISPF and TSO" on page 523

"Connect DB2I panels to the ISPF main panel" on page 524

Defining DB2 to z/OS

After you run job DSNTIJMV, you might need to define DB2 in SYS1.PARMLIB. This step is only necessary if you are changing the command prefix, or adding or changing the group attachment name.

About this task

Job DSNTIJMV makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member
- Creates new load module libraries in the LNKLST*xx* member

The load module library SDSNLINK contains the early code. SDSNLINK contains modules that must be placed in the link list look-aside address space (LLA)

because they are loaded at subsystem initialization during the IPL.

Procedure

To define DB2 to z/OS:

IPL z/OS.
During the z/OS IPL, message DSN3100I appears on the z/OS console, stating that DB2 is ready for the START command.
Alternatively, to define DB2 to z/OS until the next IPL, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands.

Related reference:

SETPROG Command

SETSSI Command

Starting the DB2 subsystem

During the process of enabling data sharing, start the DB2 subsystem after you define DB2 to z/OS.

Procedure

To start DB2:

1. Start the IRLM, if you have not requested that DB2 automatically start the IRLM. You must start IRLM before you start DB2.

   Use the following command:

   ```
   START irlmproc
   ```

   where *irlmproc* is the name that you specified for the PROC NAME option on IRLM Panel 1 (DSNTIPI).

   If you specified YES for the AUTO START option on IRLM Panel 1 (DSNTIPI), DB2 starts the IRLM automatically.
2. Start DB2 from the z/OS console. Use the following command:

   ```
   -DSN1 START DB2
   ```

   where (-DSN1) is the subsystem command prefix that you defined for DB2.

   DB2 uses the subsystem parameter module that is specified in the start-up JCL procedure in SYS1.PROCLIB:

   ```
   //IEFPROC EXEC PGM=DSNYASCP,PARM='ZPARM(DSNZPxxx)', ...
   ```

   where *DSNZPxxx* is the value that you specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

   If you need to change the name of the *DSNZPxxx* module, you can edit SYS1.PROCLIB. Alternatively, you can override the *DSNZPxxx* name by using the PARM option as follows:

   ```
   -DSN1 START DB2,PARM(DSNZPxxx)
   ```

   If you have not run job DSNTIJRT, and the administrative task scheduler starts, the administrative task scheduler issues error message DSNA679I to report SQLCODE -805 until you run job DSNTIJRT. The administrative task scheduler starts if you specified an address space procedure name in the ADMIN SCHEDULER field on installation panel DSNTIPX.

If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, possibly *ssnm*DIST, possibly *ssnm*ADMT, and *irlmproc*, where *ssnm* is the DB2 subsystem name, *ssnm*ADMT is the DB2 administrative task scheduler procedure name, and *irlmproc* is the IRLM procedure name.

If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

```
DSNR002I   RESTART COMPLETED
DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
```

When you start DB2 Version 10 for the first time, DB2 issues message DSNT501I with reason code 00C900A6. This message is expected. When you run job DSNTIJTC, the cause of this message is corrected.

You must run job DSNTIJTC to complete the tailoring of the DB2 catalog. In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

After you start DB2, identify unusual conditions for databases with the command:

```
-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
```

If DB2 does not start successfully, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Ensure that the DSNTIJID, DSNTIJIE, DSNTIJIF, DSNTIJIN, and DSNTIJUZ jobs ran correctly. Also, verify that the subsystem parameter member that you specified (or allowed by default) when you started DB2 is the one that is built by job DSNTIJUZ. Check the JCL for the DB2 startup procedure.

If you use distributed data facility, you must define VTAM before DDF can start. However, you do not need to have TCP/IP configured to start DDF. In addition, transactions such as those from DDF or CICS may fail because work files have not yet been defined.

3. Optional: Start TSO. After DB2 starts, it is available to TSO. If you want to use the TSO SUBMIT command to do housekeeping and installation verification, you must start TSO (if it is not already started).

Creating the default storage group, defining temporary work files, and binding DB2 REXX Language Support

After you start DB2 as part of the process for enabling data sharing, you must create the default storage group for user-defined DB2 tables that are not specifically assigned to a storage group. You must also define the database for temporary work files, and bind DB2 REXX Language Support.

Procedure

To create the default storage group, define temporary work files, and bind DB2 REXX Language Support:

Run job DSNTIJTM.
If job DSNTIJTM runs successfully, it produces the return codes that are shown in the following table:

Table 96. DSNTIJTM return codes

Step	PROCSTEP	Return code
DSNTIAD		
	PC	0000
	ASM	0000
	LKED	0000
DSNTIAS	IKJEFT01	0000
DSNTICR	IKJEFT01	0000
DSNTIST	IKJEFT01	0000

Results

DSNTIJTM assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically. It also defines the initial buffer pool sizes that are specified on installation panels DSNTIP1 and DSNTIP2. When you use the ENABLE or MEMBER functions, the steps to prepare and bind the DSNTIAD program are unnecessary and are removed during editing.

DSNTIJTM also provides some cleanup to ease reruns.

The SYSDEFLT storage group is the default storage group, and is used for your database, table space, and table definitions that are not related to a specific storage group. By default, it is also used for work file table spaces that are created during DB2 installation and migration. When DSNTIJTM creates the SYSDEFLT storage group, it performs the following grants:

- Grants use of the default buffer pool and storage group to PUBLIC.
- Grants use of the SYSDEFLT table space. This table space does not really exist, but this GRANT statement is necessary to give users the ability to implicitly create table spaces in the default database.
- Grants authority to create tables and table spaces in the default database to PUBLIC.

What to do next

For data sharing installations, the work file database name is the name that was specified in the WORK FILE DB field on installation panel DSNTIPK. After creating the work file database, the DSNTIJTM job creates the number of 4 KB and 32 KB work file table spaces that was specified in the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9.

You can increase the number of additional temporary work file table spaces by increasing the values for the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9, particularly if you expect a large amount of sorting at your site. Additional temporary work file table spaces can improve DB2 performance by reducing device contention among applications. These additional work files also can be used for sorting indexes on large tables during index creation.

You can choose to have job DSNTIJTM create these additional table spaces, or you can create them after you run DSNTIJTM. To create additional work file table spaces after you run job DSNTIJTM, use the DSNTWFG exec in job step DSNTIST. See the comment block for job step DSNTIST for information about the parameters for DSNTWFG.

If you do not run the installation CLIST, customize DSNTIJTM or the DSNTWFG exec to create at least one 4 KB and one 32 KB work file table space.

Creating an image copy of the DB2 directory and catalog (optional)

If you decide to image copy the DB2 directory and catalog, use the DSNTIJIC job that was generated during the installation or migration of the originating member. You do not need to make an image copy of the DB2 catalog or user data sets for recovery because DB2 uses image copies that were made before you enabled data sharing.

Procedure

To create an image copy of the directory and catalog:

1. Examine job DSNTIJIC for the following information:
   - The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also the unit name for the default archive log data sets.
   - Expiration date or retention period. You can add a retention period or an expiration date to the job.
   - The user on the JOB statement. Ensure that the user is authorized. If DB2 is started under ACCESS(MAINT), this must be the same user that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1.
2. Run job DSNTIJIC.

   The DSNTIJIC job contains a list of all the DB2 directory and catalog table spaces. When you run job DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. Having copies of table spaces enables you to recover the DB2 catalog and DB2 directory in case of a failure.
3. If the DSNTIJIC job fails or abends, verify that no problems exist with the tape setup for image copy. If you find no problems with the tape setup, examine the utility job output (JOBLOG) or the console log for problems. For example, look for I/O errors or incorrect sizes.

What to do next

Run the DSNTIJIC job periodically, perhaps daily or weekly, to reduce the amount of time required for recovering the directory or catalog. The copied data and log data sets are needed for recovery.

During the ISPF tailoring session, you named one or two IDs to have installation SYSADM authority. One of these users can now grant various levels of authority to other users. You can use SPUFI or a job similar to DSNTIJSG to perform the authorization. To grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF, issue the following statement in SPUFI:

```
GRANT SYSOPR TO userid
```

Verifying that enabling data sharing was successful (optional)

If you ran the complete set of installation verification procedure (IVP) sample jobs after you migrated to Version 10, you should not need to run these jobs again.

About this task

When you start the originating member, DB2 checks your coupling facility, group, and member definitions and verifies that data sharing is enabled. You can also verify that the group has been correctly established by issuing the DB2 command DISPLAY GROUP after the originating member has completed startup.

Procedure

To verify that enabling data sharing was successful:

Run the verification jobs that are described in Chapter 7, “Verifying that installation or migration was successful with the sample applications,” on page 743.

Adding a new member

After you install a new data sharing group or enable an existing DB2 subsystem for data sharing, you can add new data sharing members. Always add a member as a new installation. After being added to the group, the new member uses the DB2 catalog of the originating member.

About this task

When adding a new member to a group, you might need to make changes during installation to allow more XCF groups, or more XCF members per group. Or, you might need to “widen” the locks in the IRLM lock structure (for example, if the structure was initially allocated with a maximum of seven members, and the eighth member is joining the group).

DB2 does not have an automatic way to “merge” catalogs and resolve naming conflicts. If you have applications that currently run on several existing DB2 subsystems, your migration plan might include procedures for moving the relevant data and applications from those DB2 subsystems onto one or more of the group members. Your migration plan should also include procedures for resolving any catalog naming conflicts that result.

Related concepts:

“Merging existing DB2 data into the group” on page 703

Related tasks:

“Installing DB2” on page 507

Related information:

z/OS IEFSSNxx member (MVS Initialization and Tuning Reference)

Running the CLIST for adding a new member

The process of adding a new data sharing member involves running some of the installation jobs. Before you run the installation jobs, tailor them to your environment by running the installation CLIST.

About this task

Jobs that the add-member CLIST tailors:

DSNTIJIN	DSNTIJMV	DSNTIJID	DSNTIJDE
DSNTIJTM	DSNTIJUZ	DSNTIJFT	DSNTIJGF

Procedure

To tailor the installation jobs for adding a new member:

1. Invoke the installation CLIST.
2. On installation panel DSNTIPA1, specify:

```
INSTALL TYPE ===> INSTALL
DATA SHARING ===> YES
.
.
INPUT MEMBER NAME  ===> originating member's output PDS member
OUTPUT MEMBER NAME ===> new member's output PDS member
```

3. On installation panel DSNTIP0A, specify option 2 for adding a member.
4. On installation panel DSNTIPK, specify a new member name:

```
MEMBER NAME ===> new member name
```

   Verify that the *new member name* is unique within the z/OS Parallel Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the member name that you specify on panel DSNTIPK.
5. Complete the installation panels, specifying parameters according to the DB2 subsystem parameters guidelines.

   Recommendation: Rename the tailored SDSNSAMP data set for each member. This data set contains tailored JCL for each member, including jobs that are used for disabling and re-enabling data sharing, if that should ever become necessary. If you do not rename the SDSNSAMP data set, it is overwritten when you install a new member. Rename the data set by choosing a new name for the *prefix*.NEW.SDSNSAMP data set on installation panel DSNTIPT. For example, use *prefix.member_name*.SDSNSAMP.

Related concepts:

"Subsystem parameters and application defaults in a data sharing environment" on page 670

Related tasks:

"Invoking the CLIST" on page 224

Identifying DB2 to z/OS

Job DSNTIJMV completes some of the steps that are required to identify DB2 to z/OS. These steps include updating members of SYS1.PARMLIB and SYS1.PROCLIB and creating Language Environment options data sets for the DB2 core WLM environment for Java routines.

About this task

Do not overwrite the z/OS-supplied entries for DB2 and IRLM in the program properties tables (PPT) in SYS1.PARMLIB(SCHED*xx*). The operating system supplies default PPT values for all DB2 modules that require them. You should not change these values.

Procedure

To define DB2 to z/OS:

1. Run job DSNTIJMV. If job DSNTIJMV runs successfully, it produces return codes of 0.
2. Review the content of the Language Environment options data set and adjust the following environment variables as needed. The data set is created by DSNTIJMV and must be allocated by the JAVAENV DD of the address space procedure for the DB2 core WLM environment for Java routines.
   - _CEE_ENVFILE, if present, needs to be set to the name and path of the environment variable file that was created by job step DSNTIJR.
   - JVMPROPS needs to be set to the name and path of the JVM startup options file that was created by job step DSNTIJS.
   - DB2_BASE and JCC_HOME need to be set to the names of the HFS directories that you selected for the DB2 for z/OS and JCC product classes.

What to do next

z/OS requirements: Each DB2 and each IRLM that you define to z/OS in the IEFSSN*xx* PARMLIB member requires a z/OS system linkage index (LX). The default number of these indexes that z/OS reserves is 165. If you place all of your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 165 LXs; otherwise, your subsystems might not start. If you need more than 165 LXs, use the NSYSLX option on the z/OS IEASYS*xx* PARMLIB member to increase this number.

For later diagnosis of IRLM problems, also ensure that the following requirements are met:

- Specify the IRLM dump formatting modules DXRRLM50, DXRRLM55, and DXRRLS55 in a BLSCUSER member in a data set that is in the system's PARMLIB concatenation.
- Copy the DXRRL186, DXRRLFTB, DXRRLM50, DXRRLM55, and DXRRLS55 modules from the IRLM SDXRRESL data set to a data set in the system's link list. Or, add the SDXRRESL data set to the system's link list in a PROG*xx* member of a data set in the system's PARMLIB concatenation. If you use more than one level of IRLM, you will need to use a STEPLIB concatenation when using IPCS to process dumps that are taken for the levels whose modules are not in the link list.

Additional changes to SYS1.PARMLIB and SYS1.PROCLIB: Because different sites have different requirements for identifying DB2 to z/OS, job DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes to SYS1.PARMLIB and SYS1.PROCLIB might be incomplete. You might have additional procedures of your own to rename. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB, or by editing DSNTIJMV.

Recommendation: Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV step. For SYS1.PROCLIB, submit the procedure-update section of job DSNTIJMV. Before you make the updates, read the following information and examine job DSNTIJMV to study the updates that it makes. Then use an editor such as ISPF/PDF to make the updates to SYS1.PARMLIB.

Related concepts:

Runtime environment for Java routines (Application Programming for Java)

Defining system data sets

When job DSNTIJIN is tailored for adding a new member, it defines the new BSDS and active log data sets to use SHAREOPTIONS (2 3). It does not define catalog and directory data sets.

Procedure

To define system data sets:

1. Check the DEFINE CLUSTER statements in job DSNTIJIN to ensure that they allocate adequate disk space for your system.

   Recommendation: For recovery purposes, place system data sets such as the DB2 recovery log and the VSAM catalog on different disk volumes. Because these data sets are used frequently, do not migrate them by using DFSMShsm.
2. Run job DSNTIJIN.

   If DSNTIJIN runs successfully, it produces return codes of 0 for all DEFINE statements and steps. Check any VSAM messages carefully.
3. If job DSNTIJIN fails or abends, complete the following steps:
   a. Remove the z/OS catalog delete statements from job DSNTIJDE.
   b. Run job DSNTIJDE. This job deletes the previously created data sets for the DB2 directory and DB2 catalog.
   c. Run DSNTIJIN again.

Related concepts:

Allocating and extending data sets (DB2 Performance)

Defining DB2 initialization parameters

Job DSNTIJUZ generates the DB2 data-only load module DSNHMCID, the application defaults load module (*dsnhdecp*), and the subsystem parameter module. The application defaults load module is the name that was specified for the APPL DEFAULTS field on installation panel DSNTIPO3. The subsystem parameter module is the name that was specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

Procedure

To define DB2 initialization parameters:

1. If you added a STEPLIB statement to the DB2 start procedures, modify the SYSLMOD steps to point to the library in the STEPLIB statement instead of to the library in *prefix*.SDSNEXIT.
2. If the DB2 distribution library prefix is different from the target library prefix, edit DSNTIJUZ to correct the data set name for *prefix*.ADSNLOAD.
3. If you have not run the SMP/E ACCEPT job (DSNACEP1) of FMID HDBAA10, edit DSNTIJUZ to include the SMP/E temporary data set (SMPTLIB) in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. SMPTLIB is *hlq*.HDBAA10.F2, where *hlq* is from the GLOBAL SMP/E zone. Use the following SMP/E statements to get DSPREFIX:

   ```
   SET   BOUNDARY (GLOBAL).
   LIST  DDDEF ( SMPTLIB ).
   ```

4. Insert the DSPREFIX value after SDSNLOAD and ADSNLOAD.
5. Run job DSNTIJUZ.

   If you want DSNTIJUZ to establish SMP/E GENASM entries in the DB2 system parameter (DSN6*xxxx*) and application defaults module macros, you need to uncomment job step DSNTIMQ. You might receive message GIM65001 when you run steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when you run step DSNTIMQ. You can ignore these messages.
6. If job DSNTIJUZ fails or abends, correct the problem and rerun the job.

Related reference:

"PARAMETER MODULE field" on page 358

"APPL DEFAULTS field" on page 359

Initializing system data sets

If you are adding a new data sharing member, you must initialize your system data sets after you define your system data sets and DB2 initialization parameters.

Procedure

To initialize system data sets:

Run job DSNTIJID.

- If the job runs successfully, it produces a return code of 0. Otherwise, check for and resolve error conditions that might be reported by VSAM messages.
- If the job fails or abends, remove the ICF catalog delete statements (job step DSNTIVC) from job DSNTIJDE. Then, rerun jobs DSNTIJDE, DSNTIJIN, DSNTIJUZ, and DSNTIJID.

Results

Job DSNTIJID records the active log data set names to the BSDS, formats the active log data sets, and initializes the DB2 directory table spaces and indexes.

Defining user authorization exit routines (optional)

When you add a new data sharing member, you should not have to make changes to the DB2 exit routines. However, if the new member is using its own separate load module or exit library, you must define the exit routines in the new library.

Procedure

To define user authorization exit routines:

1. Modify job DSNTIJEX to reference the correct library.
2. Run job DSNTIJEX.

   If job DSNTIJEX runs successfully, it produces return codes of 4.

Related concepts:

Connection routines and sign-on routines (Managing Security)

Recording DB2 data to SMF (optional)

When you add a DB2 data sharing member, you can specify whether DB2 statistical, accounting, and audit trace data is to be collected.

Procedure

To specify the data that is to be collected:

1. Choose from the following options:

Option	Description
To have DB2 collect statistical information	Accept the default (YES class 1) for the SMF STATISTICS option on installation panel DSNTIPN. To collect statistical information for deadlock or timeout, specify class 3. To collect information about DDF error conditions, specify class 4.
To have DB2 collect accounting information	Accept the default (1) or specify * (all classes) for the SMF ACCOUNTING option on installation panel DSNTIPN.
To have DB2 collect auditing information	Specify * (all classes) for the AUDIT TRACE option on installation panel DSNTIPN.

   In all cases, DB2 invokes a trace, passing the data it collects to the System Management Facility (SMF) of z/OS.

   DB2 also passes performance data to SMF whenever an accounting, statistics, or audit trace is successfully started or stopped. DB2 can also record other performance data. After you complete the installation process, you can use commands to have DB2 record performance data for over 230 different subsystem events.

2. If, during installation, you requested that DB2 pass accounting and statistics data to SMF, update the SMFPRM*xx* member of SYS1.PARMLIB as follows:
   - Specify the ACTIVE parameter.
   - Specify STC,NODETAIL for the SUBSYS parameter.
   - Specify the proper TYPE subparameter of the SYS and SUBSYS parameters.

What to do next

During DB2 execution, you can use the SMF SET or SS command to alter the SMF parameters. For example, you can record the statistics trace class 1 IFCIDs 0001, 0002, and 0202 (SMF record type 100); accounting trace class 1 IFCIDs 0003 and 0239 (SMF record type 101); and all other DB2 trace records (SMF record type 102) to SMF. To record this information, issue the following command:

```
SYS(TYPE(100:102))
```

For DB2 to pass data to SMF, you must allocate an adequate supply of SMF buffers. The default buffer settings are probably insufficient.

You can specify SMF buffering on the VSAM BUFSP parameter of the Access Method Services DEFINE CLUSTER statement. Do not use the default settings if DB2 data is sent to SMF. Specify CISZ(4096) and BUFSP(81920) on the DEFINE CLUSTER statement for each SMF VSAM data set. These values for CISZ and BUFSP are the minimum requirement for DB2. You might need higher values for CISZ and BUFSP, depending on the requirements of all your z/OS subsystems.

You can also code an IEFU84 SMF exit routine to process the records that are produced.

Related tasks:

Controlling traces (DB2 Administration Guide)

Related reference:

-START TRACE (DB2) (DB2 Commands)

z/OS MVS Initialization and Tuning Guide

Related information:

z/OS MVS Initialization and Tuning Reference

Establishing subsystem security (optional)

DB2 can control access to data within a DB2 subsystem. DB2 also works with outside security systems, such as RACF, that control access to the DB2 subsystem.

Before you begin

The following IDs should have been specified on installation panel DSNTIPP1:

- Two system administrator (installation SYSADM) authorization IDs
- Two system operator (installation SYSOPR) authorization IDs
- Two security administrator (installation SECADM) authorization IDs or roles
- One authorization ID (installation IBMUSER) if RACF is not available for batch access and USER= is not specified in the JOB statement

Procedure

To establish subsystem security:

- Ensure that the specified installation SYSADM IDs are defined in your TSO and RACF systems before attempting to access DB2. You can also define installation SYSOPR IDs, SECADM IDs, and the installation IBMUSER ID.
- To enable primary and secondary user IDs to issue DB2 commands from the z/OS console or TSO SDSF, define RACF classes to authorize DB2 commands. Use the following RACF commands:

  ```
  SETR CLASSACT(DSNADM)
  RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
  SETR RACLIST(DSNADM) REFRESH
  PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
  SETR RACLIST(DSNADM) REFRESH
  ```

  You can grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF, but only after DB2 can process SQL.
- To separate security tasks from system administration tasks, use the SEPARATE SECURITY field on the DSNTIPP1 panel. Before setting the SEPARATE SECURITY field to YES, set at least one SECADM system parameter to an authorization ID, or create the necessary trusted contexts and roles. If you specify YES for SEPARATE SECURITY, system administrator authority cannot be used to perform security tasks, and the SECADM authority is required to manage trusted contexts and roles. If both SECADM system parameters are set to roles and those roles have not been created, no one will have the authority to manage trusted contexts and roles.
- Run job DSNTIJRA to complete the following tasks:
  - Define the administrative task scheduler started task module to RACF program control

– Define the administrative task scheduler as a trusted context in RACF

If you are using a DB2 data sharing environment, customize and run DSNTIJRA for each member of the group.

Related concepts:

Getting started with DB2 security (Managing Security)

Connecting DB2 to IMS (optional)

Connecting DB2 to IMS is an optional part of adding a new data sharing member. Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

1. Make DB2 load modules available to IMS.
2. Define DB2 to IMS.
3. Define new programs and transactions to IMS.
4. Prepare IMS applications for DB2.
5. Depending on your site, you might also need to:
   - Define DB2 plans for IMS applications.
   - Generate a user language interface.

Related tasks:

"Connecting the IMS attachment facility" on page 869

Connecting CICS to DB2 (optional)

When you add a new member, you have the option of connecting CICS to the member.

About this task

To connect DB2 to CICS, you must regenerate several CICS tables with additional entries. Coordinate this connection with your CICS support group.

Related tasks:

Defining the CICS DB2 connection

Defining DB2 to z/OS

After you run job DSNTIJMV, you need to define DB2 in SYS1.PARMLIB.

About this task

Job DSNTIJMV makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member
- Creates new load module libraries in the LNKLST*xx* member

The load module library SDSNLINK contains the early code. SDSNLINK contains modules that must be placed in the link list look-aside address space (LLA) because they are loaded at subsystem initialization during the IPL.

Procedure

To define DB2 to z/OS:

Complete one of the following actions:

- IPL z/OS.

  During the z/OS IPL, message DSN3100I appears on the z/OS console, stating that DB2 is ready for the START command.
- Use the z/OS command SETSSI to dynamically add the new DB2 and IRLM subsystems. For example, the following two commands can be used to add subsystems for DB2 and IRLM to MVS3:

  ```
  RO MVS3,SETSSI ADD,SUB=DB3A,INITRTN=DSN3INI,INITPARM='DSN3EPX,-DB3A,S,DB0A'
  RO MVS3,SETSSI ADD,SUB=DJ3A
  ```

  Recommendation: Add DB2 to the IEFSSN*xx* member so that it can be used on a subsequent IPL.

Related reference:

SETSSI Command

Starting the DB2 subsystem

During the process of adding a data sharing member, start the DB2 subsystem after you define DB2 to z/OS.

Procedure

To start DB2:

1. Start the IRLM, if you have not requested that DB2 automatically start the IRLM. You must start IRLM before you start DB2.

   Use the following command:

   ```
   START irlmproc
   ```

   where *irlmproc* is the name that you specified for the PROC NAME option on IRLM Panel 1 (DSNTIPI).

   If you specified YES for the AUTO START option on IRLM Panel 1 (DSNTIPI), DB2 starts the IRLM automatically.
2. Start DB2 from the z/OS console. Use the following command:

   ```
   -DSN1 START DB2
   ```

   where (-DSN1) is the subsystem command prefix that you defined for DB2.

   DB2 uses the subsystem parameter module that is specified in the start-up JCL procedure in SYS1.PROCLIB:

   ```
   //IEFPROC EXEC PGM=DSNYASCP,PARM='ZPARM(DSNZPxxx)', ...
   ```

   where *DSNZPxxx* is the value that you specified for the PARAMETER MODULE field on installation panel DSNTIPO3.

   If you need to change the name of the *DSNZPxxx* module, you can edit SYS1.PROCLIB. Alternatively, you can override the *DSNZPxxx* name by using the PARM option as follows:

   ```
   -DSN1 START DB2,PARM(DSNZPxxx)
   ```

   If you have not run job DSNTIJRT, and the administrative task scheduler starts, the administrative task scheduler issues error message DSNA679I to report

SQLCODE -805 until you run job DSNTIJRT. The administrative task scheduler starts if you specified an address space procedure name in the ADMIN SCHEDULER field on installation panel DSNTIPX.

If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, possibly *ssnm*DIST, possibly *ssnm*ADMT, and *irlmproc*, where *ssnm* is the DB2 subsystem name, *ssnm*ADMT is the DB2 administrative task scheduler procedure name, and *irlmproc* is the IRLM procedure name.

If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

```
DSNR002I   RESTART COMPLETED
DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
```

When you start DB2 Version 10 for the first time, DB2 issues message DSNT501I with reason code 00C900A6. This message is expected. When you run job DSNTIJTC, the cause of this message is corrected.

You must run job DSNTIJTC to complete the tailoring of the DB2 catalog. In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

After you start DB2, identify unusual conditions for databases with the command:

```
-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
```

If DB2 does not start successfully, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Ensure that the DSNTIJID, DSNTIJIE, DSNTIJIF, DSNTIJIN, and DSNTIJUZ jobs ran correctly. Also, verify that the subsystem parameter member that you specified (or allowed by default) when you started DB2 is the one that is built by job DSNTIJUZ. Check the JCL for the DB2 startup procedure.

If you use distributed data facility, you must define VTAM before DDF can start. However, you do not need to have TCP/IP configured to start DDF. In addition, transactions such as those from DDF or CICS may fail because work files have not yet been defined.

3. Optional: Start TSO. After DB2 starts, it is available to TSO. If you want to use the TSO SUBMIT command to do housekeeping and installation verification, you must start TSO (if it is not already started).

Results

Transactions might initially fail because work files are not defined until you run job DSNTIJTM.

Creating the default storage group, defining temporary work files, and binding DB2 REXX Language Support

After you start DB2 when you add a data sharing member, you must create the default storage group for user-defined DB2 tables that are not specifically assigned to a storage group. You must also define the database for temporary work files, and bind DB2 REXX Language Support.

Procedure

To create the default storage group, define temporary work files, and bind DB2 REXX Language Support:

Run job DSNTIJTM.
If job DSNTIJTM runs successfully, it produces the return codes that are shown in the following table:

Table 97. DSNTIJTM return codes

Step	PROCSTEP	Return code
DSNTIAB	(none)	0000
DSNTIAS	(none)	0000
DSNTICR	(none)	0000
DSNTIST	(none)	0000

Results

DSNTIJTM assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically. It also defines the initial buffer pool sizes that are specified on installation panels DSNTIP1 and DSNTIP2. When you use the ENABLE or MEMBER functions, the steps to prepare and bind the DSNTIAD program are unnecessary and are removed during editing.

DSNTIJTM also provides some cleanup to ease reruns.

The SYSDEFLT storage group is the default storage group, and is used for your database, table space, and table definitions that are not related to a specific storage group. By default, it is also used for work file table spaces that are created during DB2 installation and migration. When DSNTIJTM creates the SYSDEFLT storage group, it performs the following grants:

- Grants use of the default buffer pool and storage group to PUBLIC.
- Grants use of the SYSDEFLT table space. This table space does not really exist, but this GRANT statement is necessary to give users the ability to implicitly create table spaces in the default database.
- Grants authority to create tables and table spaces in the default database to PUBLIC.

What to do next

For data sharing installations, the work file database name is the name that was specified in the WORK FILE DB field on installation panel DSNTIPK. After creating the work file database, the DSNTIJTM job creates the number of 4 KB and 32 KB work file table spaces that was specified in the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9.

You can increase the number of additional temporary work file table spaces by increasing the values for the TEMP 4K TBL SPACES and TEMP 32K TBL SPACES fields on panel DSNTIP9, particularly if you expect a large amount of sorting at your site. Additional temporary work file table spaces can improve DB2 performance by reducing device contention among applications. These additional work files also can be used for sorting indexes on large tables during index creation.

You can choose to have job DSNTIJTM create these additional table spaces, or you can create them after you run DSNTIJTM. To create additional work file table

spaces after you run job DSNTIJTM, use the DSNTWFG exec in job step DSNTIST. See the comment block for job step DSNTIST for information about the parameters for DSNTWFG.

If you do not run the installation CLIST, customize DSNTIJTM or the DSNTWFG exec to create at least one 4 KB and one 32 KB work file table space.

Related concepts:

"Work file database storage requirements" on page 35

Creating an image copy of the DB2 directory and catalog

After you add a data sharing member, you need to create a backup copy of the DB2 directory and catalog. Use the DSNTIJIC job that was generated during the installation or migration of the originating member to create the image copy.

Procedure

To create an image copy of the directory and catalog:

1. Examine job DSNTIJIC for the following information:
   - The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also the unit name for the default archive log data sets.
   - Expiration date or retention period. You can add a retention period or an expiration date to the job.
   - The user on the JOB statement. Ensure that the user is authorized. If DB2 is started under ACCESS(MAINT), this must be the same user that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1.
2. Run job DSNTIJIC.

   The DSNTIJIC job contains a list of all the DB2 directory and catalog table spaces. When you run job DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. Having copies of table spaces enables you to recover the DB2 catalog and DB2 directory in case of a failure.
3. If the DSNTIJIC job fails or abends, verify that no problems exist with the tape setup for image copy. If you find no problems with the tape setup, examine the utility job output (JOBLOG) or the console log for problems. For example, look for I/O errors or incorrect sizes.

What to do next

Run the DSNTIJIC job periodically, perhaps daily or weekly, to reduce the amount of time required for recovering the directory or catalog. The copied data and log data sets are needed for recovery.

During the ISPF tailoring session, you named one or two IDs to have installation SYSADM authority. One of these users can now grant various levels of authority to other users. You can use SPUFI or a job similar to DSNTIJSG to perform the authorization. To grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF, issue the following statement in SPUFI:

```
GRANT SYSOPR TO userid
```

Related concepts:

Privileges and authorities (Managing Security)

Verifying that adding a data sharing member was successful

When you complete the procedure to add a new data sharing member, you should verify that the member was added successfully.

Procedure

To verify that adding a data sharing member was successful:

Run the verification jobs that are described in Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743.

What to do next

After you successfully add all the data sharing members for the group, you should Test the data sharing group.

Merging existing DB2 data into the group

Although DB2 cannot automatically merge catalogs, you can merge existing DB2 subsystems into a data sharing group. Consider carefully a decision to merge existing DB2 subsystems.

Merging is a very complicated process. It involves not only the physical issue of moving data, but also many other management issues, including:

- Naming conventions for users, plans, packages, databases, tables, views, and more
- Authorization techniques
- Backup and recovery conventions
- Availability practices

Before you consider merging existing DB2 subsystems into a single data sharing group, ask yourself the following question: Why are the DB2 subsystems separate now?

Merging subsystems

You should use location names that are used in applications when you merge existing subsystems into a single data sharing group.

About this task

Recommendation: Remote applications use a location name to access the DB2 subsystem, so by using a location alias, remote applications do not have to change aliases during the merging processes. You can define the previous location names of each subsystem as location alias names for the group. When the subsystem is merged, and the location name aliases are defined, the remote applications continue to work and do not have to change the location names in order to connect.

Procedure

To merge existing subsystems into a single data sharing group:

1. Choose one of the subsystems to be the originating member.
2. Move data and data definitions from the other DB2 subsystems to the originating member.
3. Use the new member installation process to add those other DB2 subsystems to the group.

Related tasks:

"Adding a new member" on page 691

Merging data

If you have an application that is currently running on independent DB2 subsystems, you might decide that the application will work well in a data sharing group.

In that case, you must move the data and merge the catalog definitions for that data from the independent DB2 subsystems into the data sharing group. Because the DB2 subsystems still exist, you cannot reuse their subsystem names when installing the subsystems as new members into the data sharing group.

DB2 does not provide an automated way to move catalog definitions from an independent DB2 subsystem into the catalog of the data sharing group. If you have procedures and tools to move applications from test to production, or to handle merging databases from enterprise reorganizations or mergers, those same procedures can be used to move applications into the data sharing group.

Existing distributed applications

If you move existing data to the data sharing group, it is likely that the location name of objects will change.

Existing distributed applications that remotely reference an object by its three-part name must be changed to reference the new name. Aliases that exist on that table must also be dropped and recreated with the new location name.

Applications that contain explicit SQL CONNECT statements that reference an old location name must be modified. Any DB2 plan that uses an old location name for the CURRENTSERVER keyword must be bound again.

Moving data that you are merging

In the context of DB2 data sharing, a *target* refers to the DB2 subsystem to which you are moving data (the data sharing group). A *source* refers to the DB2 subsystem from which you are moving data.

Procedure

To move data that you are merging:

1. Decide the method that you will use to move data. Three options are outlined below; you can use a combination of the specified methods. Each of the methods assumes that the objects that are to be created in the target system are created with the exact DDL that is used in the source system. Descriptions of methods 1, 2, and 3 are referred to throughout the procedure, and are described in detail below.

- **Method 1: Unload and Reload**: The data that is to be moved is unloaded from the source DB2 subsystem and loaded into the target data sharing group by using the DB2 online utilities, UNLOAD and LOAD. This approach is the most simple because tables are created without concern for object IDs, and data is unloaded or reloaded using conventional techniques. The major disadvantage of this method is that it is the slowest of the three methods, in terms of performance. The unload and reload methods operate against rows of a table, whereas the other techniques operate against pages or data sets as their level of granularity. Additionally, this method requires two to three times the disk space that the other methods require, depending on how you unload or reload (source, target and intermediate unload).
- **Method 2: DSN1COPY with OBID translation**: The data that is to be moved is copied from the source system to the target system using the DB2 offline utility DSN1COPY. DSN1COPY is usually faster than unload and reload because it moves pages of data, rather than rows. This method is more complicated because the object IDs in the source data will need to be translated.
- **Method 3: Use DB2 VSAM data sets from the source system in the target system**: The VSAM data sets that are used to hold the table space and index space data from the source system are used in the target system, without copying any data. Because this method does not require copying the data, it is the fastest of the three methods, in terms of performance. It is, however, the most complicated. Another advantage of this method is that it requires less disk space than the other methods; this method is generally used for very large table or index spaces. Method 3 assumes that all of the disk volumes in the source system used by objects being moved by this method are shared and accessible with the target system.

2. Choose a catalog that belongs to one of the DB2 subsystems to be the "original" catalog for the data sharing group. This DB2 subsystem is considered the originating member of the group.

   You should consider many different factors when choosing which DB2 subsystem to be the originating member of the group. For example, it makes sense to choose the member with the most database objects as the originating member in order to minimize the number of objects that you move.

   However, if all DB2 subsystems are mostly equivalent, and you are not planning to use DSN1COPY, consider the log RBA values of the existing subsystems. Compare the end-of-log RBAs with the high-order 6-bytes of the time-of-day clock timestamp on each of their systems (this is called the *truncated timestamp*). The usual case is that the RBA is less than the truncated timestamp. In this case, any DB2 subsystem is chosen as the originating member.

   In the event that the current end-of-log RBA in any of the existing DB2 subsystems is higher than the 6-byte truncated timestamp value at the time you are ready to enable sharing, you have the following choices:

   - Choose the DB2 subsystem that has the highest RBA as the originating member.
   - Use DSN1COPY with the RESET option to reset the log RBAs in each data and index page to 0 when you move databases from other DB2 subsystems to the data sharing group.
3. Resolve name conflicts among the objects and authorization IDs in the data sharing group.
4. Create the objects on the target subsystem.

When creating objects while you are using methods 1 or 2 to move the data, the CREATE statements can be entered in any order. DB2 assigns new OBIDs for these objects.

If you are using method 3, you must perform the following steps:

a. Consider creating the data sets with the minimum size settings to save space. The actual target system VSAM data sets created by the DDL execution are not used because the VSAM data sets that are from the source system are used instead. By choosing to create the objects with minimum sizes, you eventually need to issue ALTER TABLESPACE and ALTER INDEX statements to change the PRIQTY and SECQTY after the CREATE. If, however, you are using the same high-level qualifier for the source and target systems (which is not recommended), you must create the objects with DEFINE NO, and take additional steps later. DEFINE NO is needed when keeping the same high-level qualifier, because the VSAM data set name would be identical in the source and target systems. In this case, DB2 would not be able to define the data set on the target system during the create processing because the data set already exists.
b. Query the SYSIBM.SYSTABLES table in the DB2 catalog on the source subsystem to get the table OBID for tables that are within the databases that are being moved. Additionally for both the source and target systems, the DBID and PSID are needed for table spaces (from SYSIBM.SYSTABLESPACE) and the DBID and ISOBID are needed for indexes (from SYSIBM.SYSINDEXES). These will be used to build REPAIR jobs to modify these IDs in the header page, and potentially the first space map page in a later step.
c. Use the OBID clause on the CREATE TABLE statement on the target subsystem to specify an OBID that is the same as the table OBID on the source subsystem. Verify that the OBID that you specify is available. If an OBID is being used for another object within the same database, such as an index or referential constraint, DB2 does not allow you to create the table with the specified OBID.

Recommendation: To help guarantee the availability of OBIDs for all tables within a database, defer the creation of all indexes and referential constraints until all tables are created. All CREATE TABLE statements must have the OBID clause to guarantee that they are assigned the correct OBID. If an explicit table space name is specified, then the CREATE TABLESPACE statement must come immediately before the first CREATE TABLE statement for that table space. This helps prevent a group of CREATE TABLESPACE statements from using up OBIDs that are needed for the tables.

For methods two or three, the objects in the target system will be created with several assumptions that might require the additional steps that are outlined below:

- The VSAM data sets for each new object are created using the 'I' prefix for each of the data sets. It is possible that on the source system online REORG has executed, causing the data sets to have a 'J' prefix. If this is the case, you will need to code the DSN1COPY for method two, to have the correct 'J' data set designation or rename the data set back to the 'I' prefix for method three in a later step. You should query the source catalog for column IPREFIX in SYSIBM.SYSTABLEPART for table spaces, and SYSIBM.SYSINDEXPART for indexes, to find any 'J' prefix objects.
- If any of the index names involved in the move are greater than eight characters in length, DB2 generates a unique INDEXSPACE name to be used in naming the VSAM data set. The INDEXSPACE name generated will

most likely be different in the source and target systems. You should query the source and target catalogs to get a list of the INDEXSPACE names for the indexes that have a name greater than eight characters (found in SYSIBM.SYSINDEXES). These will be used in a later step.

5. On the source DB2 subsystem, run the REORG utility on any table spaces for which a table alteration has occurred with no subsequent REORG.

   If you are unsure if a table meets this criteria, query SYSIBM.SYSTABLES for those tables in which CREATEDTS does not equal ALTEREDTS. However, there is no way to tell from the DB2 catalog what kind of ALTER was performed and whether it would have affected this process. By comparing the CREATEDTS and ALTEREDTS, you will at least gather a list of candidates. If all rows have CREATEDTS equal to ALTEREDTS, running the REORG utility is unnecessary.

6. Move the data choosing one of the three methods that match the ones listed above:
   - **Method 1: Unload / Reload**:
     a. Start the objects on the source system in RO mode.
     b. Execute the UNLOAD utility on the source system to unload the data to a sequential file.
     c. Execute the LOAD utility on the target system to load the data that was unloaded when performing the step listed above.
   - **Method 2: DSN1COPY with OBID translation** :
     a. Stop the object, for both table space and index, on the source and target systems.
     b. Execute DSN1COPY with OBID translation and the RESET option to copy the data from the source system to the target system. If the online REORG utility has been run, be sure to code the proper I/J data set names in the DSN1COPY job. Also, if any indexes were created with names greater than eight characters long you will need to incorporate the INDEXSPACE name differences retrieved in an earlier step.
     c. Start the objects on the target systems R/W.
   - **Method 3: Use DB2 VSAM data sets from source system in the target system**:
     a. Stop the table spaces and index spaces in the target system.
     b. If the high level qualifier (HLQ) is not changing between the source and target systems, which is not recommended, then perform the following steps on the target system. Note that this process assumes that the objects were created in the target system with DEFINE NO utility.
        1) Change SPRMCTU in the DSN6SPRM macro to one. This enables the catalog to be updated. Also, change DLDFREQ to zero, which disables the down level detection. You must note the original value, because you will change DLDFREQ back to the original value in a later step. Assemble the changed zparm and restart the target system to pick up the change.
        2) Using SQL, update the SPACE column from SYSIBM.SYSTABLEPART and SYSIBM.SYSINDEXPART from -1 to 0 for the table and index spaces that you are moving using method three.
        3) Run the REPAIR DBD REBUILD utility for the databases created earlier. This step is needed because the above step changed the SPACE value in the DB2 catalog from -1 to 0, but the value is also in the DBD. This means that when the REPAIR DBD REBUILD utility is

run, it will take the information from the catalog and rebuild the DBD in the directory, making DB2 think that the data set has already been defined.

Note: Do not try to access the tables at this point. Both systems will think that the data set is theirs, but only one system will actually be able to access the data set. The REPAIR DBD REBUILD utility only should be executed after all table spaces and index spaces that are being moved by method three for a particular database, have their catalogs updated via SQL.

4) Change SPRMCTU back to zero, and then DLDFREQ to the previous value it was assigned. Next, reassemble the zparm and recycle the target system so it recognizes the change.

c. Delete the target system VSAM data sets for the table spaces and indexes, unless they were created with DEFINE NO, because the HLQ did not change.

d. Use the REPAIR utility on the source system to change the identifiers in the page set header page, or header pages, if you have a partitioned page set, to match the new identifiers on the target subsystem.

   The identifiers consist of two, 2-byte fields, HPGDBID and HPGPSID. You must locate and replace these identifiers with the new DBID and PSID of the target system as follows:

   – For non-partitioned page sets, locate and replace the 4-byte fields starting at X'0C', which begins on page zero.
   – For partitioned page sets, locate and replace the 4-byte fields starting at X'0C' on page zero for each partition.

e. If the table space that you are working with is segmented and compressed, and a dictionary exists, you must also run the REPAIR utility on the source system to change the OBID in the first space map page.

f. Stop the object, for table space and index, on the source system.

g. If the HLQs of the source and target systems are different, you must run the IDCAMS ALTER statements. Running the IDCAMS ALTER statements changes the HLQ of the VSAM data sets from the source system to be the new HLQ of the target system. If the HLQ is changing, then the VSAM datasets of the target system must be first deleted before running IDCAMS ALTER. Additionally, if you know from completing one of the previous steps that the online REORG utility has been executed on the source system, make sure that you change any 'J' data sets to match the target system. The target system should have all 'I' data sets, as they were newly created. Also, if any indexes were created with names greater than eight characters long, you will need to incorporate the INDEXSPACE name differences retrieved in an earlier step.

h. Start the table spaces and index spaces in the target system.

i. Use the REPAIR utility with the LEVELID option to reset the level indicator of the page sets to a neutral value on the target system.

j. Stop the object, for table space and index, on the source system.

7. You now have the option to drop the database objects on the source subsystem. If you do not choose to drop the objects at this point, and used method three to move data while keeping the same HLQ for the data sets, then you should consider altering the VCAT name in the source system to a

different invalid value. You should alter the VCAT name so that the source DB2 does not try to open the data set that is now in use by the target system.

8. Take full image copies of all data. This is the earliest time at which data recovery can occur after the merge.
9. Run the RUNSTATS utility on the target system.
10. Bind all plans and packages on the target subsystem that were bound on the source subsystem. Bind any plans and packages on the target subsystem that have changed because of name conflict resolution. Grant the appropriate authorizations to all plans and packages.

Related reference:

DSN1COPY (DB2 Utilities)

REPAIR (DB2 Utilities)

Running REPAIR to modify OBID for a compression dictionary:

Additional invocations of the REPAIR utility might be necessary if you use DB2 VSAM data sets from the source system to move data that you are merging in the target system and the table space is segmented and compressed.

About this task

If the segmented, compressed table space has a dictionary, the space map page that contains the dictionary has an OBID that is the same as the table space that was modified in the header page when you moved and merged your data. The OBID in the space map must be modified to match the new value in the header page.

Procedure

To modify the OBID in the space map to match the new value in the header page:

1. Use the DSN1PRNT utility to view the data set header page and space map. The following is an example from DSN1PRNT with the FORMAT option. On the header page, or page zero, you will see something similar to the output displayed in the following figure:

```
PAGE: # 00000000 -------------------------------------------------------------------------------
HEADER PAGE:  PGCOMB='10'X  PGLOGRBA='BC8DB9277831'X  PGNUM='00000000'X  PGFLAGS='38'X
HPGOBID='018E0002'X  HPGHPREF='000000B4'X  HPGCATRL='00'X  HPGREL='D3'X  HPGZLD='D3'X
```

Figure 77. Sample header page of DSN1PRNT command with the FORMAT option.

The following figure shows the first space map page, or page 1:

```
FIRST PART OF SEGMENTED SPACE MAP:
SEG 0001   000000020002C0 FFFF
SEG 0002   00000003000280 FFFF
SEG 0003   00000004000280 FFFF
SEG 0004   00000000000280 FFFF
SEG 0005   000000060003C0 3333
SEG 0006   00000007000380 3333
SEG 0007   00000008000380 3333
SEG 0008   00000009000380 3333
SEG 0009   0000000A000380 3333
SEG 000A   0000000B000380 3333
SEG 000B   0000000C000380 3333
SEG 000C   0000000D000380 3333
SEG 000D   0000000E000380 3333
SEG 000E   00000000000380 3300
```

Figure 78. Sample output from first space map page of DSN1PRNT command with FORMAT option.

In the header page, HPGOBID is the DBID concatenated with the OBID. In the first figure above, HPGOBID is X'018E0002', so the DBID is X'018E' and the OBID is 0002'X. You can see that same OBID in the first four segments in the space map page, or bytes 5 and 6, following the segment number.

If you run DSN1PRNT on pages 0 and 1, and you do not see the same OBID in the space map page as the one listed on the header page, it means that the dictionary is not yet built. If the dictionary is not built, you only see the table segments (segments X'5' thru X'E' shown above). If this is the case, you do not need to modify the OBID and you can stop this task.

2. Use the REPAIR utility to modify the OBIDs in the space map to match the new OBID in the header page. Only the segments that match the OBID in the old header page must be modified. The remaining segments are for the table and do not need to be modified.

   In the second figure above, you can see that the first four segments need to be modified and that the table OBID is X'0003'.

   The number of segments that need to be modified is dependent on the SEGSIZE value of the table space. The following table shows the offset of the OBIDs that need to be modified for each SEGSIZE.

Table 98. The offset of OBIDs that need to be modified in order of SEGSIZE

SEGSIZE	Offset of OBIDs to REPAIR
4	X'20' X'29' X'32' X'3B'
8	X'20' X'2B'
12	X'20' X'2D'
16–64	X'20'

Migrating an existing data sharing group to a new release

If you have a Version 8 or Version 9.1 DB2 data sharing group, and you want a Version 10 data sharing group, you can migrate to the newer release.

Before you begin

During the migration to Version 10, other group members can be active, but they might experience delays or timeouts if they attempt to access catalog objects that are being updated or that are locked by migration processing. Before beginning the migration process, confirm that Version 8 or Version 9.1 is at the proper service level. Also, check your coupling facilities to ensure that the appropriate service levels are installed. Having the wrong service levels installed can result in data corruption. No specific service level requirements exist for CFLEVELs other than 12, but CF levels 13, 14, 15, and 16 are recommended for additional function and performance improvements. Use the z/OS **D CF** command to display the service levels for IBM coupling facilities.

Important: Migration to DB2 Version 10 is only supported from Version 8 or Version 9.1.

Before you start to migrate your data sharing group:

- Read the information about migration considerations in "Changes that might affect your migration from Version 8" on page 65 or "Changes that might affect your migration from Version 9.1" on page 114.
- Read the information in "Release coexistence in a data sharing environment" on page 203.

Related concepts:

"The scope and uniqueness of DB2 subsystem parameters" on page 671

Migrating the first member of a group to Version 10

You must first complete the migration of one member of the data sharing group to Version 10 before starting any other members at Version 10. You may choose any member as the first member to migrate to the new release.

About this task

The DB2I CLISTs are edited when the first member of a data sharing group migrates to Version 10. The sample jobs are edited when the first member migrates to Version 10, and they are also edited as part of preparation for enabling-new-function mode for the group.

Jobs that the migration CLIST tailors: Jobs that are marked with an asterisk (*) are modified only when you migrate the first member of a data sharing group.

DSNTIJEX*	DSNTIJGF	DSNTIJRI	DSNTIJTM
DSNTIJIC*	DSNTIJIN*	DSNTIJSG*	DSNTIJUZ
DSNTIJFT	DSNTIJMV	DSNTIJVC*	DSNTIJRT*
DSNTIJFV	DSNTIJPM	DSNTIJTC*	DSNTIJRV*

Follow these directions carefully. The first member of the data sharing group uses DSNTIDXA as its input member name. If you use SAP, specify DSNTIDXB, the SAP-specific CLIST input member, instead of DSNTIDXA. A subsequent member **must** use a previous member's output member name as its input member name.

Procedure

To migrate the first member of a data sharing group to Version 10:

1. Make a plan to migrate the data sharing group in as short a time period as possible. Migrating an existing data sharing group requires careful planning.
2. Ensure that maintenance through the Version 10 fallback SPE is applied to all active members.

   You must apply the fallback SPE (and its prerequisite APARs) before attempting to migrate any member of the group. Stop and restart each member to activate the change.

   Version 10 members cannot start if any one of the active Version 8 or Version 9.1 members does not have the fallback SPE applied.

   To prepare for fallback, keep the subsystem parameter load module that is used by Version 8 or Version 9.1.
3. For the first member to migrate, specify the following settings on installation panel DSNTIPA1:

   ```
   INSTALL TYPE ===> MIGRATE
   DATA SHARING ===> YES
   .
   DATA SET(MEMBER)NAME ===> this member's output member from V8 or V9
   .
   INPUT MEMBER NAME  ===> DSNTIDXA or DSNTIDXB
   OUTPUT MEMBER NAME ===> this member's output PDS member
   ```
4. On installation panel DSNTIPP0B, specify 1 to indicate that this is the first member of the group to migrate.
5. Complete the installation panels. Specify parameters according to the DB2 subsystem parameters guidelines.
6. Complete all of the migration steps.

Related concepts:

"Subsystem parameters and application defaults in a data sharing environment" on page 670

Migrating subsequent members of a group to Version 10

After you complete the migration of one member of a data sharing group to Version 10, you can migrate subsequent members of the group.

About this task

To migrate subsequent members of a group to Version 10, complete these tasks for each member:

Running the CLIST for migrating data sharing members

The process of migrating data sharing members to Version 10 involves running some of the migration jobs. Before you run the migration jobs, tailor them to your environment by running the installation CLIST.

Before you begin

Be sure to complete the process of migrating the first member of the group before you begin to migrate subsequent members to Version 10.

Procedure

To tailor the installation jobs for migrating data sharing members:

1. Invoke the installation CLIST.
2. On installation panel DSNTIPA1, specify the following:

```
INSTALL TYPE ===> MIGRATE
DATA SHARING ===> YES
.
DATA SET(MEMBER)NAME ===> this member's output member from V8 or V9
.
INPUT MEMBER NAME  ===> first member's output PDS member
OUTPUT MEMBER NAME ===> this member's output PDS member
```

3. Specify 2 on installation panel DSNTIPP0B to indicate that this is not the first member of the group to migrate.
4. Complete the installation panels, specifying parameters according to the DB2 subsystem parameters guidelines.

Related concepts:

"Subsystem parameters and application defaults in a data sharing environment" on page 670

Related tasks:

"Invoking the CLIST" on page 224

Connecting DB2 to TSO

Before you migrate a data sharing member, you must connect DB2 to TSO to support the interactive component of DB2 (DB2I). This process involves making DB2 load modules available to TSO and batch users and making DB2I panels, messages, and load modules available to ISPF and TSO.

About this task

DB2I uses the ISPF PROFILE and SHARED variable pools for most panel variable fields. As a result, you can easily re-enter a panel when panel variables have previously been specified. For the DB2 subcommands that permit lists of plan names, package names, DBRMs, and ENABLE and DISABLE statements, DB2I provides ISPF to contain all the user-specified variables for these subcommand keywords.

DB2I creates and maintains a set of ISPF tables in a user-defined TSO data set that is allocated to a data set with a ddname of DSNETBLS. The DB2I-generated tables in this library are DSNCONNS, DSNDBRMS, and DSNPLPKN. The following table shows the library table member names and their contents.

Table 99. The DB2 ISPF table library

DSNCONNS	ENABLE or DISABLE connection type and connection name variables that are referenced by plan or package name
DSNDBRMS	Subcommand DBRM member and LIBRARY name variables that are referenced by plan name
DSNPLPKN	Package list variables that are referenced by package name
DSNPATHS	Schema name variables that are referenced by plan or package name and that are to be included in the PATH keyword

DB2I uses ISPF table services to maintain individual ISPF tables within the DSNETBLS data set. For performance reasons, ISPF keeps this table library in an

open state once an individual table has been updated. Attempts to close this data set by using the TSO FREE command results in error message IKJ56861I.

Procedure

To connect DB2 to TSO:

1. If you have not included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, add JOBLIB or STEPLIB statements to your logon procedures and JCL to ensure that you access the Version 10 load modules. Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in your JOBLIB or STEPLIB concatenations.
2. If you added the DB2I libraries as part of the step to prepare for running the DB2 installation CLIST, you can skip this step. Otherwise, concatenate the DB2 ISPF libraries with the ISPPLIB, ISPTLIB, ISPSLIB, and ISPMLIB DD statements in your logon procedures. You must also concatenate the DB2 ISPF libraries with these DD statements in any of your CLISTs where the libraries might be allocated. These libraries are *prefix*.SDSNSPFP, *prefix*.SDSNSPFM, *prefix*.SDSNSPFS, and *prefix*.SDSNPFPE or *prefix*.SDSNPFP. If you use English DB2I panels, concatenate SDSNPFPE to ISPPLIB. If you use Kanji DB2I panels, concatenate SDSNPFPK instead of SDSNPFPE.

   ```
   PROFILE WTP MSGID
   ALLOCATE DDNAME(SYSEXEC) DSN('prefix.SDSNCLST') SHR REUSE
   ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
       'ISP.SISPMENU') SHR REUSE
   ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' 'prefix.SDSNPFPE' +
       'ISP.SISPPENU') SHR REUSE
   ALLOCATE DDNAME(ISPSLIB) DSN('prefix.SDSNSPFS' +
       'ISP.SISPSLIB' 'ISP.SISPSENU') SHR REUSE
   ```

3. Create the DSNETBLS data set.

   When allocating this data set, you should assign the following DCB attributes, where *n* is any integer:

   ```
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(n*LRECL)
   ```

   The following example shows how to set up an ALLOCATE statement to create the data set:

   ```
   ALLOC DA(DSNSPFT) NEW SP(1 1) TR DIR(10) +
   DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(800)
   F(DSNETBLS) REUSE
   ```

   The following example shows how to allocate an existing data set to the data set with the DSNETBLS ddname:

   ```
   ALLOC DA(DSNSPFT)      F(DSNETBLS) REUSE
   ```

4. If you want to run the ISPF/CAF sample application that is provided with DB2, ensure that the data set *prefix*.RUNLIB.LOAD is included in the logon procedures or in the ISPLLIB concatenation. If you chose IBMCOB for your LANGUAGE DEFAULT on panel DSNTIPF, you should also include the data set *prefix*.CEE.SCEERUN.

What to do next

You can attach to multiple releases of DB2 with your existing TSO or CAF logon procedures without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you must update those procedures and jobs to point to the latest level of DB2 load libraries.

Connecting DB2 to IMS (optional)

Connecting DB2 to IMS is an optional part of migrating existing data sharing members to a new release. Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

1. Make DB2 load modules available to IMS.
2. Define DB2 to IMS.
3. Define new programs and transactions to IMS.
4. Prepare IMS applications for DB2.
5. Depending on your site, you might also need to:
   - Define DB2 plans for IMS applications.
   - Generate a user language interface.

Related tasks:

"Connecting the IMS attachment facility" on page 869

Connecting CICS to DB2 (optional)

Connecting CICS to DB2 is an optional part of migrating members in a data sharing group. Connecting DB2 to CICS requires that you regenerate several CICS tables with additional entries.

About this task

A macro is supplied with CICS to define the connection between CICS and DB2 by using a resource control table (RCT).

Ensure that you coordinate the attachment facility connection with your CICS support group.

Procedure

To connect the CICS attachment facility:

1. Recalculate space requirements for the CICS attachment facility.
2. Define your CICS attachment facility parameters using the RCT.
3. Update the CICS system tables.
4. Update the CICS initialization JCL.
5. Coordinate DB2 and CICS security if necessary.
6. Prepare new CICS applications for DB2 if necessary.

Related tasks:

Defining the CICS DB2 connection

Stopping DB2 activity

Before running any installation jobs to migrate a data sharing member, ensure that all work is stopped on the subsystem.

Procedure

To stop work on DB2:

1. Issue the following command, where *-DSN1* is the subsystem command prefix that is defined for DB2. You must have system administrator (SYSADM) or system operator (SYSOPR) authority to issue this command.

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI The QUIESCE keyword allows DB2 to complete the processing of programs that are currently executing. This might require some processing time.
2. Issue the following command to allow only the system administrators and system operators that were defined during installation to access DB2:

   GUPI

   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```

   GUPI If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. For example, if the bootstrap data set (BSDS) does not match the subsystem parameter values, ensure that the correct jobs were run for DSNTIJUZ. Ensure that you started DB2 with the correct subsystem parameter option.
3. Ensure that all work is complete.
   - Make sure no units of recovery remain. Issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```

     GUPI Then use -DSN1 RECOVER INDOUBT for any indoubt threads.
   - Ensure that all utilities are completed or terminated. To ensure that no utility work remains, issue the following command:

     GUPI

     ```
     -DSN1 DISPLAY UTILITY(*)
     ```

     GUPI

     After you have determined the utilities that are running, you can let them complete processing or you can terminate the utility. To stop all utilities, issue the following command:

     ```
     -DSN1 TERM UTILITY(*)
     ```
   - Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. To determine existing restrictions, issue the following commands:

     GUPI

     ```
     -DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
     -DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
     ```

     GUPI

     You must have system administrator or system operator authority (specified on installation panel DSNTIPP1) to issue this command.

Recover any table spaces and index spaces that have write error ranges or deferred restart states by using the RECOVER utility.

4. To stop DB2, issue the following command:

   GUPI

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   GUPI

   A user with system administrator (SYSADM) or system operator (SYSOPR) authority must enter this command.

Related reference:

- -DISPLAY DATABASE (DB2) (DB2 Commands)
- -DISPLAY THREAD (DB2) (DB2 Commands)
- -DISPLAY UTILITY (DB2) (DB2 Commands)
- -STOP DB2 (DB2) (DB2 Commands)
- -TERM UTILITY (DB2) (DB2 Commands)

Defining DB2 initialization parameters

Job DSNTIJUZ generates the DB2 data-only subsystem parameter module DSNZP*xxx*, the application defaults load module (*dsnhdecp*), and the data-only load module DSNHMCID. For migrating members, job DSNTIJUZ only contains a subset of the steps that are contained in the DSNTIJUZ job for the first migrated member. The application defaults load module, *dsnhdecp*, is not reassembled for each subsequent migrating member.

About this task

The subsystem parameter module consists of the expansion of six macros that contain the DB2 execution-time parameters that you selected by using the ISPF panels. The names of these macros are DSN6ARVP, DSN6FAC, DSN6GRP, DSN6LOGP, DSN6SPRM, and DSN6SYSP.

Procedure

To define the DB2 initialization parameters:

1. Save your Version 8 or Version 9.1 subsystem parameter module so that it is available in case you need to fall back.
2. Run job DSNTIJUZ.

Results

The DSNTINST CLIST performs calculations by using the values that you specified for some of the parameter values that you entered on the panels. These calculations appear in the macro descriptions.

Establishing subsystem security (optional)

DB2 lets you control access to data within DB2. DB2 also works together with outside security systems, such as RACF, that control access to the DB2 subsystem.

About this task

Because your Version 10 subsystem reuses the data objects from the Version 8 or Version 9.1 subsystem that you are migrating, you have probably already supplied the protection that those objects need. However, you probably want to protect the new (Version 10) DB2 data objects.

Procedure

To establish subsystem security:

1. Run job DSNTIJRA. Job DSNTIJRA defines the administrative task scheduler started task module to RACF program control and defines the administrative task scheduler as a trusted context in RACF. DSNTIJRA is intended for the site security administrator, and can be run any time prior to installing or migrating to Version 10.
2. If you have a DB2 data sharing environment, customize and run DSNTIJRA for each member of the group.

Related concepts:

Getting started with DB2 security (Managing Security)

Defining DB2 Version 10 to z/OS

Job DSNTIJMV completes some of the steps that are required to identify DB2 to z/OS, including updating members of SYS1.PARMLIB and SYS1.PROCLIB.

About this task

For migrating data sharing members, job DSNTIJMV contains a subset of the steps that are contained in the DSNTIJMV job that is used for the first migrating member. The steps to update the IEAAPFxx and LNKLSTxx members, and the steps to update the language procedures are not included in the job for migrating subsequent members.

Procedure

To define DB2 Version 10 to z/OS:

1. Run job DSNTIJMV. If job DSNTIJMV runs successfully, it produces return codes of 0. Because a rename can fail without setting the return code, verify all renames.
2. Review the content of the Language Environment options data set and adjust the following environment variables as needed. The data set is created by DSNTIJMV and must be allocated by the JAVAENV DD of the address space procedure for the DB2 core WLM environment for Java routines.
   - _CEE_ENVFILE, if present, needs to be set to the name and path of the environment variable file that was created by job step DSNTIJR.
   - JVMPROPS needs to be set to the name and path of the JVM startup options file that was created by job step DSNTIJS.
   - DB2_BASE and JCC_HOME need to be set to the names of the HFS directories that you selected for the DB2 for z/OS and JCC product classes.
3. From V9.1 If you are migrating from Version 9.1, review the content of the environment variable file that was created by job step DSNTIJR. Adjust the following information as needed:

- CLASSPATH needs to specify the path for the Java class that contains the JAR files for z/OS XML System Services XML schema registration (gxljapi.jar and gxljosrglmpl.jar).
- LIBPATH needs to specify the path for the native library for z/OS XML System Services (libGXLNVBAT.so).
- STEPLIB needs to specify the name of the DB2 for z/OS runtime library.

From V9.1

What to do next

Because different sites have different requirements for identifying DB2 to z/OS, DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes in SYS1.PARMLIB and SYS1.PROCLIB are incomplete. You might have additional procedures of your own to rename, or you might have to provide procedures for both releases, using alias names to indicate the current release. If you changed the DB2 subsystem name, the names of the DB2 address space startup procedures also change. If you made any changes to your Version 8 or Version 9.1 procedures (such as data set names), make similar changes to the Version 10 procedures. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB or by editing DSNTIJMV.

Recommendation: For SYS1.PROCLIB, submit the procedure-update section of DSNTIJMV as necessary. However, before you make the updates, read the information about DSNTIJMV, and examine DSNTIJMV to study the updates that it makes. Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV job.

Before starting DB2, check the private area sizes in the SYS1.PROCLIB update section to ensure that you have enough user private area.

Also, examine the size of the private area on the DB2 startup procedures. If necessary, modify them to satisfy the requirements for EDM pool size, buffers, numbers of open data sets, and the amount of available private address space.

From V9.1 If you are migrating from Version 9.1, the installation CLIST removes the _CEE_ENVFILE environment variable from the Language Environment options data set. This action allows the data set to maintain compatibility with the Version 9.1 runtime environment for the SYSPROC.XSR_COMPLETE stored procedure. After you migrate to new-function mode, you must add the _CEE_ENVFILE environment variable as part of enabling the Version 10 runtime environment, as explained in “Migrating from enabling-new-function mode (from Version 8 or Version 9.1) to new-function mode” on page 661 From V9.1

Related concepts:

Runtime environment for Java routines (Application Programming for Java)

Related reference:

SETSSI Command

SETPROG Command

IPL z/OS

When you migrate to DB2 Version 10 conversion mode (from Version 8 or Version 9.1), you should complete the z/OS initial program load (IPL) before you start the

DB2 subsystem. You can also IPL z/OS before migration if you have made the appropriate updates to the z/OS libraries.

Before you begin

Alternatively, you can use the SETSSI, SETPROG APF, and SETPROG LNKLST commands to define the subsystem to z/OS until the next IPL.

You should complete the IPL before or during migration, but IPLs are not necessary for fallback or remigration. Before you begin the IPL for z/OS, ensure that job DSNTIJMV was run.

About this task

The load module library SDSNLINK contains the early code. If all of the required maintenance has been applied to your system, the early code is upward compatible with DB2 Version 10. Ensure that the early code pre-conditioning PTFs have been installed on your system before you migrate. The Version 10 early code is downward compatible with Version 8 or Version 9.1. The Version 10 early code is incompatible with Version 7.

Procedure

To IPL z/OS:

If you are at the appropriate service level for Version 8 or Version 9.1, you can plan ahead, do PARMLIB updates (which are necessary at least to update the APF authorization list), and complete the IPL for z/OS whenever convenient, before you begin your migration. Otherwise, IPL z/OS before you start DB2 Version 10.

Results

After you complete the IPL for z/OS, message DSN3100I is displayed on the z/OS console, stating that DB2 is ready for the START command.

Related reference:

SETPROG Command

SETSSI Command

Starting DB2 Version 10

During migration to DB2 Version 10, start the DB2 subsystem after you run job DSNTIJMV and IPL z/OS.

Procedure

To start DB2 Version 10:

1. Start the IRLM. If you have not requested that DB2 automatically start the IRLM, you should start it before you start DB2. Use the following command, where *irlmproc* is the name that you assigned to the IRLM startup procedure:

   ```
   START irlmproc
   ```

   The *irlmproc* is the value that you specified for the PROC NAME option on installation panel DSNTIPI.

   If you specified YES for the AUTO START option on installation panel DSNTIPI, DB2 starts the IRLM automatically.

2. Start DB2 from the z/OS console with the following command, where -DSN1 is the subsystem command prefix that you defined for DB2, and *DSNZPxxx* is the name of the DB2 initialization parameter module:

   ```
   -DSN1 START DB2 PARM(DSNZPxxx)
   ```

   If you omit the PARM parameter, and you specified a value in the PARAMETER MODULE field on installation panel DSNTIPO3, that value is used as the parameter module name. If you did not specify a parameter module on panel DSNTIPO3, DB2 uses the default, DSNZPARM.

   If you have not run job DSNTIJRT, and the administrative task scheduler starts, the administrative task scheduler issues error message DSNA679I to report SQLCODE -805 until you run job DSNTIJRT. The administrative task scheduler starts if you specified an address space procedure name in the ADMIN SCHEDULER field on installation panel DSNTIPX.

   If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, possibly *ssnm*DIST, possibly *ssnm*ADMT, and *irlmproc*, where *ssnm* is the DB2 subsystem name, *ssnm*ADMT is the DB2 administrative task scheduler procedure name, and *irlmproc* is the IRLM procedure name.

   If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

   ```
   DSNR002I   RESTART COMPLETED
   DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
   ```

   If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. A common cause of startup failure is that the BSDS does not match the subsystem parameter values; ensure that the correct job was run for DSNTIJUZ. Also, check that the subsystem parameter member that you specified (or used by default) when you started DB2 is the one that the DSNTIJUZ job built. Check the JCL for the DB2 startup procedure.
3. If you want to use the TSO SUBMIT command to do housekeeping and migration verification, start TSO (if it is not already started).
4. If you did not enable primary user IDs to issue DB2 commands from the z/OS console or TSO SDSF when you first installed the subsystem, take one of the following actions:
   - Grant SYSOPR authority to all primary user IDs that issue DB2 commands from the z/OS console or TSO SDSF. Issue the following command:

     ```
     GRANT SYSOPR TO userid
     ```
   - Define RACF classes to authorize DB2 commands. Use the following statements:

     ```
     SETR CLASSACT(DSNADM)
     RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
     SETR RACLIST(DSNADM) REFRESH
     PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
     SETR RACLIST(DSNADM) REFRESH
     ```

Related reference:

-START DB2 (DB2) (DB2 Commands)

START irlmproc (z/OS IRLM) (DB2 Commands)

Verifying your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system (optional)

Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system by running some of the sample jobs from your previous DB2 release.

About this task

If all of the local DB2 objects from your Version 8 or Version 9.1 subsystem still exist (that is, if you have not run job DSNTEJ0), follow this procedure.

Procedure

To verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system:

1. Change the JOBLIB statements to point to *prefix*.SDSNLOAD.
2. Ensure that the DSN8EAE1 module that you created when you originally ran the Version 8 or Version 9.1 sample jobs is copied to *prefix*.SDSNEXIT. DSN8EAE1 is an EDITPROC that is used by the employee sample table.
3. Edit the Version 8 or Version 9.1 sample jobs before running them. Do **not** run all the sample jobs. Run only the specific jobs and job steps that are listed in the following steps.
4. Test the migration of the IVP phase 2 applications from your previous version:
   a. DSNTEJ2A: Perform all except the first two steps of job DSNTEJ2A. Expect a return code of 4 because table spaces DSN8D910.NEWDEPT and DSN8D910.NEWPHONE are placed in COPY-pending states.
   b. DSNTEJ2C: Issue only the RUN PROGRAM(DSN8BC3) PLAN(DSN8BH81) statement in step PH02CS04.
   c. DSNTEJ2D: Issue only the RUN PROGRAM(DSN8BD3) PLAN(DSN8BD81) statement in step PH02DS03.
   d. DSNTEJ2E: Issue only the RUN PROGRAM(DSN8BE3) PLAN(DSN8BE81) statement in step PH02ES04.
   e. DSNTEJ2F: Issue only the RUN PROGRAM(DSN8BF3) PLAN(DSN8BF81) statement in step PH02FS03.
   f. DSNTEJ2P: Run only step PH02PS05.
5. Test the migration of the IVP phase 3 applications from your previous version:
   a. Do not run job DSNTEJ3C or DSNTEJ3P.
   b. If you want to test the DB2 Version 8 or Version 9.1 ISPF-CAF applications under Version 10, place the Version 8 or Version 9.1 SDSNSPFP panel library ahead of the Version 10 SDSNSPFP panel library in the ISPPLIB concatenation. This placement is necessary so that the plans that are migrated from your previous version can be used. Remove the Version 8 or Version 9.1 SDSNSPFP library from your ISPPLIB concatenation when you are finished testing the Version 8 or Version 9.1 IVP applications under Version 10.

What to do next

Do not run any other Version 8 or Version 9.1 sample jobs.

Related tasks:

"Ensure that sample objects from your previous release are available" on page 578

Migrating a data sharing group to new-function mode

After you migrate all members of a data sharing group to Version 10 conversion mode (from Version 8 or Version 9.1), you can migrate the group to new-function mode. You cannot use new DB2 Version 10 facilities until the group is in new-function mode.

About this task

Unlike migrating to conversion mode (from Version 8 or Version 9.1), migrating to new-function mode is a group-wide process that requires tasks to be completed on a single member.

Procedure

To migrate a data sharing group to new-function mode:

Complete the following tasks on a single member: "Migrating your DB2 subsystem to new-function mode" on page 649.

Mixed releases in a data sharing group

DB2 allows the data sharing group to remain available while you migrate members of the group to the newest release.

However, planning the migration for periods of low activity in the group is recommended because, due to locks that are obtained by the catalog migration utility (CATMAINT), parts of the DB2 catalog are unavailable during the migration of the first member. The parts that are unavailable vary from release to release, depending on what catalog parts require modification.

The purpose of coexistence is to allow applications to continue to access DB2 data while you migrate the members of the data sharing group. However, you must weigh the benefit of continuous availability against the operational costs of running in coexistence mode: new functions are not available, and there are some system management issues to consider.

If you do not require continuous availability, consider shutting down the group during the migration to avoid the coexistence environment. If you need to run in coexistence mode, make a plan to migrate the members in as short a time as possible so that you can minimize the operational complexity.

DB2 allows a maximum of two different release levels to coexist in a data sharing group at any one time.

Special considerations for WLM-managed stored procedures in coexistence

The WLM-managed stored procedure address space (WLM-SPAS) that processes a stored procedure or UDF program always connects to the DB2 member that processes the SQL call request. Therefore, the DB2 code that is executed in the WLM-SPAS must match the DB2 release of the DBM1 address space, or an error occurs.

When running in a data sharing group, each member has a different DBM1 address space. This means that the DB2 code in the WLM-SPAS will have to match more than one DB2 release level. This can cause a problem when you want to use a single JCL definition for the procedure name listed in the WLM definition.

To define one JCL procedure to reference DB2 code data sets at different release levels, use a simple data set and alias naming convention. If the procedure library is shared throughout the SYSPLEX, this allows for a single JCL definition for the Procedure Name that is listed in the WLM definition.

In the following example, the DB2SSN parameter is used as a part of the data set name in STEPLIB. This allows redirection to a data set name based on the SSN of the member that is invoking the stored procedure or UDF.

In the WLM definition:

```
Application Environment Name . : DB2GROUP1
Description  . . . . . . . . . . GROUP 1 APPS
Subsystem Type . . . . . . . . . DB2
Procedure Name . . . . . . . . . DB2AWLM1
Start Parameters . . . . . . . . DB2SSN=&IWMSSNM
                                 ____________________________________

Limit on starting server address spaces for a subsystem instance:
1   1.  No limit
    2.  Single address space per system
    3.  Single address space per Sysplex
```

The special keyword &IWMSSNM in the WLM definition causes WLM to send the subsystem name as the DB2SSN parm when the address space is started. In the WLM-SPAS JCL:

```
//DB2AWLM1 PROC RGN=0M,DB2SSN=,NUMTCB=1,APPLENV=DB2GROUP1
//IEFPROC EXEC PGM=DSNX9WLM,REGION=&RGN,TIME=NOLIMIT,
//        PARM='&DB2SSN,&NUMTCB,&APPLENV'
//STEPLIB  DD  DISP=SHR,DSN=DSNT2.&DB2SSN..SDSNEXIT
//         DD  DISP=SHR,DSN=DSNT2.&DB2SSN..SDSNLOAD
//         DD  DISP=SHR,DSN=DSNT2.&DB2SSN..SDSNLOD2
//         DD  DISP=SHR,DSN=DSNT2.RUNLIB.LOAD
```

If one subsystem name is DT21 then the following alias can be used to redirect the library name to a release-specific library for that member. This would allow you to have a single release-specific library for a data sharing group.

```
DSNT2.DT21.SDSNLOAD                                        *ALIAS
```

As members are migrated, the ALIAS can be changed to reflect the new release that the member is running.

Determining the release of a data sharing group in coexistence

When the first data sharing member starts Version 10, the catalog is considered to be at that level, even though not all members of the group have migrated. This means that the *catalog level* for the group is Version 10.

Procedure

GUPI To determine the catalog level of a group:

Issue the DISPLAY GROUP command.

Example

The following example shows the output of the DISPLAY GROUP command. The catalog level is shown near the top of the display.

```
- -DB1A DISPLAY GROUP
- DSN7107I -DB1A DSN7GCMD
- *** BEGIN DISPLAY OF GROUP(DSNDB0A) CATALOG LEVEL(101) MODE(X   )
-     PROTOCOL LEVEL(1)  GROUP ATTACH NAME(DB0A)
- ------------------------------------------------------------------
- DB2                                    DB2 SYSTEM    IRLM
- MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
- -------- --- ----   --------  -------- --- --------  ----   --------
- DB1A       1 DB1A   -DB1A     ACTIVE   101 MVS1      DJ1A   DB1AIRLM
- DB2A       2 DB2A   -DB2A     ACTIVE   101 MVS2      DJ2A   DB2AIRLM
- DB3A       3 DB3A   -DB3A     ACTIVE   101 MVS3      DJ3A   DB3AIRLM
- DB4A       4 DB4A   -DB4A     FAILED   101 MVS4      DJ4A   DB4AIRLM
- ------------------------------------------------------------------
- SCA   STRUCTURE SIZE:     1024 KB, STATUS= AC,   SCA IN USE:    11 %
- LOCK1 STRUCTURE SIZE:     1536 KB                                 %
- NUMBER  LOCK ENTRIES:      262144
- NUMBER  LIST ENTRIES:        7353, LIST ENTRIES  IN USE:           0
- *** END DISPLAY OF GROUP(DSNDB0A )
- DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 79. DISPLAY GROUP command shows catalog and member release level

GUPI

Determining the function level of an IRLM group in coexistence

IRLM communicates coexistence information by using function levels. A function level is an ever-increasing number that each IRLM can use to tell other IRLMs in the group what level of function it supports.

About this task

The group function level is the **minimum** of the individual IRLM function levels for all IRLMs that can coexist. An IRLM that tries to join a data sharing group is prevented from doing so by active members that cannot coexist with the new IRLM's function level.

When the function level for the group changes, that change is serialized by IRLM with lock structure rebuilds. In most cases, however, the lock structure does not actually do a full rebuild. The first phase of the rebuild is enough to quiesce the work and cause the function level change to occur. These “partial” rebuilds take place when an IRLM joins or leaves the group and if that activity causes the group function level to change. For example, if the IRLM group is currently at function level n, and the IRLM member that wants to join the group is at n-1, the partial rebuild occurs to lower the group function level. Conversely, if the lowest level member leaves the group, the partial rebuild might occur if the group can coexist at a higher function level.

Procedure

To determine IRLM function levels:

Enter the following command:

GUPI

```
MODIFY irlmproc,STATUS,ALLI
```

Examples

Example 1: Assume a data sharing group exists with two members running DB2 Version 10 conversion mode from Version 9.1 and one member running DB2 Version 9.1. Issuing the MODIFY *irlmproc*, STATUS,ALLI command from one of the two Version 10 members produces output that is similar to the following output:

```
DXR103I DJ1A001 STATUS
         IRLMS PARTICIPATING IN DATA SHARING GROUP FUNCTION LEVEL=1.022
         IRLM_NAME IRLMID STATUS LEVEL SERVICE MIN_LEVEL MIN_SERVICE
           DJ1A     001    UP    2.022 HIR2220   2.022     HIR2220
           DJ2A     002    UP    2.022 HIR2220   2.022     HIR2220
           DJ3A     003    UP    1.022 PQ52360   1.012     PN90337
     DXR103I End of display
```

GUPI

The IRLMs in the preceding example are at group function level 1.022, which is the lowest level of any of the individual members (DJ3A). The MIN_LEVEL field shows the minimum level with which this IRLM can coexist. MIN_SERVICE indicates the service or release that corresponds with that MIN_LEVEL.

GUPI

Example 2: Assume the same data sharing group as described in the previous example. Issuing the MODIFY *irlmproc*, STATUS,ALLI command from the Version 9.1 member produces output that is similar to the following output:

```
DXR103I DJ3A003 STATUS
         IRLMS PARTICIPATING IN DATA SHARING GROUP FUNCTION LEVEL=022
         IRLM_NAME IRLMID STATUS LEVEL SERVICE MIN_LEVEL MIN_SERVICE
           DJ1A     001    UP      094 HIR2220     022     HIR2220
           DJ2A     002    UP      094 HIR2220     022     HIR2220
           DJ3A     003    UP      022 PQ52360     012     PN90337
     DXR103I End of display
```

GUPI

Call attachment and TSO attachment in coexistence

While operating in a coexistence environment, you can attach to either release of DB2 with your existing TSO logon procedures or with JCL.

After you migrate all members of the group to DB2 Version 10, update those procedures and jobs to point to the Version 10 load libraries.

Automatic rebinds in coexistence

New functions that are introduced in Version 10 are not available to members of the group that have not migrated, so you should plan accordingly when developing your migration plan.

It is best to prevent packages and plans that are bound on Version 10 from executing on members that have not yet been migrated. Or, do not bind plans and packages on Version 10.

This prevention serves two purposes. First, if plans or packages that are bound on Version 10 use new functions, you can avoid the application errors that would occur if the plan or package tried to execute an SQL statement that is not allowed

in Version 8 or Version 9. Second, you can prevent the automatic rebind that occurs when any plan or package that is bound on Version 10 is run on Version 8 or Version 9. It also prevents the automatic rebind that occurs when a Version 10-bound plan or package that was automatically rebound on Version 8 or Version 9 is later run on Version 10. (Such a bind is called a *remigration rebind*.)

If you are enforcing where plan or package runs are not possible, consider how you want to handle binds and automatic rebinds while two releases are coexisting.

Related tasks:

"Disallowing the automatic remigration rebind" on page 728

Disallowing all automatic rebinds

One approach to handle binds and automatic rebinds while two releases are coexisting is to disallow all binds and disable all automatic rebinds on Version 10 members.

Procedure

To disallow or avoid automatic rebinds on a Version 10 member:

- Specify NO for the ABIND subsystem parameter. This disables all automatic rebinds on the Version 10 member for any reason. Therefore, you cannot run a plan or package on a Version 10 member if it has gone through the following scenario:
  1. Binds on a Version 10 member.
  2. Runs on a Version 8 or a Version 9.1 member. This action causes an automatic rebind on the Version 8 or Version 9.1 member.
  3. Attempts to run on a Version 10 member.

     (This action returns a -908 SQLCODE (SQLSTATE '23510') because DB2 must automatically rebind the plan or package on Version 10 before running it on the Version 10 member.)
- GUPI Use the resource limit facility to disallow BIND operations. Do this by inserting rows in the resource limit specification table (RLST) to set RLFFUNC to "1" and RLFBIND to "N". This ensures that nobody binds plans or packages on Version 10.

  Example: The following is an INSERT statement for the RLST that disallows all BIND operations for all authorization IDs (except those with installation SYSADM or installation SYSOPR authority) for all packages and plans:

  ```
  INSERT INTO authid.DSNRLSTxx
    (RLFFUNC,RLFBIND) VALUES('1','N');
  ```

  GUPI

What to do next

After all the members of a data sharing group have migrated to the current release, enable automatic rebinds by setting the ABIND subsystem parameter to YES. Allow bind operations by changing the RLST accordingly or by stopping the resource limit facility using the STOP RLIMIT command.

Related reference:

"AUTO BIND field (ABIND subsystem parameter)" on page 354

Disallowing the automatic remigration rebind

One approach to handle binds and automatic rebinds while two releases are coexisting is to disable only automatic remigration rebinds. By disabling automatic remigration rebinds, you avoid the thrashing that can occur when a plan or package is rebound each time it runs on a member of a different version.

Procedure

To avoid the automatic remigration rebind situation:

Specify COEXIST for the ABIND subsystem parameter on the Version 10 members. Automatic rebind then occurs on Version 10 in the following circumstances:

- The package is marked invalid.
- A plan or package that was previously bound on DB2 Version 10 runs on DB2 Version 8 or Version 9.1.

When all members are at Version 10, packages that are remigrated from DB2 Version 8 or Version 9.1 are not automatically rebound.

Related concepts:

"Automatic rebinds in coexistence" on page 726

Related reference:

"AUTO BIND field (ABIND subsystem parameter)" on page 354

Recommendations for BIND in coexistence

If DSN is at Version 10 and the member that is named in the DSN command is at Version 8 or Version 9.1, using certain bind options causes a BIND or REBIND subcommand to be rejected.

If you are migrating from Version 8 or Version 9.1, the following bind option causes rejection:

- ENCODING for BIND and REBIND PLAN or PACKAGE

To avoid problems, make sure that the member that is named in the DSN subcommand matches the load libraries that are used for the DSN command.

Recommendations for utilities in coexistence

Until all members of the data sharing group are running Version 10, avoid using any of the new utility functions that are available in Version 10. However, as long as you use utility options that are supported in Version 8 or Version 9, utilities can attach to either a Version 8 or Version 9 member.

The utilities batch program (DSNUTILB) is split into multiple load modules: a release-independent load module called DSNUTILB, multiple release-dependent module DSNUT810 (or DSNUT910, if you are migrating from Version 9) or DSNUT101, and multiple utility-dependent load modules. To operate in a mixed-release data sharing environment, you must have DSNUT810 or DSNUT910 (if applicable), and DSNUT101. And you must have all utility-dependent load modules and their aliases for the utilities that you have purchased available to the utility jobs that operate across the data sharing group. The instructions on making these load modules available are below.

Changing STEPLIB in DSNUPROC: The recommended way of making the release-dependent modules available for utility jobs is to change the STEPLIB in DSNUPROC to include the other release (DSN=DSN810.SDSNLOAD for Version 8, or DSN=DSN910.SDSNLOAD for Version 9), as in the following example:

```
//DSNUPROC PROC LIB='DSN1010.SDSNLOAD',
//         SYSTEM=DSN,
//         SIZE=0K,UID=',UTPROC='
//DSNUPROC EXEC PGM=DSNUTILB,REGION=&SIZE,
//         PARM='&SYSTEM,&UID,&UTPROC'
//STEPLIB  DD   DSN=&LIB,DISP=SHR;
//         DD   DSN=DSN810.SDSNLOAD,DISP=SHR <— coexistence
//SYSPRINT DD   SYSOUT=*
b//UTPRINT  DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//*DSNUPROC PEND        REMOVE * FOR USE AS INSTREAM PROCEDURE
```

Cross-copy into load libraries: Another approach, which is not recommended for long-term use, is to cross-copy the release-dependent modules into the load libraries of the other release. For example, copy DSNUT910 into the Version 10 load libraries, and copy DSNUT101 and all applicable utility-dependent load modules into the Version 9 load libraries. Or, for example, copy DSNUT810 into the Version 10 load libraries, and copy DSNUT101 and all applicable utility-dependent load modules into the Version 8 load libraries. The problem with this approach is that you must repeat this procedure every time you apply maintenance to these modules. Thus, as with coexistence in general, this approach is only for short-term use.

The following example shows sample JCL to perform a cross-copy from Version 10 to Version 9 and from Version 9 to Version 10:

```
//CROSCOPY PROC V10PREF='DSN1010',
//            V9PREF='DSN910',
//            RGN=4096K,SOUT='*'
//* ******************************************************************
//*    FOR EXECUTION OF IEBCOPY - DB2 POST-INSTALLATION            ***
//* ******************************************************************
//COPY     EXEC PGM=IEBCOPY,REGION=&RGN
//SYSUT3   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT4   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//* **************  DB2 TARGET LIBRARIES  ****************************
//*
//V9LOAD   DD  DSN=&V9PREF..SDSNLOAD,DISP=OLD
//V10LOAD  DD  DSN=&V10PREF..SDSNLOAD,DISP=OLD
//*
//         PEND
//COEXIST  EXEC PROC=CROSCOPY
//SYSIN    DD *
  COPYMOD INDD=((V10LOAD,R)),OUTDD=V9LOAD
  SELECT MEMBER=(DSNUT101)     <-- add any utility-dependent modules here
  COPYMOD INDD=((V9LOAD,R)),OUTDD=V10LOAD
  SELECT MEMBER=(DSNUT910)
```

The following example shows sample JCL to perform a cross-copy from Version 10 to Version 8 and from Version 8 to Version 10:

```
//CROSCOPY PROC V10PREF='DSN1010',
//            V8PREF='DSN810',
//            RGN=4096K,SOUT='*'
//* ******************************************************************
//*    FOR EXECUTION OF IEBCOPY - DB2 POST-INSTALLATION            ***
//* ******************************************************************
//COPY     EXEC PGM=IEBCOPY,REGION=&RGN
//SYSUT3   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT4   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
```

```
//* **************  DB2 TARGET LIBRARIES  ****************************
//*
//V8LOAD   DD  DSN=&V8PREF..SDSNLOAD,DISP=OLD
//V10LOAD  DD  DSN=&V10PREF..SDSNLOAD,DISP=OLD
//*
//         PEND
//COEXIST  EXEC PROC=CROSCOPY
//SYSIN    DD *
  COPYMOD INDD=((V10LOAD,R)),OUTDD=V8LOAD
  SELECT MEMBER=(DSNUT101)     <-- add any utility-dependent modules here
  COPYMOD INDD=((V8LOAD,R)),OUTDD=V10LOAD
  SELECT MEMBER=(DSNUT810)
```

Related concepts:

Operation of DB2 utilities in a mixed-release data sharing environment (DB2 Utilities)

Related reference:

"Load module names for running purchased utilities in coexistence"

Load module names for running purchased utilities in coexistence

Each utility that you purchase has a separate load module that is associated with one of 20 load modules.

When a utility is executed, it is loaded by using its load module, which is listed in the following table. By including in the link list the load module name for those utilities that you want, you can run in coexistence mode without specifying a STEPLIB in the JCL.

Table 100. Version 10 utility features

Feature	Load Module Names
CATMAINT	DSNU10LA
CHECK	DSNU10LB
COPY	DSNU10LC
DIAGNOSE [1]	DSNU10LD
LISTDEF	DSNU10LE
LOAD	DSNU10LF
MERGECOPY	DSNU10LG
MODIFY	DSNU10LH
OPTIONS	DSNU10LI
QUIESCE	DSNU10LJ
REBUILD	DSNU10LK
RECOVER	DSNU91LL
REORG	DSNU10LM
REPAIR	DSNU10LN
REPORT	DSNU10LO
RUNSTATS	DSNU10LP
STOSPACE	DSNU10LQ
TEMPLATE	DSNU10LR
UNLOAD	DSNU10LS

Table 100. Version 10 utility features (continued)

Feature	Load Module Names
COPYTOCOPY	DSNU10LT
EXEC SQL	DSNU10LU
BACKUP SYSTEM and RESTORE SYSTEM	DSNU10LV
[1]Use this utility only under the direction of IBM Software Support.	

Table 101. Core utility load modules that are shipped with the base

DSNU10LA	DSNU10LD	DSNU10LE	DSNU10LI
DSNU10LJ	DSNU10LN	DSNU10LO	DSNU10LR

Recommendation for group restart in coexistence

If a group restart is necessary while the data sharing group is running in a coexistence environment, issue the START command only for Version 10 members.

Do not start the Version 8 or Version 9.1 members until the Version 10 members have completed forward log recovery. If a Version 8 or Version 9.1 member performs the group restart for a Version 10 member, the previous version adds pages to the logical page list during the peer-forward recovery phase when it tries to apply redo log records against a release-dependent object.

Recommendation for SPUFI in coexistence

When you migrate the first member of the data sharing group to Version 10, you run job DSNTIJSG, which rebinds SPUFI in Version 10.

Binding SPUFI in Version 10 causes SPUFI to be unavailable to the Version 8 or Version 9.1 members. If you attempt to run an SQL statement on a member that has yet to migrate to Version 10, expect messages that indicate an unavailable resource.

Testing the data sharing group

When you installed DB2, sample objects were created in job DSNTEJ1. The DSNTESD member of *prefix*.SDSNSAMP contains SQL statements that refer to these objects. Use these SQL statements to test group buffer pool caching, global lock serialization, and concurrency in the data sharing group.

Perform these tests after installing several data sharing members.

Testing group buffer pool caching

You can use SQL statements in member DSNTESD to verify that the group buffer pool caching operates correctly.

Procedure

To verify that the group buffer pool operates correctly:

1. Run SPUFI on more than one data sharing member, using member DSNTESD as the input data set. Specify AUTOCOMMIT=YES on the SPUFI panel.

   Run SPUFI on the different members serially, a few seconds apart if possible. (The runs must be close enough together to avoid having DB2 close the page

set because of infrequent updates. The default amount of time between updates before DB2 switches the page set from read-write to read-only is 10 minutes, as specified by the RO SWITCH TIME field of installation panel DSNTIPL1. RO SWITCH CHKPTS can also cause the data set to switch to read-only. It is a number of consecutive checkpoints.

By running the SQL statements serially, DB2 detects inter-DB2 read/write interest on the table space and index and uses the group buffer pool.

2. Verify that ITEM_COUNT increases by five after each run.
3. Issue the following command to determine whether the table space and index are using the group buffer pool:

   GUPI

   ```
   DIS DB(DSN8D10A) SPACENAM(DSN8S10S,XPARTS) LOCKS
   ```

   GUPI

   If the P-lock state is IX or SIX, then the table space and index are group buffer pool dependent, which is the correct state.
4. Issue the following command to display the statistics for GBP0:

   GUPI

   ```
   DIS GBPOOL(GBP0) GDETAIL
   ```

   GUPI

   In the group detail statistics, look for non-zero values in the READS and WRITES values of the display. This indicates that DB2 is using the group buffer pool successfully for caching.

Testing global lock serialization

Use the SQL statements in member DSNTESD to verify that locks are acquired and released correctly across multiple data sharing members.

Procedure

To test global lock serialization:

1. Run SPUFI, using member DSNTESD as input, on **member 1**. Specify AUTOCOMMIT=NO.

   Because you have inserted data into DSN81010.PARTS but have not committed, **member 1** holds global locks.
2. Run SPUFI, using member DSNTESD as input, on **member 2**. Specify AUTOCOMMIT=NO.

   Because **member 1** holds global locks, **member 2** must wait to perform the insert.
3. In less than one minute, commit on **member 1**. (If you wait too long to commit, **member 2** will experience a lock timeout.)

   The global locks should be released, and **member 2** should be able to proceed. Verify that ITEM_COUNT has increased by five.

Testing concurrency

You can use the SQL statements in member DSNTESD to test concurrency within the data sharing group.

Procedure

To test concurrency:

1. Run SPUFI concurrently on different data sharing members. Specify AUTOCOMMIT=YES.

   Global locking ensures that inserts to DSN81010.PARTS are coordinated across data sharing members.
2. Verify that ITEM_COUNT increases by five each time the run completes successfully.

Testing Sysplex query parallelism

No sample procedure to test Sysplex query parallelism is provided in member DSNTESD. However, you can use your own data to confirm that a single query can be processed on more than one member of the data sharing group.

About this task

Choose an existing query that you know uses CP parallelism, such as a SELECT COUNT(*) for a table in a large partitioned table space, and use this procedure to force DB2 to split the query across multiple members of the data sharing group.

Procedure

To test Sysplex query parallelism:

1. Decide which member will be the query coordinator for your test, and make sure that the COORDINATOR field on installation panel DSNTIPK is set to YES for that member.
2. Make sure that all potential assistants have the ASSISTANT field on installation panel DSNTIPK set to YES.
3. Make sure that the query does not include any query restrictions for parallel processing.
4. Run EXPLAIN on the query.

   An X in the PARALLELISM_MODE column of the PLAN_TABLE output indicates that this query can be split across multiple members.
5. Set buffer pool allocation thresholds on the members that you want considered as possible assistants:

   GUPI

   ```
   ALTER BUFFERPOOL (BPn) VPSIZE(z) VPSEQT(100) VPPSEQT(100) VPXPSEQT(100)
   ```

   GUPI

   Ensure that the VPSIZE is large enough to support parallel processing. Start with at least 50 buffers on each query assistant.
6. Make sure that accounting trace class 3 is active on the parallelism coordinator.
7. Run the query.
8. PSPI Inspect the IFCID 0003 trace record. Field QWA01RBN corresponds to the number of assisting members. This value should be greater than zero.

   PSPI

Related tasks:

Programming for parallel processing (DB2 Performance)

Related reference:

"Define group or member panel: DSNTIPK" on page 286

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

Falling back and remigrating in a data sharing environment

In a data sharing environment, you can fall back from Version 10 or remigrate one member at a time or all members at the same time.

Related concepts:

"Mixed releases in a data sharing group" on page 723

Falling back in a data sharing environment

If you need to fall back to Version 8 or Version 9.1, you can fall back one member of the data sharing group at a time. Other members can continue to run while one member is falling back.

Before you begin

Before falling back to Version 8 or Version 9.1, you must have the fallback and coexistence SPE and its prerequisite APARs installed on the Version 8 or Version 9.1 load library.

If all members have already been migrated to the new release, and you fall back one member at a time, the group is running in coexistence mode after the first member falls back.

Procedure

To fall back to Version 8 or Version 9.1:

1. Stop DB2 on the member that is falling back or stop DB2 on all members if you are falling back all members at once.
2. Run job DSNTIJFV to reactivate Version 8 or Version 9.1 for that member or for all members.
3. Reconnect TSO, IMS, CICS to Version 8 or Version 9.1.
4. Start Version 8 or Version 9.1:
   a. Enter the command START DB2:
      1) Check for indoubt units of recovery.
      2) Check for outstanding restrictive states.
5. Modify your DSNTIJIC job to account for new table spaces that are added to the DB2 catalog in conversion mode (from Version 8 or Version 9.1). This step prepares your DSNTIJIC job to be used in fallback mode.
   - From V8 If you migrated to Version 10 from Version 8, add the following table spaces to the Version 8 DSNTIJIC job:
     - DSNDB06.SYSCONTX
     - DSNDB06.SYSPLUXA
     - DSNDB06.SYSROLES
     - DSNDB06.SYSRTSTS
     - DSNDB06.SYSTARG

- DSNDB06.SYSTSASC
- DSNDB06.SYSTSUNI
- DSNDB06.SYSXML

From V8

- From V9.1 If you migrated to Version 10 from Version 9.1, add the following table spaces to the Version 9.1 DSNTIJIC job:
  - DSNDB06.SYSTSASC
  - DSNDB06.SYSTSUNI

From V9.1

6. Verify Version 8 or Version 9.1.
7. Repeat steps 1 through 6 for each member (or all members) of the data sharing group.

Related concepts:

"Mixed releases in a data sharing group" on page 723

Related tasks:

"Creating an automatic restart policy" on page 168

Remigrating in a data sharing environment

You can remigrate one member at a time or all members of a data sharing group.

Before you begin

Before you remigrate, refer to "Changes that might affect your migration from Version 8" on page 65 or "Changes that might affect your migration from Version 9.1" on page 114 because many of those considerations apply to remigrations, too. Whether each consideration applies to your subsystem depends on the type of activity that occurred on your Version 8 or Version 9.1 subsystem after fallback.

About this task

A plan or package is automatically rebound in Version 10 when it is executed for the first time after remigration if it was not explicitly bound in Version 8 or Version 9.1. However, if you specified NO for the ABIND subsystem parameter, automatic binds are disabled. This means that the plan or package from your previous release is the one that runs in Version 10, so the plan or package does not benefit from Version 10 enhancements.

Procedure

To remigrate your subsystem to Version 10:

1. Check the integrity of your Version 8 or Version 9.1 catalog and directory:
   - Run the DSN1COPY utility with the CHECK option on your catalog table spaces.
   - Run the link checker utility, DSN1CHKR, to ensure that no existing links are broken.
   - Run the queries in member DSNTESQ of *prefix*.SDSNSAMP.

2. Optional: Ensure that your Version 8 or Version 9.1 DSNTIJIC job has been modified to copy new Version 10 catalog table spaces, and then use DSNTIJIC to take an image copy of your Version 8 or Version 9.1 catalog. This step is not required, but it is recommended.
3. Stop your Version 8 or Version 9.1 subsystem.
4. Reconnect TSO, IMS, and CICS to DB2 Version 10. Re-establish your Version 10 logon procedures and JCL, as well as your Version 10 CICS and IMS connections.
5. Rebuild Version 10 cataloged procedures. Rename the Version 10 procedures that were renamed by job DSNTIJFV during fallback. If job DSNTIJFV was not run, you need to rerun job DSNTIJMV. Comment out step 1 (DSNTIMP), which defines Version 10 to z/OS, and run the job. (You do not need to define Version 10 to z/OS a second time.)
6. Start DB2 Version 10. Ensure that you are using your Version 10 subsystem parameter load module.
7. Take an image copy of the Version 10 catalog by using the DSNTIJIC job.
8. Verify your DB2 Version 10 system.
9. From V9.1 If you used the SYSPROC.XSR_COMPLETE stored procedure in Version 9.1, ensure that the WLM environment that is used for running XSR_COMPLETE is updated to use the Version 10 code paths. The following parameters in the Java Language Environment (JAVAENV) data set for the WLM environment are affected:
   - DB2_BASE
   - JCC_HOME
   - JAVA_HOME

   From V9.1

Related concepts:

"Mixed releases in a data sharing group" on page 723

Related tasks:

"Falling back in a data sharing environment" on page 734

"Re-enabling DB2 data sharing" on page 740

Disabling and re-enabling DB2 data sharing

Disabling DB2 data sharing is a complex procedure and is very disruptive. Do not attempt to disable DB2 data sharing without a thorough understanding of the process.

Do not make disabling DB2 data sharing part of your contingency plans for handling recovery situations. For temporary bypasses to data sharing problems, try moving to one-way data sharing, which involves stopping all but one member and having that member perform the work for the group. If one-way data sharing does not work, your data sharing problem might be resolved by performing a group restart.

The disabling procedure is included in this section for completeness, but it should rarely be used. Situations in which it might be necessary to disable DB2 data sharing are:

- You made a strategic decision to move away from a data sharing environment.
- One-way data sharing is not working.

After you have disabled DB2 data sharing, only **one** member of the data sharing group can access the previously shared data. That member is called the *surviving member*.

If you are planning to re-enable data sharing for a group, do not change any group-wide information in the surviving member's BSDS. Group-wide information includes the catalog alias name and the database password. It also includes the DDF name and password information, even if you are not planning to use DDF when you re-enable DB2 data sharing. If you change any of this information, you will have to change the information in every member's BSDS before you start the group.

Disabling data sharing

You should only disable data sharing with caution and with a thorough understanding of the process and results. Very few situations create a need to disable DB2 data sharing, such as if you have data sharing problems and one-way data sharing does not work.

About this task

The procedure to disable data sharing ensures that the most recent versions of all pages are externalized from the group buffer pool to disk. DB2 does not use the group buffer pool after data sharing is disabled. You must ensure that data is written to disk, or else you lose data when you start DB2 after disabling data sharing.

You must also ensure that there is no need to recover data from information that is contained in other members' logs after you disable data sharing. **Other members' logs are not available to the surviving member after you disable data sharing.** To prevent the surviving member from applying inconsistent updates during recovery processing, a cold start is required to disable data sharing.

Important: Do not attempt to go through the installation process to re-enable data sharing after you have disabled it.

Procedure

To disable data sharing and return to a non-data-sharing environment:

1. Decide which member will be the surviving member of the group. The surviving member is the only member that can access previously shared data.
2. Stop all members by entering the following command for each member of the data sharing group:

   GUPI

   ```
   STOP DB2 MODE(QUIESCE)
   ```

   GUPI

3. Start the surviving member in maintenance mode by using the following command:

   GUPI

   ```
   START DB2 ACCESS(MAINT)
   ```

GUPI

4. Make sure that data is consistent. Enter the following commands from the surviving member of the group. Do not proceed to the next step until all problems are resolved.
   - DISPLAY GROUP

     Ensure that the status of the surviving member is ACTIVE and that the status of all other members is QUIESCED. If the status of a non-surviving member is not QUIESCED, take the following actions, depending on the member's status:
     - If the member's status is FAILED, restart and then stop the member.
     - If the member has castout problems, indoubt units of recovery, or outstanding resynchronization problems, start that member in maintenance mode and fix the problem.
   - DISPLAY UTILITY(*)

     If any utility work remains for any member of the group, restart that member with ACCESS(MAINT) and either stop the utility or let it finish.
   - DISPLAY DATABASE(*) SPACENAM(*) RESTRICT

     If any restricted table spaces or index spaces (such as write error ranges, recovery pending status, or logical page list entries) exist, recover them from the surviving member.
5. Stop the surviving member by using the following command:

   GUPI

   ```
   STOP DB2 MODE(QUIESCE)
   ```

   GUPI
6. Stop any IRLMs that have not stopped by using the following command:

   GUPI

   ```
   STOP irlmproc
   ```

   GUPI
7. Dismantle the data sharing group.
   a. Enter the following command to display the structures for the data sharing group:

      ```
      D XCF,STRUCTURE,STRNAME=grpname*
      ```
   b. For all structures that are still allocated (STATUS:ALLOCATED) and that still have connections (which appear as FAILED PERSISTENT), enter the following command to force the connections off of those structures:

      ```
      SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL
      ```

      Important: If your site is running z/OS with APAR OA02620 applied, you do not need to force failed-persistent connections off of the lock structure. When you forcibly deallocate the lock structure (see the next step), the system deletes failed-persistent connections to the structure for you.
   c. Delete all the DB2 coupling facility structures by using the following command for each structure:

      ```
      SETXCF FORCE,STRUCTURE,STRNAME=strname
      ```
   d. Edit the JCL in job DSNTIJGF to point to the correct BSDS data sets.

DSNTIJGF is a change log inventory job that sets up the surviving member for a cold start.

Important: Do not change the hex values that appear in the change log inventory CRESTART control statement. They are not real RBA values.

e. Run job DSNTIJGF.

After you run this job, do not try to restart any of the non-surviving members. None of those members can start successfully.

8. Change the IRLM procedure to SCOPE=LOCAL.
9. Start the surviving member with ACCESS(MAINT). Specify the old DSNZP*xxx* from the non-data sharing environment. If the surviving member is not the originating member, you must reassemble the surviving member's subsystem parameters, specifying the subsystem parameter DSHARE=NO in the invocation of the DSN6GRP macro. Also, comment out all steps from the DSNTIJUZ job except for those that reassemble and link-edit the subsystem parameters.

   When you start DB2 after running DSNTIJGF, respond with Y to a cold start prompt (message DSNJ246I on the z/OS console).

   This is a cold start because DB2 increases the log RBA to a value higher than any LRSN used while sharing data. From this point on, your RBAs look like LRSNs.
10. Edit and run job DSNTIJFT, if necessary, to ensure that the surviving member's work file database is DSNDB07.

    The surviving member must use DSNDB07 as its work file database. If the work file database for the surviving member is not DSNDB07, drop that work file database and run job DSNTIJFT.
11. Verify that the surviving member works by running a subset of the Version 10 installation verification sample jobs.
12. To establish a new recovery point, take a full or incremental image copy or non-DB2 backup of all data. Run job DSNTIJIC to image copy the DB2 catalog and directory.

    Recommendation: Perform this step as soon as possible after data sharing is disabled.
13. Stop and restart DB2 for normal unrestricted access.

What to do next

After disabling DB2 data sharing, you can continue to use the group attachment name or subgroup attachment name. The group attachment name or subgroup attachment name does not need to be changed to the surviving member's subsystem ID.

Related concepts:

"Data recovery after disabling DB2 data sharing"

Related tasks:

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Data recovery after disabling DB2 data sharing

After DB2 data sharing is disabled, you cannot recover to the current point or to a previous point in time if that recovery depends on any portion of the log that was made before you disabled data sharing.

Therefore, if the table space is updated between the time of the copy and the time you disabled data sharing, DB2 does not let you use that copy as the basis for recovery. This is why you should create a new recovery point as soon as possible after you have disabled data sharing.

Re-enabling DB2 data sharing

Re-enabling DB2 data sharing involves a subset of the original tasks that were required to enable data sharing. You cannot use the enabling process of installation to re-enable DB2 data sharing.

About this task

The following output from the original DB2 data sharing enabling procedure remains intact after disabling DB2 data sharing and does not need to be re-created or re-specified:

- Data sharing subsystem parameters (output from the CLIST processing when enabling data sharing)
- XCF definitions
- Coupling facility definitions
- RACF definitions
- DB2 catalog and directory

Procedure

To re-enable DB2 data sharing:

1. Edit the JCL in job DSNTIJGF.

   DSNTIJGF is a change log inventory job that sets up the BSDS of the surviving member for data sharing.
2. Run job DSNTIJGF.
3. Change the IRLM procedure to SCOPE=GLOBAL.
4. Start the surviving member with the subsystem parameters that were used when data sharing was originally enabled.

   During startup, you are asked to start all other members that were not quiesced at the time that you disabled DB2 data sharing. You must start all these members.
5. Edit and run job DSNTIJFT on the surviving member to recreate the work file database for data sharing.

   See the directions in the prologue of job DSNTIJFT for information about editing this job to re-enable DB2 data sharing.

Renaming a member by performing an IPL

If you want to rename a member of the data sharing group, you should do so either before enabling DB2 data sharing or during the process of enabling DB2 data sharing. Because DB2 must be shut down during the enabling process, you can perform the tasks that are necessary to rename the member at the same time.

About this task

Renaming a member is an activity that you should plan for very carefully. Because every installation has a different configuration, this procedure is not guaranteed to be complete.

In the following renaming example procedure, the member name for DB2P is changed to DB1A to conform to the naming convention for data sharing that is used in this publication. If you choose to do any of the following tasks, you must IPL z/OS to pick up the changes.

Procedure

To rename a data sharing member:

1. Modify the IEFSSN*xx* member for this subsystem to include the group attachment name. To avoid having to modify JCL for all your jobs, use the existing subsystem name as the group attachment name. This step is necessary to ensure that there are no problems when the call attachment and TSO attachment facilities try to use the group attachment name.

   Example: If the existing IEFSSN*xx* member looks like:

   ```
   DB2P,DSN3INI,'DSN3EPX,?'
   ```

   Change it to look like:

   ```
   DB2P,DSN3INI,'DSN3EPX,?,S,DB2P'
   ```

2. Add new IEFSSN*xx* definitions for the new name and prefix:

   ```
   DB1A,DSN3INI,'DSN3EPX,-DB1A,S,DB2P'
   DJ1A
   ```

   Optionally, you can use the z/OS command SETSSI to add the new IEFSSN*xx* statements without an IPL. You can add the new statements when you add the definitions for the new names or later during the enabling process. Do not forget to add these names to the IEFSSN*xx* member before you IPL again.

   The command prefix -DB1A is not a subset or a superset of DB2P's command prefix. For example, ?DB1A is invalid in this context.

3. Make sure that RACF definitions are in place to handle the new subsystem names. Add the correct names to the following tables:
   - The RACF router table
   - The started procedures table (ICHRIN03), if used

Related tasks:

Changing the high-level qualifier for DB2 data sets (DB2 Administration Guide)

Renaming a member by stopping DB2

If you want to rename a member of the data sharing group, you should do so either before or while enabling DB2 data sharing. Because DB2 must be shut down during the enabling process, you can perform the tasks that are necessary to rename the member at the same time.

Before you begin

Renaming a member is an activity that you should plan for very carefully. Because every installation has a different configuration, this procedure is not guaranteed to be complete.

About this task

You can do the following tasks when you bring down the DB2 subsystem for the enabling procedure.

Procedure

To rename a member by stopping DB2:

1. Define the correct profile names for the DSNR class.
2. Replicate existing PERMIT commands to allow users and groups to access the new profiles.
3. Complete the installation panels for enabling data sharing. (You cannot change the subsystem name on the installation panels.)

   On installation panel DSNTIPH, modify the archive prefix names to include the member name, like DB2PCAT.DB1A.ARCLG1. Specify the old subsystem name (for example, DB2P) as the group attachment name.
4. Stop DB2 with MODE(QUIESCE).
5. Run job DSNTIJUZ to assemble and link-edit the new subsystem parameter data set (DSNZP*xxx*) and DSNHDECP.
6. Rename the BSDS and active log data sets with the new prefix, like DB2PCAT.DB1A.BSDS01.
7. Update the BSDS with the renamed log data sets. **Include the same RBA ranges as the original active log data sets:**
   a. Run the utility print log map (DSNJU004) to obtain the start and end RBAs.
   b. Use access method services to rename the log data set.
   c. Run the utility change log inventory (DSNJU003) to delete the active logs with the old names.
   d. Run the utility change log inventory (DSNJU003) to add the renamed active logs with the correct ranges. You do not need to add the archive log data sets, because the archive log data sets that you specified on panel DSNTIPH will replace them.
8. If necessary, increase the size of the BSDS.
9. If necessary, increase the size of the SYSLGRNX table space.
10. Rename the startup procedures. For example, change DB2PMSTR to DB1AMSTR. Do not forget to change the BSDS names to the new names in the *ssnm*MSTR startup procedure.
11. Make sure that CICS can connect to the new subsystem. There are several ways of doing this, depending on which level of CICS you are running. Here are a couple of techniques:
    - Change the CICS DB2CONN to use the new DB2 subsystem name, reassemble, and link-edit.
    - Modify the JCL for CICS to include the new subsystem name on the INITPARM.
12. Make sure that IMS can connect to the new subsystem.
13. Enter the command START DB2 and continue with the rest of the enabling process.

Related tasks:

"Enabling DB2 data sharing" on page 679

"Increasing the size of the BSDS" on page 196

"Increasing the size of the SYSLGRNX table space" on page 196

Changing the high-level qualifier for DB2 data sets (DB2 Administration Guide)

Chapter 7. Verifying that installation or migration was successful with the sample applications

You can use the DB2 Version 10 sample applications to verify either installation or migration to new-function mode. These sample applications are also called installation verification procedures (IVPs).

About this task

Attention: Do not run the Version 10 IVP jobs unless your subsystems are in Version 10 new-function mode.

Recommendations:

- During verification, run all sample applications under the same user ID. This user ID must have SYSADM authority. Otherwise, errors might occur.
- Run the IVP jobs with DB2 started in unrestricted mode because restricted mode (ACCESS(MAINT)) does not allow use of stored procedures and user-defined functions.
- If you are installing a data sharing group, run the IVP jobs after you install or migrate the originating system. You do not need to run the IVP jobs after you enable the originating system or after you install a new data sharing member.
- You must use the same compiler for each job. For example, if you use IBMCOB for DSNTEJ2C, you must use IBMCOB for all COBOL verification programs.
- After you designate a client and server subsystem, run all verification jobs on the same client and server subsystems.

Procedure

To verify a successful installation or migration:

1. For testing purposes, identify one local subsystem to act as the client and one remote subsystem to act as the server. Both subsystems must be at Version 10. Although not recommended, the client and server subsystem can be the same if a remote subsystem is not available for testing.
2. Ensure that the client subsystem is specified in the DB2 LOCATION NAME field of installation panel DSNTIPR.
3. Ensure that the server subsystem is specified in the REMOTE LOCATION field of installation panel DSNTIPY.
4. Identify the IVP jobs that you need to run based on the attachment facilities you installed, the languages you use, and whether the sample objects exist.
5. Edit the sample jobs as necessary based on the information that is provided for the jobs.
6. Run the necessary IVP jobs.

What to do next

When you complete the verification procedure, save the verification objects. You will need them when you migrate to the next release of DB2.

© Copyright IBM Corp. 1982, 2012

Related tasks:

"Enabling DB2 data sharing" on page 679

"Migration step 30: Verify your DB2 Version 10 conversion mode (from Version 8 or Version 9.1) system (optional)" on page 636

Installation verification phases and programs

In each phase of the installation verification procedure (IVP), you must run specific programs. These programs need to be run sequentially by phase because the output of some jobs is used as input for subsequent jobs.

The IVP consist of eight phases: seven verification phases and one cleanup phase that drops sample objects (Phase 0). Each of the seven verification phases tests one or more DB2 functions or attachment facilities. Certain phases of the verification procedure might not apply to the environment in which your DB2 subsystem operates, so you might not need to complete all phases. In some cases, the steps and return codes differ when you run the fallback release and Version 10 phases. These differences are noted under the proper phase.

Run Phase 0 (job DSNTEJ0) only if you want to remove all the verification processing that you have completed so that you can begin the verification procedure again. Phases 1-3 test the TSO and batch environments, including user-defined functions. Phase 4 is for IMS users only, and Phase 5 is for CICS users only. Phase 6 sets up the sample tables and stored procedures for distributed processing. Phase 7 tests the LOB feature with sample tables, data, and programs.

prefix.SDSNSAMP contains the program source.

Each phase consists of one or more batch jobs and some phases also consist of online activities. You run the same jobs regardless of whether you are installing DB2 for the first time or migrating your Version 10 catalog to new-function mode. The installation CLIST tailored and loaded these jobs into *prefix*.NEW.SDSNSAMP, which you created during your installation or migration to new-function mode. The verification jobs use source code and data from the untailored *prefix*.SDSNSAMP library. Most DB2 sample objects have unique names to differentiate them from objects of previous releases. This allows sample programs for multiple releases to coexist. The JCL that is provided for IMS sets up transaction identifiers for the sample applications.

The following table shows the programs and jobs that are run in each phase of the verification procedure. These jobs are designed to run with minimal interaction on your part. However, before running these jobs, you should make any modifications that are suggested in the information about each job.

Table 102. Relationship of IVP phases to programs

Phase	Job	Program	Description
0	DSNTEJ0	DSNTIAD	Remove sample applications and sample schema authorizations
1	DSNTEJ1	DSNTIAD	Create tables
		DSN8CA	Assembler interface to call attach facility
		DSN8EAE1	Edit exit routine
		DSN8HUFF	Huffman compression exit routine

Table 102. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
		DSNUTILB	Utilities
	DSNTEJ1L	DSNTEP2	Dynamic SQL program
	DSNTEJ1P	DSNTEP2	Dynamic SQL program
	DSNTEJ1S[1]	DSNHSP	See note 1
	DSNTEJ1T[2]	DSNUPROC	See note 2
	DSNTEJ1U	DSNTIAD	Create Unicode table
		DSNUTILB	Load Unicode table
		DSNTEP2	Select Unicode table
2	DSNTEJ2A	DSNTIAD	Grant execution
		DSNTIAUL	Unload and load tables
		DSNUTILB	Utilities
	DSNTEJ2C	DCLGEN	Generate declarations
		DSNTIAD	Grant execution
		DSN8BC3	COBOL phone application
	DSNTEJ2D	DSNTIAD	Grant execution
		DSN8BD3	C phone application
	DSNTEJ2E	DSN8MDG	Prepare error message routine
		DSN8BECL	Prepare classes used by C++ phone application
		DSNTIAD	Grant execution
		DSN8BE3	C++ phone application
	DSNTEJ2F	DSNTIAD	Grant execution
		DSN8BF3	Fortran phone application
	DSNTEJ2H	DSN8EDXR	Sample application to format and display an XML record
		DSN8EDXU	Sample application to format and update an existing XML record
		DSN8EDXI	Sample application to insert an XML record
		DSN8EDXD	Sample application to delete an XML record
	DSNTEJ2P	DCLGEN	Generate declarations
		DSNTIAD	Grant execution
		DSN8BP3	PL/I phone application
	DSNTEJ2U	DSNTIAD	Register sample user-defined functions
		DSN8DUAD	C program for ALTDATE function (current date)
		DSN8DUCD	C program for ALTDATE function (given date)
		DSN8DUAT	C program for ALTTIME function (current time).

Table 102. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
		DSN8DUCT	C program for ALTTIME function (given time)
		DSN8DUCY	C program for CURRENCY function
		DSN8DUTI	C program for TABLE_NAME, TABLE_SCHEMA, and TABLE_LOCATION functions
		DSN8EUDN	C++ program for DAYNAME function
		DSN8EUMN	C++ program for MONTHNAME function
		DSNTEP2	Dynamic SQL program
		DSN8DUWF	User-defined table function sample
		DSN8DUWC	C program on client for user-defined table function sample
3		SPUFI	Sample SPUFI input
	DSNTEJ3C	DSNTIAD	Grant execution
		DSN8CC	COBOL interface to call attachment facility
		DSN8SCM	COBOL connection manager
		DSN8SC3	COBOL phone application
		DSN8HC3	COBOL organization application
	DSNTEJ3M	DSNTIAD	Drop sample MQT objects, create sample MQT objects, grant authority to MQT objects, and populate sample MQT tables
		DSNUTILB	Produce statistics for MQT
		DSNTEP2	Process EXPLAINs on sample MQT tables
	DSNTEJ3P	DSNTEP2	Dynamic SQL application
		DSNTIAD	Grant execution
		DSN8SPM	PL/I connection manager
		DSN8SP3	PL/I phone application
4	DSNTEJ4C	DSNTIAD	Grant execution
		DSN8IC*x*	Organization application
		DSN8MC*x*	Copy code
	DSNTEJ4P	DSNTIAD	Grant execution
		DSN8IP*x*	Organization, project applications
		DSN8MP*x*	Copy code
5	DSNTEJ5A	DSNTIAC	CICS SQLCA formatter front-end
	DSNTEJ5C	DSNTIAD	Grant execution
		DSN8CC*x*	Organization application
		DSN8MC*x*	Copy code
	DSNTEJ5P	DSNTIAD	Grant execution

Table 102. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
		DSN8CP*x*	Organization, project applications
		DSN8MP*x*	Copy code
6	DSNTEJ6	DSNTIAD	Update location column in the department table to the sample location entered at installation time
	DSNTEJ6D	DSN8ED1	Compile, link-edit, bind, and run a sample application program that calls a stored procedure
	DSNTEJ6O	DSN8DTS1, DSN8DTS2	Create and run sample for IBM Text Search for DB2 for z/OS
	DSNTEJ6P	DSN8EP1	Invoke the sample stored procedure
	DSNTEJ6R	DSN8ED8	Compile, link-edit, bind, and run a sample application that calls DSNUTILU, the DB2 Utilities Unicode parser stored procedure
	DSNTEJ6S	DSN8EP2	Create sample stored procedure
	DSNTEJ6T	DSN8ED2	Register, prepare, and bind the stored procedure sample application
	DSNTEJ6U	DSN8EPU	Compile, link-edit, bind, and run a sample application that calls DSNUTILS, a stored procedure for executing DB2 Utilities
	DSNTEJ6V	DSN8EE0	C++ class that calls DSNTIAR
		DSN8EE1	C++ class that calls DSNUTILS
		DSN8EE2	C++ client for DSN8EE1
	DSNTEJ6W	DSN8ED6	Call the WLM_REFRESH stored procedure
	DSNTEJ6Z	DSN8ED7	C class that calls DSNWZP stored procedure
	DSNTEJ61	DSN8EC1	Create sample ODBA stored procedure
	DSNTEJ62	DSN8EC2	Invoke sample ODBA stored procedure
	DSNTEJ63	DSN8ES1	Run sample external SQL procedure that calculates employee earnings.
	DSNTEJ64	DSN8ED3	Call the sample external SQL procedure, DSN8ES1 from a client
	DSNTEJ65	DSN8ED4	Call the external SQL procedure processor, DSNTPSMP
		DSN8ES2	Run external SQL procedure that calculates employee bonuses
		DSN8ED5	Call external SQL procedure, DSN8ES2
	DSNTEJ66	DSN8ES3	Create and alter a sample native SQL procedure
		DSN8ED9	Prepare and execute a sample caller of the sample native SQL procedure

Table 102. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
7	DSNTEJ7	DSNTIAD	Create sample LOB table
		DSNTIAD	Create synonyms, grant access to LOB tables
		DSNUTILB	Load sample LOB table
		DSNUTILB	Produce statistics for LOB table spaces
	DSNTEJ71	DSNTIAD	Grant access to plans
		DSN8DLPL	Populate sample LOB table with BLOB data
		DSN8DLTC	Verify contents of LOB table
	DSNTEJ73	DSNTIAD	Grant access to plans
		DSN8DLRV	C employee resume application
	DSNTEJ75	DSNTIAD	Grant access to plans
		DSN8DLPV	C employee photo application
	DSNTEJ76	DSN8CLPL	Create COBOL LOB sample table
		DSN8CLTC	Create synonyms, grant access to COBOL LOB tables
	DSNTEJ77	DSN8CLRV	Load sample COBOL LOB table
	DSNTEJ78	DSN8CLPV	Produce statistics for COBOL BLOB table spaces

Notes:

1. Job DSNTEJ1S, which contains the sample JCL to run the schema processor, is not a part of the sample applications to verify installation.
2. Job DSNTEJ1T, which adds rows to SYSIBM.SYSSTRINGS for character conversion purposes, is not a part of the sample applications to verify installation.

Related concepts:

"Falling back" on page 638

"Printing options for the sample application listings" on page 804

Related tasks:

Creating a schema by using the schema processor (DB2 Administration Guide)

Planning for verification

Before performing any of the verification phases, you must make certain decisions about your verification strategy.

About this task

DB2 system administrators and system administrators for ISPF, TSO, batch, IMS, and CICS must be involved in these decisions.

Procedure

To prepare for verification

- Determine the verification phases that you plan to perform.

Examine the description of each verification phase in this topic, and determine which phases apply to your needs.

- Identify any phases that you want to modify before you perform them.

  Verification is designed to run with little interaction on your part. This topic does not discuss how to modify any of the phases, but you can adapt any of the seven phases to your needs. If this is your intent, identify and describe any modifications you plan to make.

- Establish additional testing steps to complete the verification.

  The verification phases and the jobs that you run to perform them are valuable tools for testing DB2. They are not a substitute for a thorough subsystem test. You must plan and perform your own additional testing to complete the verification. To help you assess which additional tests might be necessary, examine the sample applications that are provided with DB2.

- Start any DB2 databases that are not currently started.

COBOL options to use with the installation verification procedures

The DB2 COBOL samples can be run with some compiler options, which control compilation.

IBM tested these options, which are shown in the table below.

If you have a problem executing the DB2 COBOL samples, ensure that your compiler options are consistent with the Enterprise COBOL options in the table below. Remember that if you are using CICS, the options that you need to use depend on the CICS environment. To verify that you are using the correct options in your CICS environment, refer to COBOL programming restrictions and requirements.

Table 103. Enterprise COBOL options

ADV	NOEXIT	NOTYPECHK
BUFSIZE(4096)	NOFASTSRT	NOVBREF
DATA(31)	NOFLAGMIG	NOWORD
FLAG(I)	NOFLAGSTD	NOXREF
INTDATE(ANSI)	NOIDLGEN	NUMPROC(NOPFD)
LANGUAGE(EN)	NOLIB	OBJECT
LINECOUNT(60)	NOLIST	OUTDD(SYSOUT)
NOADATA	NOMAP	PGMNAME(LONGUPPER)
NOAWO	NONAME	QUOTE
NOCMPR2	NONUMBER	RENT[3]
NOCOMPILE(S)	NOOFFSET	RMODE(AUTO)
NOCURRENCY	NOOPTIMIZE	SIZE(MAX)
NODBCS	NOSEQUENCE	SOURCE
NODECK	NOSSRANGE	SPACE(1)
NODUMP	NOTERM[1] or TERM[2]	TRUNC(STD)
NODYNAM	NOTEST	ZWB

Note:

[1] Refers to jobs DSNTEJ2C, DSNTEJ3C, and DSNTEJ4C only.

[2] Refers to job DSNTEJ5C only.

[3] See the CICS documentation for actual options to use.

For more detailed instructions, see Enterprise COBOL for z/OS and Language Environment Programming Guide (z/OS Language Environment Programming Guide).

Related reference:

DSNH (TSO CLIST) (DB2 Commands)

C and C++ language options to use with the installation verification procedures

The DB2 C and C++ samples can be run with some compiler options.

These options, which were tested by IBM, are shown in the following tables.

If you have a problem executing the DB2 C and C++ samples, ensure that your compiler options are consistent with the options in the following tables.

Table 104. C language options

NOAGGR	NOEVENTS	NOHWOPTS	NOMEMORY	NOPPONLY
NOALIAS	EXECOPS	NOINLINE[3]	NESTINC(255)	REDIR[1]
ARGPARSE[1]	NOEXPMAC	LIST	OBJECT	NORENT
NOCHECKOUT[2]	NOEXPORTALL	NOLOCALE	NOOE[1]	NOSEARCH
NOCSECT	FLAG(I)	NOLONGNAME	NOOFFSET	NOSHOWINC
NODECK	NOGONUMBER	NOLSEARCH	NOOPTIMIZE	SOURCE
NODLL	HALT(16)[1]	MAXMEM(2000)[4]	PLIST(HOST[1]	SPILL(128)[4]
NOSSCOMM	START	TARGET(LE)	TERMINAL	NOTEST[5]
NOUPCONV	XREF			

Note:
- [1] This option is used by IBM C/C++ for z/OS.
- [2] NOPPTRACE, PPCHECK, GOTO, ACCURACY, PARM, NOENUM, NOEXTERN, TRUNC, INIT, NOPORT, GENERAL.
- [3] AUTO, NOREPORT, 100, 1000.
- [4] This option is used only by IBM AD/Cycle C/370 V1R2.
- [5] SYM, BLOCK, LINE, NOPATH.

The table below contains the C++ language options.

Table 105. C++ language options

ARGPARSE	NOIDL[1]	NESTINC(255)[2]	NOSHOWINC	TERMINAL
NOATTRIBUTE	NOINFO	OBJECT	NOSOM	NOTEST
NOCSECT	NOINLRPT	NOOE[2]	SOMEINIT	XREF
EXECOPS	LANGLVL(EXTENDED)	NOOFFSET	NOSOMGS	HALT(16)
NOEXPMAC	NOLIST	OPTIMIZE(0)	SOURCE	MEMORY[2]
NOEXPORTALL	NOLOCALE[2]	PLIST(HOST)[2]	NOSRCMSG	NOSEQUENCE
FLAG(I)	LONGNAME	NOPPONLY	START[2]	TEMPINC
NOGONUMBER	MARGINS	REDIR	TARGET(LE)[2]	

Note:
- [1] This option is used by IBM C/C++ for MVS/ESA V3R1 and subsequent releases.
- [2] This option is used by IBM C/C++ for z/OS and subsequent releases.

The installation CLIST customizes C++ compiler parameters in sample job DSNTEJ2E if you have specified C++ for MVS/ESA V3R2 or a subsequent release on panel DSNTIPU.

PL/I options to use with the installation verification procedures

The DB2 PL/I samples can be run with some compiler options, which control compilation.

These options, which were tested by IBM, are shown in the table below.

If you have a problem executing the DB2 PL/I samples, ensure that your compiler options are consistent with these options.

Table 106. Enterprise PL/I options

CHARSET(60,EBCDIC)	LINECOUNT(55)	LMESSAGE	MARGINS(2,72,0)	NOAGGREGATE
NOATTRIBUTES	NOCOMPILE(S)	NOCOUNT	NOESD	NOFLOW
NOGONUMBER	NOGOSTMT	NOGRAPHIC	NOIMPRECISE	NOINCLUDE
NOINTERRUPT	NOLIST	NOMAP	NOMARGINI	NONEST
NONUMBER	NOOFFSET	NOOPTIMIZE	NOSTORAGE	NOSYNTAX(S)
NOTERMINAL	NOXREF	OBJECT	ODECK	SEQUENCE(73,80)
SIZE(506756)	SOURCE	STMT		

Phase 0: Deleting the sample objects (DSNTEJ0)

Phase 0 consists of one job, DSNTEJ0. It frees all plans, drops all objects, and deletes data sets so that Phase 1 can be run again.

Run Phase 0 (job DSNTEJ0) only if you want to remove all the verification processing that you have done so far so that you can begin the verification procedure again. When you complete the verification procedure, save the verification objects; you need them when you migrate to the next release of DB2.

If a sample application abends while running a utility, ensure that the utility is terminated before attempting to rerun the job.

Even when DSNTEJ0 runs successfully, some of the FREE, DROP, and DELETE commands often fail because the object was not created earlier. You can ignore these errors, even though they might generate return codes of 8 or 12. Check other errors.

If DSNTEJ0 runs successfully, it produces the return codes that are shown in the table below.

Table 107. DSNTEJ0 return codes

Step	PROCSTEP	Return code
PH00S01		0000, 0008, or 0012
PH00S02		0000, 0004, or 0008
PH00S03	DSNUPROC	0004 or 0008
PH00S04		0000, 0004, or 0008
PH00S05		0000, 0004, or 0008
PH00S06		0000, 0004, or 0008
PH00S07		0000, 0004, or 0008
PH00S08		0000, 0004, or 0008

If this job fails or abends, ensure that the user that is specified on the JOB statement is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP1 is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups. Then correct any other problems, and rerun the job from the last successful step.

If the subsystem data sets were deleted before the DB2 sample objects are deleted, you must delete the data sets by using access method services commands or TSO commands. In all of the following examples, *vcatalog* is the catalog alias name that you specified for the CATALOG ALIAS field on installation panel DSNTIPA2. The *y* is either I or J.

The following access method services commands, which can be executed under TSO, delete the Version 10 sample data sets:

```
DELETE 'vcatalog.DSNDBD.DSN8D10A.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81L.*.y0001.*'
DELETE 'vcatalog.DSNDBD.DSN8D10P.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D10U.*.y0001.*'
DELETE 'vcatalog.DSNDB04.STAFF.y0001.A001'
DELETE 'vcatalog.DSNDB04.TESTSTUF.y0001.A001'
DELETE 'vcatalog.DSNDBC.DSN8D81E.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81Y.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81X.*.y0001.*'
```

Related reference:

-TERM UTILITY (DB2) (DB2 Commands)

Phase 1: Creating and loading sample tables

Phase 1 of installation or migration verification consists of four jobs: DSNTEJ1, DSNTEJ1L, DSNTEJ1P, and DSNTEJ1U.

DSNTEJ1 invokes program DSNTIAD, which creates objects during the verification procedure. Run DSNTIEJ1 before running any other sample jobs.

DSNTEJ1L and DSNTEJ1P prepare and invoke program DSNTEP2, which lists the contents of the sample tables. The difference between the jobs is that DSNTEJ1P requires the PL/I compiler and allows you to customize DSNTEP2.

DSNTEJ1U creates, populates, and tests a sample Unicode database.

Job DSNTEJ1

Job DSNTEJ1 creates all objects that are to be used by the sample verification jobs. Run this job before any other verification job.

This job creates a sample edit exit routine and module called DSN8EAE1. It then processes SQL statements that cause DB2 to load and execute the module. If the module resides in a library that DB2 accesses from the link list or LLA, DSNTEJ1 might fail at job step PH01S13 with abend S04E reason code 0E403180 because of unavailable resource (reason code 00C9008A). In this event, refresh the link list or LLA and then restart DSNTEJ1 at job step PH01S13 (PROCSTEP DSNUPROC).

From V8 If you are migrating from DB2 Version 8 and you set the RRF subsystem parameter to ENABLE, use DSN8EAE9 rather than DSN8EAE1 as the name of the sample edit exit routine and module. Setting the RRF subsystem parameter to ENABLE specifies that newly created table spaces are to store data in reordered row format. If DB2 has already loaded the DSN8EAE1 edit exit routine for Version 8, sample job DSNTEJ1 might fail in job step PH01S13 with ABEND0C4 reason code 11. The failure occurs because the Version 8 DSN8EAE1 module supports only the basic row format. DSN8EAE1 is automatically changed to DSN8EAE9 in job DSNTEJ1 when you run the installation CLIST in ENFM mode. From V8

The job consists of the following steps.

Table 108. Steps in job DSNTEJ1

Step	Function
1-4	Creates all objects (storage group, databases, table spaces, tables, indexes, and views) that are used by the samples.
5	Drops synonyms.
6	Creates synonyms and grants authorization on objects to PUBLIC AT ALL LOCATIONS. This step creates synonyms for the sample tables, indexes, and views, so that the currently running authorization ID can execute the sample application and grant appropriate authority. The sample dynamic SQL program DSNTIAD processes the DB2 object definitions in this step and several others.
7	Uses the ASMCL procedure to create DSN8EAE1, an edit exit routine. From V8 If you migrated from DB2 Version 8, verify that this routine is called DSN8EAE9. From V8
8	Assembles and link-edits DSNHUFF.
9	Assembles and link-edits DSN8FPRC, a sample field procedure.
10	Prepares the sample call attachment facility assembler interface. You must link-edit ISPLINK, the ISPF interface module, with this CAF sample load module. To do this, ensure that the link-edit SYSLIB statement that retrieves the ISPF load module library in procedure DSNHASM is not commented out.
11	Creates the sample utility list.
12	Loads the programming-related tables by using the LOAD utility.
13	Loads the sample tables by using the LOAD utility.
14	Checks data for referential integrity.
15	Establishes a quiesce point by using both log and image copies.
16	Makes an image copy of all the sample tables by using the COPY utility.
17	Establishes another quiesce point by using only image copies.
18	Reorganizes a table space and compiles statistics on all table spaces by using the REORG and RUNSTATS utilities.
19	Performs a REORG TABLESPACE with SHRLEVEL CHANGE.
20	Loads the sample tables by using the LOAD utility.

Table 108. Steps in job DSNTEJ1 (continued)

Step	Function
21	Sets the CURRENT RULES register and adds a check constraint using ALTER TABLE.
22	Checks data for referential integrity.
23	Checks data for check integrity.
24-27	Performs the operations in steps 15-18 except for the REORG on partition 3 of the Employee table space.
28	Unloads data from a partitioned table.
29	Reduces the partition key on the fourth partition of table space DSN8S81E.
30	Processes an online schema change to add a fifth partition to table space DSN8S81E.
31	Reorganizes a partitioned table space.
32	Performs various online schema changes, including lengthening a character field and converting an integer field to a decimal field.
33	Reorganizes table spaces DSN8S91R and DSN8S91S to reset advisory REORG-pending (AREO) status

If DSNTEJ1 runs successfully, it produces the return codes that are shown in the following table.

Table 109. DSNTEJ1 return codes

Step	PROCSTEP	Return code
PH01S01		0000
PH01S02		0000
PH01S03		0000
PH01S04		0000
PH01S05		0000
PH01S06		0000
PH01S07		
	ASM	0000
	LKED	0000
PH01S08		
	ASM	0000
	LKED	0000
PH01S09		
	ASM	0000
	LKED	0000
PH01S10		
	PC	0004
	ASM	0000
	LKED	0000
PH01S11	IEBGENER	0000
PH01S12	DSNUPROC	0000
PH01S13	DSNUPROC	0004
PH01S14	DSNUPROC	0000

Table 109. DSNTEJ1 return codes (continued)

Step	PROCSTEP	Return code
PH01S15	DSNUPROC	0000
PH01S16	DSNUPROC	0000
PH01S17	DSNUPROC	0000
PH01S18	DSNUPROC	0000 or 0004
PH01S19	DSNUPROC	0000
PH01S20	DSNUPROC	0004
PH01S21		0004
PH01S22	DSNUPROC	0004
PH01S23		0000
PH01S24	DSNUPROC	0000
PH01S25	DSNUPROC	0000
PH01S26	DSNUPROC	0000
PH01S27	DSNUPROC	0000
PH01S28	DSNUPROC	0000
PH01S29		0000 or 0004
PH01S30		0000
PH01S31	DSNUPROC	0000
PH01S32		0000
PH01S33	DSNUPROC	0000

DB2 issues the following message for every SQL statement, except for the drop synonym and insert statements:

```
DSNT400I SQLCODE = 0, SUCCESSFUL EXECUTION
```

If the synonyms in the DROP SYNONYM statements are not defined, SQL return codes of -204 result. The INSERT statements violate a check constraint on the EMP table. This results in an SQL return code of -545.

Job DSNTEJ1L

DSNTEJ1L link-edits the DSNTEP2 object deck (DSNTEP2L) to create an executable load module DSNTEP2.

DSNTEJ1L link-edits the DSNTEP4 object deck (DSNTEP4L) to create an executable load module DSNTEP4.

DSNTEJ1L also binds and runs programs DSNTEP2 and DSNTEP4. DSNTEP2 lists the sample database tables and views. DSNTEP2 is a dynamic PL/I program that accepts SQL statements. DSNTEP2 produces a listing of the results of SELECT statements. DSNTEP4 is identical to DSNTEP2, except that it uses multi-row fetch.

Job DSNTEJ1L requires the Language Environment link-edit and run-time libraries. DSNTEJ1L does not require the PL/I compiler.

If DSNTEJ1L runs successfully, it produces the return codes that are shown in the table below.

Table 110. DSNTEJ1L return codes

Step	PROCSTEP	Return code
PH01PS01		0000
PH01PS02		0000 or 0004
PH01PS03		0000
PH01PS04		0000 or 0004

You can compare the output from this job with the sample output for DSNTEJ1L, which is found in member DSN8TJ1L in your *prefix*.SDSNIVPD data set.

If you run DSNTEJ1P before DSNTEJ1L, you can expect step PH01PS02 of job DSNTEJ1L to produce a return code of 0004 and the following message:

```
SQLWARNING ON GRANT   COMMAND, EXECUTE FUNCTION
  RESULT OF SQL STATEMENT:
  DSNT404I SQLCODE = 562, WARNING:  A GRANT OF A PRIVILEGE WAS IGNORED
           BECAUSE THE GRANTEE ALREADY HAS THE PRIVILEGE FROM THE GRANTOR
```

If either DSNTEJ1 or DSNTEJ1L fails or abends, ensure that the user that is specified in the JOB statements is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP1 is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups.

Then, correct any other problems. Before rerunning DSNTEJ1, run DSNTEJ0 to drop the sample data. If you rerun DSNTEJ1L, rerun it from the last successful step.

Related reference:

DSNTEP2 and DSNTEP4 (DB2 Application programming and SQL)

Job DSNTEJ1P

If you have run DSNTEJ1L, you do not need to run DSNTEJ1P because these jobs produce the same results. The major difference is that DSNTEJ1P uses the PL/I compiler and allows you to customize DSNTEP2 and DSNTEP4.

DSNTEJ1P precompiles, compiles, and link-edits PL/I program DSNTEP2. This program then lists the sample database tables and views. It is a dynamic PL/I program that accepts SQL statements. It produces a listing of the results of SELECT statements.

If DSNTEJ1P runs successfully, it produces the return codes that are shown in the table below.

Table 111. DSNTEJ1P return codes

Step	PROCSTEP	Return code
PH01PS01		
	PPLI	0000
	PC	0000
	PLI	0000
	PLKED	0004
	LKED	0000

Table 111. DSNTEJ1P return codes (continued)

Step	PROCSTEP	Return code
PH01PS02		0000 or 0004
PH01PS03		
	PPLI	0000
	PC	0000
	PLI	0000
	PLKED	0004
	LKED	0000
PH01PS04		0000 or 0004

You can compare the output from this job with the sample output for DSNTEJ1P, which is found in member DSN8TJ1P in your *prefix*.SDSNIVPD data set.

If you run DSNTEJ1L before DSNTEJ1P, you can expect step PH01PS02 of job DSNTEJ1P to produce a return code of 0004 and the following message:

```
SQLWARNING ON GRANT   COMMAND, EXECUTE FUNCTION
  RESULT OF SQL STATEMENT:
  DSNT404I SQLCODE = 562, WARNING:  A GRANT OF A PRIVILEGE WAS IGNORED
           BECAUSE THE GRANTEE ALREADY HAS THE PRIVILEGE FROM THE GRANTOR
```

If either DSNTEJ1 or DSNTEJ1P fails or abends, ensure that the user that is specified in the JOB statements is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP1 is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups.

Then, correct any other problems. Before rerunning DSNTEJ1, run DSNTEJ0 to drop the sample data. If you rerun DSNTEJ1P, rerun it from the last successful step.

Related reference:

DSNTEP2 and DSNTEP4 (DB2 Application programming and SQL)

Job DSNTEJ1U

DSNTEJ1U creates a database, table space, and table with Unicode CCSID.

DSNTEJ1U loads data into the table from a data set that contains a full range of characters in an EBCDIC Latin-1 code page, which results in a mix of single and double-byte characters in the Unicode table. It then runs DSNTEP2 to select and display the data in hex format.

If DSNTEJ1U runs successfully, it produces the return codes that are shown in the table below.

Table 112. DSNTEJ1U return codes

Step	PROCSTEP	Return code
PH01US01		0000
PH01US02		0000
PH01US03	DSNUPROC	0000

Table 112. DSNTEJ1U return codes (continued)

Step	PROCSTEP	Return code
PH01US04		0000

Phase 2: Testing the batch environment

Phase 2 of the installation or migration verification process consists of several jobs. Run the jobs to test the program preparation procedures for various languages.

If any of the Phase 2 jobs fail or abend, be sure that the user that is specified in the JOB statements is authorized. Use the name that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP1. Then correct any other problems, and rerun the jobs from the last successful step.

Job DSNTEJ2A

DSNTEJ2A tests the assembler program preparation procedures.

This job prepares and invokes program DSNTIAUL, which demonstrates the use of dynamic SQL in assembler to unload the data from tables or views. It also generates LOAD utility statements so the data can be loaded into another table. DSNTEJ2A then uses the LOAD utility to put data into copies of the unloaded tables.

If DSNTEJ2C runs successfully, it produces the return codes that are shown in the table below.

Table 113. DSNTEJ2A return codes

Step	PROCSTEP	Return code
PREPUNL		
	PC	0000
	ASM	0000 or 0004
	LKED	0000 or 0004
BINDUNL		0000
DELETE		0000
CREATE		0000
UNLOAD		0000
EDIT		0000
LOAD	DSNUPROC	0004

You can compare the output from this job with the sample output for DSNTEJ2A found in member DSN8TJ2A in your *prefix*.SDSNIVPD data set.

Related reference:

DSNTIAUL (DB2 Application programming and SQL)

Job DSNTEJ2C

Job DSNTEJ2C tests the COBOL program preparation procedures. This job runs the phone application.

The phone application processes a table of telephone numbers, executing various types of SELECT statements and producing the corresponding listings. It can also update a phone number.

If DSNTEJ2C runs successfully, it produces the return codes that are shown in the following table.

Table 114. DSNTEJ2C return codes

Step	PROCSTEP	Return code
PH02CS01		0000
PH02CS02		
	PC	0004
	COB	0000 or 0004
	PLKED	0004
	LKED	0000 or 0004
PH02CS03		
	PC	0000
	COB	0000 or 0004
	PLKED	0004
	LKED	0000
PH02CS04		0000

You can compare the output from this job with the sample output for DSNTEJ2C found in member DSN8TJ2C in your *prefix*.SDSNIVPD data set.

Related concepts:

"COBOL options to use with the installation verification procedures" on page 749

Related tasks:

"Working with the phone application scenario" on page 814

Job DSNTEJ2D

Job DSNTEJ2D tests the C program preparation procedures.

You must have sequence numbering on to run this job from an ISPF session. The C job runs only the phone application.

If DSNTEJ2D runs successfully, it produces the return codes that are shown in the table below.

Table 115. DSNTE2D return codes

Step	PROCSTEP	Return code
PH02DS01		
	PC	0004
	C	0000
	PLKED	0000 or 0004
	LKED	0004
PH02DS02		
	PC	0000
	C	0000
	PLKED	0000 or 0004
	LKED	0000 or 0004
PH02DS03		0000

You can compare the output from this job with the sample output for DSNTEJ2D found in member DSN8TJ2D in your *prefix*.SDSNIVPD data set.

Related concepts:

"Job DSNTEJ2C" on page 758

Job DSNTEJ2E

Job DSNTEJ2E tests the C++ program preparation procedures.

You must have sequence numbering on to run this job from an ISPF session. The C++ job runs only the phone application.

If DSNTEJ2E runs successfully, it produces the return codes that are shown in the table below.

Table 116. DSNTE2E return codes

Step	PROCSTEP	Return code
PH02ES01		
	PC	0004
	C	0000
	PLKED	0000 or 0004
	LKED	0004
PH02ES02		
	PC	0000
	CP	0000
	PLKED	0000 or 0004
	LKED	0004
PH02ES03		
	PC	0004
	CP	0000
	PLKED	0000 or 0004
	LKED	0000
PH02ES04		0000

You can compare the output from this job with the sample output for DSNTEJ2E found in member DSN8TJ2E in your *prefix*.SDSNIVPD data set.

Related concepts:

"Job DSNTEJ2C" on page 758

Job DSNTEJ2F

Job DSNTEJ2F tests the Fortran program preparation procedures.

The FORTRAN job runs only the phone application.

If DSNTEJ2F runs successfully, it produces the return codes that are shown in the table below.

Table 117. DSNTE2F return codes

Step	PROCSTEP	Return code
PH02FS01		
	PC	0004
	ASM	0000
	LKED	0004

Table 117. DSNTE2F return codes (continued)

Step	PROCSTEP	Return code
PH02FS02		
	PC	0000
	FORT	0000
	LKED	0000
PH02FS03		0000

You can compare the output from this job with the sample output for DSNTEJ2F found in member DSN8TJ2F in your *prefix*.SDSNIVPD data set.

Related concepts:

"Job DSNTEJ2C" on page 758

Job DSNTEJ2H

Job DSNTEJ2H tests the XML program preparation procedures.

If this job runs successfully, it produces the following return codes:

Table 118. DSNTEJ2H return codes

Step	PROCSTEP	Return code
PH02HS01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02HS02	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02HS03	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02HS04	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02HS05		0000
PH02HS06		0000
PH02HS07		0000
PH02HS08		0000
PH02HS09		0000

Job DSNTEJ2P

Job DSNTEJ2P tests the PL/I program preparation procedures.

The PL/I job runs the phone application.

If DSNTEJ2P runs successfully, it produces the return codes that are shown in the table below.

Table 119. DSNTEJ2P return codes

Step	PROCSTEP	Return code
PH02PS01		0000
PH02PS02		
	PPLI	0000
	PC	0004
	PLI	0000
	PLKED	0004
	LKED	0004
PH02PS03		
	PPLI	0000
	PC	0000
	PLI	0000
	PLKED	0004
	LKED	0000
PH02PS04		
		0000
PH02PS05		0000

You can compare the output from this job with the sample output for DSNTEJ2P found in member DSN8TJ2P in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2U

DSNTEJ2U prepares and tests several sample user-defined functions and a driver program that exercises the user-defined functions.

In order for the installation CLIST to generate job DSNTEJ2U, you must complete the following steps during installation:

- Specify that the host has access to C/C++ for z/OS on installation panel DSNTIPU.
- Specify the name of the default WLM environment on installation panel DSNTIPX.

The sample user-defined functions are:

Function
Description

ALTDATE
Returns the current date in a user-specified format or converts a user-specified date from one format to another.

ALTTIME
Returns the current time in a user-specified format or converts a user-specified time from one format to another.

CURRENCY
Formats a floating point number as a currency value.

DAYNAME
Returns the day of the week for a user-specified date in ISO format.

MONTHNAME
Returns the month for a user-specified date in ISO format.

TABLE_LOCATION
Returns the location name of a table, view, or undefined object found after resolving aliases for a user-specified object.

TABLE_NAME
Returns the name of a table, view, or undefined object found after resolving aliases for a user-specified object.

TABLE_SCHEMA
Returns the schema name of a table, view, or undefined object found after resolving aliases for a user-specified object.

WEATHER
Returns sample weather data obtained from a TSO data set by way of demonstrating the usefulness of a user-defined function table function.

If you do not have C++ installed, skip steps PH02US08 and PH02US09. Also remove all statements that refer to DAYNAME and MONTHNAME from part DSNTESU in the *prefix*.SDSNSAMP library.

Job DSNTEJ2U consists of the steps that are listed in the table below.

Table 120. Steps in job DSNTEJ2U

Step	Function
1	Drops all specific sample user-defined functions.
2	Creates and registers all sample user-defined functions. Grants EXECUTE authority to PUBLIC for the sample user-defined functions.
3-10	Prepares the eight external programs used by specific user-defined functions.
11	Binds the package for the TABLE_NAME, TABLE_SCHEMA, and TABLE_LOCATION functions. These are the only sample functions that issue SQL statements. This step also grants EXECUTE authority on these two samples to PUBLIC.
12	Invokes DSNTEP2 to exercise the sample user-defined functions.
13	Prepares DSN8DUWF, the external module for the sample user defined table function, WEATHER.
14	Prepares DSN8DUWC, a sample client function for statically invoking the WEATHER user-defined table function.
15	Binds the package and plan for DSN8DUWC and grants the necessary authorities.
16	Invokes DSN8DUWC.

If DSNTEJ2U runs successfully, it produces the return codes that are shown in the table below.

Table 121. DSNTEJ2U return codes

Step	PROCSTEP	Return code
PHO2US01		0000
PH02US02		0000 or 0004

Table 121. DSNTEJ2U return codes (continued)

Step	PROCSTEP	Return code
PH02US03 - PH02US07		
	PC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH02US08 - PH02US09		
	PC	0004
	CP	0000
	PLKED	0004
	LKED	0000
PH02US10		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02US11		0000
PH02US12		0004
PH02US13		
	PC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH02US14		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH02US15		0000 or 0004
PH02US16		0000

You can compare the output from this job with the sample output for DSNTEJ2U found in member DSN8TJ2U in your *prefix*.SDSNIVPD data set.

Related concepts:

Sample user-defined functions (DB2 SQL)

User-defined functions (DB2 Application programming and SQL)

Related reference:

"Data set names panel 2: DSNTIPU" on page 300

"Routine parameters panel: DSNTIPX" on page 453

Phase 3: Testing SPUFI, DRDA access, dynamic SQL, and TSO

Phase 3 of the installation or migration verification process allows you to test SPUFI (SQL Processor Using File Input) and DRDA access. You can also run dynamic SQL statements, run the phone application in TSO, and bind packages at the local and remote locations.

SPUFI is a facility of DB2I. You can only run SPUFI under ISPF. You can run dynamic SQL whether or not you have ISPF.

Related tasks:
"Testing SPUFI"

Testing SPUFI

During Phase 3 of the installation or migration verification process, you can test SPUFI to ensure that it is working properly.

Procedure

To test SPUFI:

1. Log on to TSO.
2. Enter ISPF (this might be done for you, depending on your site's standard practice).
3. On the DB2I defaults panel, change the DB2 name to the DB2 subsystem name you entered on panel DSNTIPM during installation. Then select DB2I on the ISPF Primary Option Menu.
4. Select SPUFI on the DB2I menu.
5. Enter the library name '*prefix*.NEW.SDSNSAMP(DSNTESA)' as input to SPUFI on line 1, the DATASET NAME parameter. If your site uses the comma as a decimal point, the library name entered must be for the tailored version of job DSNTESA that was modified by the installation CLIST.
6. Define an output data set name on line 4, the output DATASET NAME parameter of the panel. This allows you to review the output.
7. Press ENTER, and examine the results. These SQL statements require a significant amount of DB2 processing; you could have to wait for the output.
8. Run steps 5, 6, and 7 three times:
   - Once with member DSNTESA, which uses a set of SQL statements to create a short-lived table space and table.
   - Once with member DSNTESC, which creates objects that support EXPLAIN information. If you are migrating from an earlier DB2 version, DSNTESC is customized by the DB2 installation CLIST to migrate the Version 8 or Version 9.1 PLAN_TABLE, DSN_FUNCTION_TABLE, DSN_STATEMNT_TABLE, and DSN_STATEMENT_CACHE_TABLE to DB2 Version 10 with a schema name of DSN81010. If your Version 8 or Version 9.1 tables have a different schema name, edit DSNTESC to use your schema name. Migration of these tables from a prior release should not be performed more than once.
   - Once with member DSNTESE, which retrieves the EXPLAIN information.

What to do next

If any step fails or abends, be sure that the DB2 subsystem name is specified in the DB2 NAME field on the DB2I Defaults panel.

If you must drop a Version 8, Version 9.1, or Version 10 PLAN_TABLE, remove the appropriate comments from the job to issue the DROP statements.

Also, make sure that the user ID that you are using is authorized. If the name that you specified for either the SYSADM or SYSADM2 subsystem parameter is a primary authorization ID, use this name. If the sample authorization exit and RACF are installed, and both SYSADM and SYSADM2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of

these RACF groups. Then, correct any other problems and rerun the scenario from the last successful step.

Related concepts:

"Dynamic SQL statements: DSNTESA, DSNTESQ" on page 830

Related tasks:

Creating EXPLAIN tables (DB2 Performance)

Related reference:

EXPLAIN tables (DB2 Performance)

Dynamic SQL and the ISPF/CAF application

The Phase 3 jobs of the installation or migration verification process install the ISPF/CAF sample application. This sample consists of an assembler or COBOL call attachment facility (CAF) interface, a connection manager program, and the phone application. The sample also includes the distributed application that uses DRDA access.

Job DSNTEJ1 prepares the assembler interface, and job DSNTEJ3C prepares the COBOL interface. The connection manager program and the phone application each exist in COBOL and PL/I. Job DSNTEJ3C prepares the COBOL version; job DSNTEJ3P prepares the PL/I version. The distributed application using DRDA access is written in COBOL.

Related concepts:

"COBOL options to use with the installation verification procedures" on page 749

Jobs DSNTEJ3C and DSNTEJ3P

Job DSNTEJ3C prepares a COBOL ISPF/CAF sample application and job DSNTEJ3P prepares a PL/I ISPF/CAF sample application.

To prepare for the distributed sample application, job DSNTEJ3C binds a package at the local (client) and remote (server) subsystems. For concurrent installations at two DB2 locations, designate one location as the client and the other location as the server. Both the client and server subsystems must be running DB2 Version 10. Because DSNTEJ3C does a remote bind, you must set up your client and server systems for remote communication before running this job. The sample jobs DSNTEJ1 and DSNTEJ2C or DSNTEJ2P must have been run on the server subsystem.

PH03CS06, the last job step of DSNTEJ3C, binds the sample organization application at a server DB2 subsystem if the following conditions are met. If these conditions are not met, job step PH03CS06 is removed by the installation CLIST.

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY and it does not match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

If DSNTEJ3C runs successfully, it produces the return codes that are shown in the following table.

Table 122. DSNTEJ3C return codes

Step	PROCSTEP	Return code
PH03CS01		
	PC	0004
	COB	0000 or 0004
	PLKED	0004
	LKED	0000
PH03CS02		
	PC	0004
	COB	0000 or 0004
	PLKED	0004
	LKED	0000
PH03CS03		
	PC	0000
	COB	0004
	PLKED	0004
	LKED	0000
PH03CS04		
	PC	0000 or 0004
	COB	0000 or 0004
	PLKED	0004
	LKED	0000
PH03CS05		0000 or 0004
PH03CS06		0000 or 0004

Step PH03CS06 can give a return code of 0004 if sample job DSNTEJ1 was not run on the remote system. For testing, you should run job DSNTEJ1 on the remote system.

If job DSNTEJ3P runs successfully, it produces the return codes that are shown in the following table.

Table 123. DSNTEJ3P return codes

Step	PROCSTEP	Return code
PH03PS01		
	PPLI	0000
	PC	0004
	PLI	0000
	PLKED	0004
	LKED	0000
PH03PS02		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0000
PH03PS03		0000

Related tasks:

"Connecting distributed database systems" on page 877

Related reference:

"Job editing panel: DSNTIPY" on page 488

Job DSNTEJ3M

Job DSNTEJ3M creates, populates, and processes a database that demonstrates the use of DB2 materialized query tables (MQTs).

Before you run job DSNTEJ3M, you must create a PLAN_TABLE. To create this table, specify the member DSNTESC as an input data set name to SPUFI.

If DSNTEJ3M runs successfully, it produces the return codes that are shown in the table below.

Table 124. DSNTEJ3M return codes

Step	PROCSTEP	Return code
PH03MS00		0000
PH03MS01		0000
PH03MS02		0000
PH03MS03		0000 or 0004
PH03MS04		0000
PH03MS05		0000
PH03MS06		0000
PH03MS07		0000

Starting an application in an ISPF/TSO environment

You must have access to ISPF load module libraries to run the ISPF/CAF sample application.

Procedure

To start an application in a TSO environment:

1. Enter a CALL command for option 6 of the ISPF primary option menu.
2. To start the COBOL phone sample version of the connection manager, enter the following command:

   ```
   CALL 'prefix.RUNLIB.LOAD(DSN8SCM)'
   ```

3. To start the PL/I phone sample version of the connection manager, enter the following command:

   ```
   CALL 'prefix.RUNLIB.LOAD(DSN8SPM)'
   ```

   After you enter one of these commands, DB2 displays the sample applications panel, which is shown in the following figure.

```
            DB2 SAMPLE APPLICATIONS MENU
===>

Select one of the following options and press enter.

 1  COBOL PHONE SAMPLE     (DB2 ISPF COBOL Application)
 2  PL/I  PHONE SAMPLE     (DB2 ISPF PL/I  Application)
 3  COBOL ORGANIZATION     (DB2 ISPF COBOL Application)

 4  C   EMPLOYEE RESUME    (DB2 ISPF C Application)
 5  C   EMPLOYEE PHOTO     (DB2 ISPF & GDDM C Application)

 6  COBOL EMPLOYEE RESUME  (DB2 ISPF COBOL Application)
 7  COBOL EMPLOYEE PHOTO   (DB2 ISPF & GDDM COBOL Application)

SPECIFY DB2 SUBSYSTEM NAME ===> DSN

PRESS:   END TO EXIT
```

Figure 80. Initial panel of the ISPF/CAF application

Choosing option 1 or 2 on the sample applications panel during Phase 3 invokes either the COBOL or the PL/I version of the phone application. Choosing option 3 on the sample applications panel during Phase 6 invokes the COBOL organization application, which uses DRDA access to distributed data.

Choosing options 4, 5, 6, and 7 on the sample applications panel during Phase 7 invokes the "Employee Resume" and "Employee Photo" applications, which processes LOB data. Options 4 and 5 access the C language sample applications, and options 6 and 7 access the COBOL language sample applications. You must run job DSNTEJ73 before you can access option 4. You must run job DSNTEJ75 before you can access option 5. You must run job DSNTEJ77 before you can access option 6. You must run job DSNTEJ78 before you can access option 7.

Related tasks:

"Working with the distributed organization application scenario" on page 818

"Working with the employee resume and photo scenarios" on page 826

"Make DB2I panels, messages, and load modules available to ISPF and TSO" on page 586

"Working with the phone application scenario" on page 814

Phase 4: Testing the IMS environment

Phase 4 of the installation or migration verification process installs the sample IMS transactions for both COBOL and PL/I.

In the PL/I version, the phone application discussed in Phase 2 is also installed as an online transaction.

Related tasks:

"Working with the phone application scenario" on page 814

Jobs DSNTEJ4C and DSNTEJ4P

Job DSNTEJ4C is for COBOL; DSNTEJ4P is for PL/I. Both jobs complete the same functions.

Jobs DSNTEJ4C and DSNTEJ4P complete the following tasks:

- Precompile, compile, and link-edit the IMS online applications.
- Bind the IMS online applications.

- Create the message format service (MFS) panels for the online applications.
- Run the required PSBGEN and ACBGEN.

Select the proper job and define the applications and transactions to IMS. Member DSN8FIMS in *prefix*.SDSNSAMP contains information to assist in the definition step.

The verification transactions are single mode, single segment, and nonconversational.

Recommendation: Use SSM error option R because the program handles any errors. A resource translation table is not required.

Invoke the transaction by using the FORMAT command. The programs accept several lines of input on the first panel and display the results after you press ENTER.

If DSNTEJ4C runs successfully, it produces the return codes that are shown in the following table.

Table 125. DSNTEJ4C return codes

Step	PROCSTEP	Return code
PH04CS01		
	PC	0004
	COB	0000
	PLKED	0004
	LKED	0004
PH04CS02		
	PC	0000
	COB	0000
	PLKED	0004
	LKED	0004
PH04CS03		
	PC	0000
	COB	0000
	PLKED	0004
	LKED	0004
PH04CS04		0000
PH04CS05		0000
PH04CS06		
	S1	0000
	S2	0004
PH04CS07		
	S1	0000
	S2	0004
PH04CS08		
	C	0000
	L	0000
PH04CS09	G	0000

For DSNTEJ4C, the warning code that is expected from the precompiler step PH04CS01 is:

```
DB2 SQL PRECOMPILER     MESSAGES
DSNH0531 W   NO SQL STATEMENTS WERE FOUND
```

If DSNTEJ4P runs successfully, it produces the return codes that are shown in the following table.

Table 126. DSNTEJ4P return codes

Step	PROCSTEP	Return Code
PH04PS01		
	PPLI	0000
	PC	0004
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS02		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS03		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS04		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS05		
	PPLI	0000
	PC	0004
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS06		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS07		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH04PS08		0000
PH04PS09		0000
PH04PS10		
	S1	0000
	S2	0004

Table 126. DSNTEJ4P return codes (continued)

Step	PROCSTEP	Return Code
PH04PS11		
	S1	0000
	S2	0004
PH04PS12		
	S1	0000
	S2	0000 or 0004
PH04PS13		
	S1	0000
	S2	0000 or 0004
PH04PS14		
	S1	0000
	S2	0000 or 0004
PH04PS15		
	S1	0000
	S2	0000 or 0004
PH04PS16		
	C	0000
	L	0000
PH04PS17	G	0000
PH04PS18		
	C	0000
	L	0000
PH04PS19	G	0000
PH04PS20		
	C	0000
	L	0000
PH04PS21	G	0000

For DSNTEJ4P, the warning code expected from the precompiler step PH04PS01 is:

```
DB2 SQL PRECOMPILER     MESSAGES
DSNH0531 W   NO SQL STATEMENTS WERE FOUND
```

If either job DSNTEJ4C or job DSNTEJ4P fails or abends, rerun the jobs from the last successful step.

Related concepts:

"COBOL options to use with the installation verification procedures" on page 749

Starting an application in an IMS environment

After logging on to IMS, you can start the organization application or project application.

Procedure

To start an application in an IMS environment:

Enter an IMS FORMAT command
The FORMAT commands are:
- /FORMAT DSN8IPGO, which starts the PL/I organization version
- /FORMAT DSN8ICGO, which starts the COBOL organization version.

When you enter either of these two commands, the panel that is shown in the following figure is displayed.
When the following command is entered, the panel that is shown in the figure

```
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
```

Figure 81. Organization version of FORMAT command display

below is displayed.

```
/FORMAT DSN8IPFO
```

starts the PL/I projects version.

```
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
```

Figure 82. Project version of FORMAT command display

Using the phone application in IMS

When you use IMS, information is interactively processed.

Procedure

To begin using the phone application in IMS:

Clear the screen and type in a FORMAT command. The FORMAT command that starts PL/I phone application is:

```
/FORMAT DSN8IPNO
```

When the FORMAT command is entered, the panel that is shown in the following figure is displayed.

```
---------------------------- TELEPHONE DIRECTORY ----------------------------


                          LAST  NAME ==>

                          FIRST NAME ==>


LAST NAME           :  *  FOR LIST OF ENTIRE DIRECTORY
                       %  FOR GENERIC LIST (EX. K% = ALL  K - NAMES)
FIRST NAME(OPTIONAL):  %  FOR GENERIC LIST
```

Figure 83. Starting the phone application

Phase 5: Testing the CICS environment

Phase 5 of the installation or migration verification process tests the CICS environment. Phase 5 installs the sample applications for COBOL and PL/I, and it prepares the CICS SQLCA formatter front-end.

Job DSNTEJ5A

Job DSNTEJ5A assembles and link-edits DSNTIAC, the CICS SQLCA formatter front-end. DSNTEJ5A also assembles and link-edits the RCT and optionally adds the sample definitions to the CSD.

Use DSNTIAC as an alternative to DSNTIAR when you want CICS services to do storage handling and program loading. If you are using CICS Version 4 or CICS Transaction Server 1.1, you need to modify job DSNTEJ5A to use steps DSN8FRCT and DSN8FRDO. You might need to tailor step DSN8FRDO for your system.

If DSNTEJ5A runs successfully, it produces the return codes that are shown in the table below.

Table 127. DSNTEJ5A return codes

Step	PROCSTEP	Return code
PH05AS01		0000
PH05AS02		0000
PH05AS03		0000

Jobs DSNTEJ5C and DSNTEJ5P

Job DSNTEJ5C installs the sample application transactions in COBOL and prepares the organization application. Job DSNTEJ5P installs the transactions in PL/I and prepares the organization, project, and phone applications.

Both phase 5 jobs perform the following functions:

- Compile and link-edit the CICS online applications
- Bind the CICS online applications
- Create the BMS maps for the online applications.

Select the proper job, and define transactions, programs, and BMS maps to CICS.

prefix=.SDSNSAMP members DSN8FPPT, DSN8FPCT, and DSN8FRCT contain the respective PPT, PCT, and RCT entries required for the phase 5 applications. These members help you perform the definition step. Make sure that the subsystem ID (SUBID) in the RCT entry matches your DB2 subsystem ID.

If DSNTEJ5C runs successfully, it produces the return codes that are shown in the following table.

Table 128. DSNTEJ5C return codes

Step	PROCSTEP	Return code
MAPG	ASSEM	0000
MAPD	ASSEM	0000
DSNH		0000 or 0004
BIND		0000
MAPGP	ASSEM	0000
MAPGL		0000
MAPDP	ASSEM	0000
MAPDL		0000

If DSNTEJ5C fails or abends, rerun the job from the last successful step. To receive more prepare-time detail from DSNTEJ5C, change the parameters TERM(LEAVE) and PRINT(LEAVE) to TERM(TERM) and PRINT(TERM).

If DSNTEJ5P runs successfully, it produces the return codes that are shown in the following table.

Table 129. DSNTEJ5P return codes

Step	PROCSTEP	Return code
PH05PS01	ASSEM	0000
PH05PS02	ASSEM	0000
PH05PS03	ASSEM	0000
PH05PS04	ASSEM	0000
PH05PS05	ASSEM	0000
PH05PS06		0004
PH05PS07		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS08		0004
PH05PS09		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS10		0004
PH05PS11		
	PPLI	0004
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS12		0004
PH05PS13		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS14	ASSEM	0000
PH05PS15	ASSEM	0000
PH05PS16		0004
PH05PS17		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004

Table 129. DSNTEJ5P return codes (continued)

Step	PROCSTEP	Return code
PH05PS18		0004
PH05PS19		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS20		0004
PH05PS21		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0004
PH05PS22		0000 or 0004
PH05PS23		0000
PH05PS24	ASSEM	0000
PH05PH25		0000
PH05PS26	ASSEM	0000
PH05PH27		0000
PH05PS28	ASSEM	0000
PH05PH29		0000
PH05PS30	ASSEM	0000
PH05PH31		0000
PH05PS32	ASSEM	0000
PH05PH33		0000
PH05PS34	ASSEM	0000
PH05PH35		0000
PH05PS36	ASSEM	0000
PH05PH37		0000

If DSNTEJ5P fails or abends, rerun the job from the last successful step. You might find it convenient to break up DSNTEJ5P and run only the unsuccessful steps.

Related concepts:

"COBOL options to use with the installation verification procedures" on page 749

Related reference:

DSNH (TSO CLIST) (DB2 Commands)

Starting an application in a CICS environment

After logging on to CICS, you can start an organization application or project application in the CICS environment.

Procedure

To start an application in a CICS environment:

Enter a CICS transaction code.
The CICS transaction codes are:

- D8PP, which starts the PL/I project version
- D8PS, which starts the PL/I organization version
- D8CS, which starts the COBOL organization version.

When these transaction codes are entered, the panels that are shown in the following two figures are displayed.

```
              ACTION SELECTION
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
SELECT AN ACTION FROM FOLLOWING LIST

   A     ADD (INSERT)
   D     DISPLAY (SHOW)
   E     ERASE (REMOVE)
   U     UPDATE (CHANGE)
```

Figure 84. Initial panel for the organization application in CICS

```
              ACTION SELECTION
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
SELECT AN ACTION FROM FOLLOWING LIST

   A     ADD (INSERT)
   D     DISPLAY (SHOW)
   E     ERASE (REMOVE)
   U     UPDATE (CHANGE)
```

Figure 85. Initial panel for the project application in CICS

Related concepts:
"The sample application panels" on page 804

Using the phone application in CICS

When you use CICS, information is interactively processed.

Procedure

To begin using the phone application in CICS:

Clear the screen and type in the transaction code:

```
D8PT
```

You can change the transaction codes when you install DB2. Check with your system administrator to find out if they have been changed from those shown.

Using CICS storage-handling facilities

You can use the CICS storage-handling facilities when you run the CICS sample applications.

Procedure

To use the CICS storage-handling facilities when running the CICS sample applications:

1. Change your DSNTIAR calls to DSNTIAC calls in DSN8MC*xx* and DSN8MP*xx*. The calls should look like this:

   ```
   CALL DSNTIAC(EIB,COMMAREA,SQLCA,MSG,LRECL)
   ```

2. Define DSNTIAC and DSNTIA1 in the CSD.
3. Rerun job DSNTEJ5C or job DSNTEJ5P.

Phase 6: Accessing data at a remote site

You can use phase 6 of the DB2 installation or migration verification process to verify that the features of DRDA access are working correctly, to prepare and call sample stored procedures, and to prepare and execute sample callers of certain DB2-supplied stored procedures.

During this optional phase, you test accessing data at a remote site by using the DRDA access application (job DSNTEJ6 in conjunction with job DSNTEJ3C).

You can use the following jobs to test creating and calling stored procedures at either the local site or a remote site.

DSNTEJ6S and DSNTEJ6P
: Tests the stored procedure without result set sample

DSNTEJ6T and DSNTEJ6D
: Tests the stored procedure with result set sample

DSNTEJ61 and DSNTEJ62
: Tests the stored procedure for IMS Open Database Access

DSNTEJ63 and DSNTEJ64
: Tests the external SQL procedure batch sample

DSNTEJ65
: Tests the external SQL procedures processor invocation sample

DSNTEJ66
: Tests the stored procedure that prepares a sample native SQL procedure and caller

You can use the following jobs to test calling DB2-supplied stored procedures at the local site:

DSNTEJ6U, DSNTEJ6R, and DSNTEJ6V
: Tests the stored procedure for invoking utilities

DSNTEJ6W
: Tests the stored procedure for invoking WLM_REFRESH

DSNTEJ6Z
: Tests the stored procedure for formatting the IFCID 106 trace record to obtain current DB2 configuration settings

ISPF panel requirements

The installation CLIST prepares samples DSNTEJ6, DSNTEJ6S, DSNTEJ6P, DSNTEJ6T, DSNTEJ6D, DSNTEJ6Z, DSNTEJ61, DSNTEJ62, DSNTEJ63, DSNTEJ64, DSNTEJ65 and DSNTEJ66 only if you specify COMMAND or AUTO in the DDF STARTUP OPTION field of panel DSNTIPR. If you specify NO, the installation CLIST does not prepare these samples. In this case, you should not try to run these samples.

The installation CLIST prepares the stored procedures samples only if you specify a sample WLM environment name in the WLM ENVIRONMENT field on panel DSNTIPX. If you replace the default value with blanks in the WLM ENVIRONMENT field, you cannot start the sample WLM-established stored procedures address space until you update the subsystem parameter.

The installation CLIST tailors the phase 6 sample jobs according to the information that you specify in the REMOTE LOCATION field of panel DSNTIPY. The guidelines for this field are:

- If the field is blank, the installation CLIST only customizes phase 6 sample jobs DSNTEJ63, DSNTEJ64, DSNTEJ65, DSNTEJ66, DSNTEJ6R, DSNTEJ6U, DSNTEJ6V, DSNTEJ6W, and DSNTEJ6Z. Jobs DSNTEJ63, DSNTEJ64, DSNTEJ65, and DSNTEJ66 use the local location name when the REMOTE LOCATION field is blank. Jobs DSNTEJ6R, DSNTEJ6U, DSNTEJ6V, DSNTEJ6W, and DSNTEJ6Z do not use a remote location name.
- If the value in the field is the same as the location name for the DB2 subsystem that you are installing (DB2 LOCATION NAME field of panel DSNTIPR), the stored procedures samples are prepared and customized for local use. However, the DRDA access sample is not prepared, including job DSNTEJ6 and the DRDA access component of job DSNTEJ3C.
- If the value in the field is different from the DB2 LOCATION NAME field, the installation CLIST prepares the phase 6 samples assuming that the remote location is the server and that the local system is the client.

The following table summarizes the ISPF panel values that are required for the installation CLIST to prepare the phase 6 sample jobs.

Table 130. IVP phase 6 job requirements. The installation CLIST only customizes the phase 6 sample jobs if certain fields on the ISPF panels meet requirements.

Job name	Requires COMMAND or AUTO in the DDF STARTUP OPTION field of panel DSNTIPR?	Requires a name in the REMOTE LOCATION field of panel DSNTIPY?	Requires a name in the WLM ENVIRONMENT field of panel DSNTIPX?
DSNTEJ6	Yes	Yes	No
DSNTEJ6S	Yes	Yes	Yes
DSNTEJ6P	Yes	Yes	Yes
DSNTEJ6T	Yes	Yes	Yes
DSNTEJ6D	Yes	Yes	Yes
DSNTEJ6R	No	No	Yes
DSNTEJ6U	No	No	No
DSNTEJ6V	No	No	No
DSNTEJ6W	No	No	Yes

Table 130. IVP phase 6 job requirements (continued). The installation CLIST only customizes the phase 6 sample jobs if certain fields on the ISPF panels meet requirements.

Job name	Requires COMMAND or AUTO in the DDF STARTUP OPTION field of panel DSNTIPR?	Requires a name in the REMOTE LOCATION field of panel DSNTIPY?	Requires a name in the WLM ENVIRONMENT field of panel DSNTIPX?
DSNTEJ6Z	Yes	No	Yes
DSNTEJ61	Yes	Yes	Yes
DSNTEJ62	Yes	Yes	Yes
DSNTEJ63	Yes	No	Yes
DSNTEJ64	Yes	No	Yes
DSNTEJ65	Yes	No	Yes
DSNTEJ66	Yes	No	Yes

Testing multiple subsystems

If you are installing and testing two DB2 subsystems concurrently, you must designate one as the server and the other as the client. If you change these designations during your testing, your results will be unpredictable. Verify that your VTAM APPL statement has the parameter SYNCLVL=SYNCPT defined. This setting allows updates at several locations.

DRDA access sample

The distributed sample application that uses DRDA access is executed as part of Phase 6 of the installation or migration verification process.

The application is prepared in Phase 3 as part of DSNTEJ3C. Before this application can be run correctly as a DRDA access sample, you must run job DSNTEJ6 at both the local and remote sites to tailor the DEPT sample table for use in a distributed environment.

To set up your samples testing for concurrent installations at two DB2 locations, follow these guidelines:

- Designate one location as the requester (the client) and the other location as the server.
- Run the client version of DSNTEJ6 at the client site only; do **not** run the client version of DSNTEJ6 at the server.
- Edit the server version of DSNTEJ6 at the remote server site; do **not** run the server version of DSNTEJ6 at the client.
- GUPI Locate the following text in the server version of DSNTEJ6 within step PH06S01:

  ```
  UPDATE DEPT SET LOCATION = (your remote location name) WHERE DEPTNO = 'F22';
  UPDATE DEPT SET LOCATION = (your location name) WHERE LOCATION = '  ';
  ```

  This text should be replaced with:

  ```
  UPDATE DEPT SET LOCATION = (your location name) WHERE DEPTNO = 'F22';
  UPDATE DEPT SET LOCATION = (your remote location name) WHERE LOCATION = '  ';
  ```

  GUPI

Job DSNTEJ6

Job DSNTEJ6 consists of a single step that updates the location column in the sample DEPT table at the client and server DB2 subsystems.

The employees in department F22 are recorded as working at the server location that was entered in the REMOTE LOCATION field of panel DSNTIPY. All other employees are recorded as working at the client location.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY and it does not match the name specified in the DB2 LOCATION NAME field of panel DSNTIPR

Before you run DSNTEJ6, ensure that the server DB2 subsystem's DSNTEJ1 job has been run on the server DB2 subsystem. Run the client DB2 subsystem's DSNTEJ6 job only on the server DB2 subsystem.

If DSNTEJ6 runs successfully, it produces the return code that is shown in the table below.

Table 131. DSNTEJ6 return codes

Step	PROCSTEP	Return code
PH06S01		0000

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 6"

Related reference:

"Distributed data facility panel 1: DSNTIPR" on page 442

"Job editing panel: DSNTIPY" on page 488

Starting an application in an ISPF/TSO environment in phase 6

You must have access to ISPF load module libraries in order to run the ISPF/CAF sample application.

Procedure

To start the sample application:

1. Enter a CALL command from option 6 of the ISPF primary option menu.
2. To start the COBOL sample version of the connection manager, enter the following:

   ```
   CALL 'prefix.RUNLIB.LOAD(DSN8SCM)'
   ```

   After you enter this command, DB2 displays the sample applications panel. Choosing option 3 on the sample applications panel during Phase 6 invokes the COBOL organization application, which uses DRDA access for distributed data.

Related tasks:

"Working with the distributed organization application scenario" on page 818

"Starting an application in an ISPF/TSO environment in phase 6" on page 781

"Make DB2I panels, messages, and load modules available to ISPF and TSO" on page 586

Stored procedure samples

DB2 provides a set of stored procedure sample applications. These sample applications demonstrate different ways that a stored procedure can be used by a client to issue DB2 commands to a DB2 server.

Most of these applications prepare and run two programs. One program prepares a stored procedure, and one program executes a client program that calls the stored procedure and returns some response.

DB2 provides the following stored procedure samples:

- A sample stored procedure without a result set
- A sample stored procedure with a result set
- Two samples that invoke the utilities stored procedure DSNUTILS
- A sample that invokes the utilities stored procedure DSNUTILU
- A sample that invokes the stored procedure WLM_REFRESH, which refreshes the WLM environment
- A sample that invokes the stored procedure DSNWZP, which lists the settings of subsystem parameters
- A sample stored procedure that uses IMS Open Database Access (ODBA) support
- Two samples of external SQL procedures

All of these samples are part of the installation verification samples, which are located in *prefix*.NEW.SDSNSAMP. The installation CLIST tailored and loaded these jobs during your installation or migration to new-function mode.

To edit any of these sample jobs, specify the appropriate fields on the installation panels. Using the installation panels is the only method that you can use to edit these jobs.

Before you run the sample stored procedures, ensure that you have completed one of the following tasks:

- Configure DB2 for running stored procedures and user-defined functions during installation
- Configure DB2 for running stored procedures and user-defined functions during migration

Related concepts:

"Installation verification phases and programs" on page 744

Stored procedure sample without result set

DB2 provides a sample stored procedure that uses the instrumentation facility interface (IFI) to issue DB2 commands. The sample stored procedure also returns the responses to the calling program in a parameter.

This sample application consists of two jobs: DSNTEJ6S and DSNTEJ6P. These jobs are part of the installation verification samples, which are located in

prefix.NEW.SDSNSAMP. The installation CLIST tailored and loaded these jobs during your installation or migration to new-function mode.

Requirements: You must satisfy the following requirements when running these jobs:

- Run job DSNTEJ6S before job DSNTEJ6P.
- Ensure that the PL/I product is installed on your client and server systems.
- Ensure that Language Environment is installed.

This application prepares and runs the following programs:

- A stored procedure that uses the IFI to issue DB2 commands.
- A client program that receives DB2 command text and calls the stored procedure to issue the commands. When the program receives the responses from the stored procedure in a parameter that is passed back, it prints the results.

For concurrent installations at two DB2 locations, perform the following actions:

- Run the server version of DSNTEJ6S on the server system only; do **not** run the client version of DSNTEJ6S on the server.
- Run the client version of DSNTEJ6P on the client system only; do **not** run the server version of DSNTEJ6P on the client.

Job DSNTEJ6S:

Job DSNTEJ6S compiles and link-edits the sample stored procedure DSN8EP2. DSNTEJ6S also updates the SYSIBM.SYSROUTINES catalog table with information about the stored procedure.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

If you have SQL statements in your stored procedure, you must remove the comment character in the JCL from the step that binds the stored procedure package.

You must run job DSNTEJ6S on the server DB2 subsystem.

If DSNTEJ6S runs successfully, it produces the return codes that are shown in the table below.

Table 132. DSNTEJ6S return codes

Step	PROCSTEP	Return code
PH06SS01		0000
PH06SS02		0000

Table 132. DSNTEJ6S return codes (continued)

Step	PROCSTEP	Return code
PH06SS03		
	PPLI	0000
	PC	0004
	PLI	0004
	PLKED	0004
	LKED	0000

If SQLCODE -592 is received during the execution of DSNTEJ6T, ensure that SPA is active.

Job DSNTEJ6P:

Job DSNTEJ6P compiles, link-edits, binds, and runs a sample program, DSN8EP1, which calls the sample stored procedure that is created by job DSNTIJ6S.

The sample caller first connects to the location that is specified in the REMOTE LOCATION field of panel DSNTIPY. If the remote location name is the same as the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the local DB2 subsystem is used as both the client and server.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

Before you run DSNTEJ6P, ensure that DSNTEJ6S has been run on the server DB2 subsystem to create the sample stored procedure.

If DSNTEJ6P runs successfully, it produces the following return codes.

Table 133. DSNTEJ6P return codes

Step	PROCSTEP	Return code
PH06PS01		
	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0000
PH06PS02		0004
PH06PS03		0000

Output from a successful execution of DSNTEJ6P lists each DB2 command that was executed, followed by the messages that were generated by the DB2 command processor.

You can compare the output from this job with the sample output for DSNTEJ6P, which is in member DSN8TJ6P of your *prefix*.SDSNIVPD data set.

Stored procedure sample with result set

DB2 provides a sample stored procedure that uses the instrumentation facility interface (IFI) to issue DB2 commands. The sample stored procedure also returns the responses to the calling program in a result set.

This sample application consists of two jobs: DSNTEJ6T and DSNTEJ6D. These jobs are part of the installation verification samples, which are located in *prefix*.NEW.SDSNSAMP. The installation CLIST tailored and loaded these jobs during your installation or migration to new-function mode.

Requirements: You must satisfy the following requirements when running these jobs:

- Run job DSNTEJ6T before job DSNTEJ6D.
- Ensure that C for z/OS is installed.
- Ensure that Language Environment is installed.

This application prepares and runs the following programs:

- A stored procedure that uses the IFI to issue DB2 commands.
- A client program that receives DB2 command text and calls the stored procedure to issue the commands. When the program receives the responses from the stored procedure in a result set, it prints the results.

For concurrent installations at two DB2 locations, perform the following actions:

- Run the server version of job DSNTEJ6T on the server side only; do **not** run it on the client side.
- Run the client version of job DSNTEJ6D on the client side only; do **not** run it on the server side.

Job DSNTEJ6T:

Job DSNTEJ6T registers, prepares, and binds the sample stored procedure DSN8ED2 on the server. DSNTEJ6T also defines a created temporary table to receive the IFI output that is returned as a result set.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

You must run job DSNTEJ6T on the server DB2 subsystem.

If DSNTEJ6T runs successfully, it produces the return codes that are shown in the table below.

Table 134. DSNTEJ6T return codes

Step	PROCSTEP	Return code
PH06TS01		0000
PH06TS02		0000

Table 134. DSNTEJ6T return codes (continued)

Step	PROCSTEP	Return code
PH06TS03		
	PC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH06TS04		0000 or 0004

If SQLCODE -592 is received during the execution of DSNTEJ6T, ensure that SPA is active.

Job DSNTEJ6D:

Job DSNTEJ6D compiles, link-edits, binds, and runs sample program DSN8ED1, which calls the sample stored procedure that is created by job DSNTEJ6T.

The sample caller first connects to the location that is specified in the REMOTE LOCATION field of panel DSNTIPY. If the remote location name is the same as the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the local DB2 subsystem is used as both the client and server.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

Before you run DSNTEJ6D, ensure that DSNTEJ6T has been run on the server DB2 subsystem to create the sample stored procedure.

If DSNTEJ6D runs successfully, it produces the return codes that are shown in the table below.

Table 135. DSNTEJ6D return codes

Step	PROCSTEP	Return code
PH06DS01		
	PC	0000
	C	0000 or 0004
	PLKED	0000 or 0004
	LKED	0000
PH06DS02		0004
PH06DS03		0000

Output from a successful execution of DSNTEJ6D lists each DB2 command that was executed, followed by the messages that were generated by the DB2 command processor.

You can compare the output from this job with the sample output for DSNTEJ6D, which is in member DSN8TJ6D of your *prefix*.SDSNIVPD data set.

Sample callers of utilities stored procedures

The DSNUTILS stored procedure enables execution of DB2 utilities from aDB2 application program by using the SQL CALL statement.

When called, DSNUTILS dynamically allocates the specified data sets, creates the utility input stream (SYSIN), invokes DB2 utilities (DSNUTILB), deletes all rows currently in the created temporary table (SYSIBM.SYSPRINT), captures the utility output stream (SYSPRINT), and puts this output into the created temporary table (SYSIBM.SYSPRINT). The DSNUTILU stored procedure is very similar to DSNUTILS, but accepts parameters in Unicode.

The DSNUTILS and DSNUTILU stored procedures must run as a WLM-managed stored procedure.

Job DSNTEJ6U:

Job DSNTEJ6U compiles, link-edits, binds, and runs sample PL/I program DSN8EPU, which invokes the DSNUTILS stored procedure to execute a utility.

Job DSNTEJ6U uses the local DB2 subsystem as both the client and the server subsystem.

The installation CLIST always edits this job.

Before running job DSNTEJ6U, ensure that DSNUTILS has been created as part of the installation or migration process.

Recommendation: Use the DSNWLM_UTILS core WLM environment for DSNUTILS.

If DSNTEJ6U completes successfully, it produces the return codes that are shown in the table below.

Table 136. DSNTEJ6U return codes

Step	PROCSTEP	Return code
PH06US01	PPLI	0000
	PC	0000
	PLI	0000
	PLKED	0004
	LKED	0000
PH06US02		0000
PH06US03		0000

Output from a successful execution of DSNTEJ6U lists the parameters specified followed by the messages generated by the DB2 DIAGNOSE DISPLAY MEPL utility.

You can compare the output from this job with the sample output for DSNTEJ6U found in member DSN8TJ6U in your *prefix*.SDSNIVPD data set.

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Related reference:

Deprecated: DSNUTILS stored procedure (DB2 Utilities)

Job DSNTEJ6R:

Job DSNTEJ6R compiles, link-edits, binds, and runs sample C program DSN8ED8, which invokes the DSNUTILU stored procedure to execute a utility.

For convenience on TSO, the utility control statement that is input to DSN8ED8 is encoded in EBCDIC and converted to Unicode before being passed to DSNUTILU.

Job DSNTEJ6R uses the local DB2 subsystem as both the client and the server subsystem.

The installation CLIST edits this job only if you specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

Before running job DSNTEJ6R, ensure that DSNUTILU has been created as part of the installation or migration process.

Recommendation: Use the DSNWLM_UTILS core WLM environment for DSNUTILU.

If DSNTEJ6R completes successfully, it produces the return codes that are shown in the table below.

Table 137. DSNTEJ6R return codes

Step	PROCSTEP	Return code
PH06RS00		0000
PH06RS01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH06RS02		0000 or 0004
PH06RS03		0000

Output from a successful execution of DSNTEJ6R lists the parameters specified followed by the messages generated by the DB2 DIAGNOSE DISPLAY MEPL utility.

You can compare the output from this job with the sample output for DSNTEJ6U found in member DSN8TJ6U in your *prefix*.SDSNIVPD data set.

Related concepts:

Utility control statements (DB2 Utilities)

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Related reference:

DSNUTILU stored procedure (DB2 Utilities)

Job DSNTEJ6V:

Job DSNTEJ6V compiles, link-edits, binds, and runs sample C++ program DSN8EE1, which invokes the DSNUTILS stored procedure to execute a utility.

Job DSNTEJ6V uses the local DB2 subsystem as both the client and the server subsystem.

The installation CLIST edits this job only if the following conditions are met:

- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You indicated on panel DSNTIPU that an IBM C++ compiler is installed.

Before running job DSNTEJ6V, ensure that DSNUTILS has been created as part of the installation or migration process.

Recommendation: Use the DSNWLM_UTILS core WLM environment for DSNUTILS.

If DSNTEJ6V completes successfully, it produces the return codes that are shown in the table below.

Table 138. DSNTEJ6V return codes

Step	PROCSTEP	Return code
PH06VS01	PC	0004
	CP	0000
	PLKED	0004
	LKED	0004
PH06VS02	PC1	0000
	CP1	0000
	PC2	0004
	CP2	0000
	PLKED	0004
	LKED	0000
PH06VS03		0000 or 0004
PH06VS04		0000

A successful execution of DSNTEJ6V unloads rows and columns from the PROJ sample table.

You can compare the output from this job with the sample output for DSNTEJ6V found in member DSN8TJ6V in your *prefix*.SDSNIVPD data set.

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Related reference:

Deprecated: DSNUTILS stored procedure (DB2 Utilities)

Job DSNTEJ6W

Job DSNTEJ6W is a JCL job that creates and initializes a sample SAF resource profile. DSNTEJ6W also prepares, binds, and executes DSN8ED6.

DSN8ED6 is a C language caller of the WLM_REFRESH stored procedure. It accepts the WLM environment name and, optionally, the subsystem ID and an SQLID to be in effect when WLM_REFRESH is invoked.

If a name is specified in the WLM ENVIRONMENT field on installation panel DSNTIPX, the installation CLIST customizes DSNTEJ6W to run in and recycle that WLM environment.

DSNTEJ6W also includes commented-out job steps that show how to prepare the DSNTWR external module from the DSNTWRS sample source code. Do not uncomment and run these jobs unless you maintain a customized copy of DSNTWRS. In general, use the DB2-supplied DSNTWR module in the *prefix*.SDSNLOAD library and the package that is bound by installation job DSNTIJRT.

Job DSNTEJ6W uses the local DB2 subsystem as both the client and the server subsystem.

Before running job DSNTEJ6W, ensure that WLM_REFRESH has been created as part of the installation or migration process. If you are not authorized to create special resource profiles, have your system security administrator perform the first step of this job. The SQLID that is used to run DSNTEJ6W needs READ access to this profile.

If DSNTEJ6W completes successfully, it produces the return codes that are shown in the table below.

Table 139. DSNTEJ6W return codes

Step	PROCSTEP	Return code
PH06WS01[1]	PC	0000
	ASM	0000
	LKED	0000
PH06WS02[1]		0000
PH06WS03	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH06WS04		0000
PH06WS05		0000 or 0004

Table 139. DSNTEJ6W return codes (continued)

Step	PROCSTEP	Return code
Note:		
1. Return codes for PH06WS01 and PH06WS02 are generated only if the job steps are uncommented.		

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Related reference:

WLM_REFRESH stored procedure (DB2 Application programming and SQL)

Job DSNTEJ6Z

Job DSNTEJ6Z generates a report of current subsystem parameter settings. This report is generated by DSN8ED7, a C-language caller of stored procedure DSNWZP. You must have TRACE and MONITOR1 privileges to run DSNWZP.

Job DSNTEJ6Z uses the local DB2 subsystem as both the client and the server subsystem.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

Before running job DSNTEJ6Z, ensure that DSNWZP has been created as part of the installation or migration process.

If DSNTEJ6Z completes successfully, it produces the return codes that are shown in the table below.

Table 140. DSNTEJ6Z return codes

Step	PROCSTEP	Return code
PH06ZS01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH06ZS02		0000 or 0004
PH06ZS03		0000

A successful execution of DSNTEJ6Z provides a report as shown in the figure below.

```
DSN8ED7: Sample DB2 Configuration Setting Report Generator

Macro    Parameter  Current     Description/         Install   Fld
Name     Name       Setting     Install Field Name   Panel ID  No.
_____________________________________________________________________
DSN6SYSP AUDITST    0000000000  AUDIT TRACE          DSNTIPN      1
DSN6SYSP CONDBAT    0000010000  MAX REMOTE CONNECTED DSNTIPE      4
DSN6SYSP CTHREAD    00200       MAX USERS            DSNTIPE      2
DSN6SYSP PCLOSEN    00010       RO SWITCH CHKPTS     DSNTIPL1     8
...
```

Figure 86. DSNTEJ6Z report format

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Sample ODBA stored procedure

IMS Open Database Access (ODBA) support enables a DB2 stored procedure to directly connect to an IMS DBCTL system and issue DL/I calls to access IMS databases.

This sample ODBA stored procedure application consists of two jobs: DSNTEJ61 and DSNTEJ62. These jobs are part of the installation verification samples, which are located in *prefix*.NEW.SDSNSAMP. The installation CLIST tailored and loaded these jobs during your installation or migration to new-function mode.

Requirements: You must satisfy the following requirements when running these jobs:

- Run job DSNTEJ61 before DSNTEJ62.
- Ensure that Enterprise COBOL for z/OS is installed.
- Ensure that Language Environment is installed.
- Start a WLM-established stored procedure address space to run DSNTEJ61 and DSNTEJ62. You need to update the startup procedure for the WLM-established stored procedure address space to add the ODBA data set names to the STEPLIB and DFSRESLB concatenations. An example of a data set name for ODBA is IMSVS.RESLIB.

Related tasks:

Assigning stored procedures and functions to WLM application environments (DB2 Performance)

Writing an external procedure to access IMS databases (DB2 Application programming and SQL)

Job DSNTEJ61:

Job DSNTEJ61 prepares sample stored procedure DSN8EC1, which uses ODBA. DSN8EC1 can add, update, delete, and display telephone directory records from the IMS sample database, DFSIVD1. DSN8EC1 shows how the AERTDLI API is used to issue IMS DL/I calls.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

Before running job DSNTEJ61, read the dependencies information of the job prolog to verify that the server site is correctly configured and has a connection to an IMS subsystem.

If DSNTEJ61 runs successfully, it produces the return codes that are shown in the table below.

Table 141. DSNTEJ61 return codes

Step	PROCSTEP	Return code
PH061S01		0000
PH061S02		0000
PH061S03	PC	0004
	COB	0000
	PLKED	0004
	LKED	0000

Related concepts:
"Job DSNTEJ62"

Job DSNTEJ62:

Job DSNTEJ62 prepares and invokes the sample client program DSN8EC2, which calls stored procedure DSN8EC1 multiple times to add, delete, and display telephone directory records. DSN8EC1 is created by job DSNTEJ61.

The sample caller first connects to the location that is specified in the REMOTE LOCATION field of panel DSNTIPY. If the remote location name is the same as the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the local DB2 subsystem is used as both the client and server.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.
- You specified a name in the REMOTE LOCATION field of panel DSNTIPY. The name can match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR.

Before running DSNTEJ62, complete the manual editing that is described in the dependencies information in the job prolog.

If DSNTEJ62 runs successfully, it produces the return codes that are shown in the table below.

Table 142. DSNTEJ62 return codes

Step	PROCSTEP	Return code
PH062S01	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH062S02		0000 or 0004
PH062S03		0000

Related concepts:

"Job DSNTEJ61" on page 792

Sample SQL procedures

DB2 for z/OS provides three applications for external SQL procedures and one application for native SQL procedures.

The applications for external SQL procedures are:

- DSNTEJ63
- DSNTEJ64
- DSNTEJ65

  C and Language Environment are required for jobs DSNTEJ63, DSNTEJ64, and DSNTEJ65.

The application for native SQL procedures is DSNTEJ66.

Related tasks:

Assigning stored procedures and functions to WLM application environments (DB2 Performance)

Job DSNTEJ63:

Job DSNTEJ63 prepares the sample external SQL procedure, DSN8ES1, which accepts a department number and returns salary and bonus data.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

If a remote location was specified in the REMOTE LOCATION field of panel DSNTIPY, job DSNTEJ63 should be run on the remote server subsystem.

Before running DSNTEJ63, you should manually edit the job as described in the dependencies information in the job prolog.

If DSNTEJ63 runs successfully, it produces the return codes that are shown in the following table.

Table 143. DSNTEJ63 return codes

Step	PROCSTEP	Return code
PH063S01		0000
PH063S02	PC	0004
	PCC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH063S03		0000 or 0004

Job DSNTEJ64:

Job DSNTEJ64 prepares and executes DSN8ED3, which is a sample routine that calls the sample external SQL procedure, DSN8ES1. DSN8ES1 was created by job DSNTIJ63.

The sample caller first connects to the location that is specified in the REMOTE LOCATION field of panel DSNTIPY. If the remote location name is the same as the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the local DB2 subsystem is used as both the client and server.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

Before running job DSNTEJ64, ensure that job DSNTEJ6S was run on the server subsystem. Also run job DSNTEJ63 before running job DSNTEJ64. Before running DSNTEJ64, you should manually edit the job as described in the dependencies information in the job prolog.

If DSNTEJ64 runs successfully, it produces the return codes that are shown in the following table.

Table 144. DSNTEJ64 return codes

Step	PROCSTEP	Return code
PH064S01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH064S02		0000 or 0004
PH064S03		0000

You can compare the output from this job to the sample output for DSNTEJ64, which is found in member DSN8TJ64 in data set named *prefix*.SDSNIVPD.

Job DSNTEJ65:

Job DSNTEJ65 prepares and runs a sample application that calls the SYSPROC.DSNTPSMP stored procedure processor to create a sample external SQL procedure. The sample external SQL procedure is created at the location that is specified in the REMOTE LOCATION field of panel DSNTIPY.

Job DSNTEJ65 then prepares and runs a sample application that calls the external SQL procedure. The sample caller first connects to the location that is specified in the REMOTE LOCATION field of panel DSNTIPY.

If a remote location is specified in the REMOTE LOCATION field name on installation panel DSNTIPY, ensure that both DB2 REXX language support and the DSNTPSMP stored procedure are available at that location. If the remote location name is the same as the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the local DB2 subsystem is used as both the client and server.

The major components of job DSNTEJ65 are as follows:

- DSNWLM_REXX is the WLM core environment for DSNTPSMP.
- DSN8ED4 is a sample C program that calls the DB2 external SQL procedures processor.
- DSN8ES2 is a sample external SQL procedure that calculates employee bonuses.
- DSN8ED5 is a sample C program that calls the external SQL procedure DSN8ES2.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

Before running job DSNTEJ65, ensure that DSNTPSMP has been created as part of the installation or migration process.

Recommendation: DSNTPSMP requires that the default EBCDIC CCSID that is used by DB2 also be compatible with the C compiler. Do not run DSNTEJ65 if the default EBCDIC CCSID for DB2 is not compatible with the C compiler. Using an incompatible CCSID results in compile-time errors. Examples of incompatible CCSIDs include 290, 930, 1026, and 1155.

If DSNTEJ65 runs successfully, it produces the return codes that are shown in the following table.

Table 145. DSNTEJ65 return codes

Step	PROCSTEP	Return code
PH065S01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000

Table 145. DSNTEJ65 return codes (continued)

Step	PROCSTEP	Return code
PH065S02		0000 or 0004
PH065S03		0000 or 0004
PH065S04		0000 or 0004
PH065S05		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH065S06		0000 or 0004
PH065S07		0000

You can compare the output from this job to the sample output for DSNTEJ65, which is found in member DSN8TJ65 in the data set named *prefix*.SDSNIVPD.

Related tasks:

"Installation step 20: Set up DB2-supplied routines" on page 547

"Migration step 26: Set up DB2-supplied routines" on page 622

Job DSNTEJ66:

Job DSNTEJ66 creates a sample native SQL procedure. It also prepares and executes a sample caller of that SQL procedure on the local DB2 subsystem. It then modifies the SQL procedure slightly, activates the change on the local subsystem, and executes the sample caller to call the updated SQL procedure.

Specifically, job DSNTEJ66 completes the following actions:

- Creates a sample native SQL procedure called DSN8.DSN8ES3 that generates and returns a CREATE PROCEDURE statement for a given stored procedure.
- Prepares and executes a sample caller of DSN8ES3 called DSN8ED9.
- Illustrates how to use ALTER PROCEDURE... ADD VERSION to create a version V2 of DSN8ES3 that does the same thing as the original version but also adds a terminating semicolon at the end of the generated CREATE PROCEDURE statement.
- Illustrates how to use ALTER ACTIVATE to make version V2 the active version of DSN8ES3.
- If you specify a remote DB2 subsystem in the REMOTE LOCATION field and it does not match the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, additional job steps do the following:
  - Deploys the sample native SQL procedure on the remote DB2 subsystem.
  - Binds the sample caller on the remote DB2 subsystem and execute it to call the native SQL procedure at that site.

  If the REMOTE LOCATION field is blank or matches the name that is specified in the DB2 LOCATION NAME field of panel DSNTIPR, the deploy-related steps are removed from DSNTEJ66 by the installation CLIST.

As part of the setup for deploying DSN8ES3 on a remote server, the DSNTEP2 application needs to be able to connect to the remote site.

The installation CLIST edits this job only if the following conditions are met:

- You specified AUTO or COMMAND in the DDF STARTUP OPTION field of panel DSNTIPR.
- You specified a name in the WLM ENVIRONMENT field of panel DSNTIPX.

Before running DSNTEJ66, perform the manual editing described in the dependencies information in the job prolog. You must also manually tailor DSN8WLMP, the sample WLM startup procedure for DSNTPSMP. If DSNTEJ66 runs successfully, it produces the return codes that are shown in the table below.

Table 146. DSNTEJ66 return codes

Step	PROCSTEP	Return code
PH066S01		0000
PH066S02		0000
PH066S03		0000
PH066S04		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH066S05		0000 or 0004
PH066S06		0000
PH066S07		0000
PH066S08		0000
PH066S09		0000
PH066S10		0000
PH066S11		0000
PH066S12		0000 or 0004
PH066S13		0000
PH066S14		0000
PH066S15		0000 or 0004
PH066S16		0000

You can compare the output from this job to the sample output for DSNTEJ66, which is found in member DSN8TJ66 in the data set named *prefix*.SDSNIVPD.

Phase 7: Accessing LOB data

Phase 7 of the installation or migration verification process is an optional phase that demonstrates how to set up and use a DB2 LOB application. This phase creates an extension to the Employee sample database to manage employee resumes and photo images.

You run these jobs in this phase:

- DSNTEJ7: Creates and loads the employee resume and photo table.

- DSNTEJ71, DSNTEJ76: Uses LOB locators to populate the photo images, then validates that the resume and photo data is stored correctly.
- DSNTEJ73, DSNTEJ77: Prepares an ISPF application for viewing employee resume data.
- DSNTEJ75, DSNTEJ78: Prepares a GDDM application for viewing employee photo images.

Jobs DSNTEJ75 and DSNTEJ78 are not tailored by the installation CLIST unless you specify non-blank values for GDDM MACLIB and GDDM LOAD MODULES on panel DSNTIPW.

After you run these jobs, you can use ISPF and GDDM to view the sample employee resume and photo data.

Job DSNTEJ7

Job DSNTEJ7 demonstrates how to create a LOB table with all the accompanying LOB table spaces, auxiliary tables, and indexes. DSNTEJ7 also demonstrates how to use the DB2 LOAD utility to load a LOB table.

Job DSNTEJ7 consists of the following steps:

Step	Function
1	Drops sample LOB objects
2	Creates sample Employee Resume and Photo LOB table
3	Creates aliases for the table, then grants access to it
4	Uses the DB2 LOAD utility to populate the table
5	Generates run-time statistics on the table

If DSNTEJ7 runs successfully, it produces the return codes that are shown in the table below.

Table 147. DSNTEJ7 return codes

JOBSTEP	PROCSTEP	Return Code
PH07S01		0000
PH07S02		0000
PH07S03		0000 or 0004
PH07S04	DSNUPROC	0000
PH07S05	DSNUPROC	0000
PH07S06		0000
PH07S07		0000

Job DSNTEJ71 (optional)

Job DSNTEJ71 compiles, link-edits, and binds two sample applications that manipulate LOB data.

The DSN8DLPL sample application demonstrates how to use LOB locators to populate a LOB column. The DSN8DLTC sample application validates the contents of the LOB table, verifying that it was populated correctly.

In previous releases, you had to run job DSNTEJ71 to populate LOB columns that were greater than 32 KB. In Version 10, the LOAD utility populates all columns of the sample LOB table, so job DSNTEJ71 is optional.

If DSNTEJ71 runs successfully, it produces the return codes that are shown in the table below.

Table 148. DSNTEJ71 return codes

JOBSTEP	PROCSTEP	Return Code
PH071S01		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH071S02		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH071S03		0000
PH071S04		0000
PH071S05		0000

You can compare the output from this job with the sample output for DSNTEJ71 found in member DSN8TJ71 in your *prefix*.SDSNIVPD data set.

Job DSNTEJ73

Job DSNTEJ73 compiles the DSN8DLRV sample application. DSN8DLRV demonstrates how to use built-in functions like POSSTR and SUBSTR to traverse a CLOB column and extract data from it.

DSN8DLRV is written in C language and linked with ISPF and DB2 Call Attach Facility. The package name and plan name are both DSN8LR*vr*, where *vr* is the DB2 version and release.

If DSNTEJ73 runs successfully, it produces the return codes that are shown in the table below.

Table 149. DSNTEJ73 return codes

JOBSTEP	PROCSTEP	Return Code
PH073S01		
	PC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH073S02		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH073S03		0000 or 0004

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 7" on page 803

Job DSNTEJ75

Job DSNTEJ75 runs sample program DSN8DLPV, which demonstrates how to manipulate BLOB data (employee photo images).

This job is not edited by the CLIST unless the GDDM MACLIB and GDDM LOAD MODULES fields on panel DSNTIPW are non-blank.

DSN8DLPV is a C language program that is linked with ISPF, GDDM and the DB2 Call Attach Facility. The package name and plan name are both DSN8LP*vr*, where *vr* is the DB2 version and release. To run DSN8DLPV you must include the GDDM load module library (SADMMOD) in the logon procedure or in the ISPLLIB concatenation.

If DSNTEJ75 runs successfully, it produces the return codes that are shown in the table below.

Table 150. DSNTEJ75 return codes

JOBSTEP	PROCSTEP	Return Code
PH075S01		
	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH075S02		0000 or 0004

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 7" on page 803

Job DSNTEJ76

Job DSNTEJ76 compiles, link-edits, and binds two sample COBOL applications that manipulate LOB data.

The DSN8CLPL sample application demonstrates how to use LOB locators to populate a LOB column that is larger than 32 KB. The DSN8CLTC sample application validates the contents of the LOB table, verifying that it was populated correctly.

If DSNTEJ76 runs successfully, it produces the return codes that are shown in the table below.

Table 151. DSNTEJ76 return codes

JOBSTEP	PROCSTEP	Return code
PH076S01	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH076S02	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000

Table 151. DSNTEJ76 return codes (continued)

JOBSTEP	PROCSTEP	Return code
PH076S03		0000 or 0004
PH076S04		0000
PH076S05		0000

After running DSNTEJ76, you can compare the output with the sample output for DSNTEJ76 found in member DSN8TJ76 in your *prefix*.SDSNIVPD data set.

Job DSNTEJ77

Job DSNTEJ77 compiles the DSN8CLRV sample COBOL application. DSN8CLRV demonstrates how to use built-in functions, such as POSSTR and SUBSTR, to traverse a CLOB column and extract data from it.

DSN8CLRV is written in the COBOL language and linked with ISPF and DB2 Call Attach Facility. The package name and plan name are both DSN8CR*vr*, where *vr* is the DB2 version and release.

To run DSN8CLRV, you must run the first step of job DSNTEJ73 to compile DSN8SDM.

If DSNTEJ77 runs successfully, it produces the return codes that are shown in the table below.

Table 152. DSNTEJ77 return codes

JOBSTEP	PROCSTEP	Return Code
PH077S01		
	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH077S02		0000 or 0004

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 7" on page 803

Job DSNTEJ78

Job DSNTEJ78 runs sample COBOL program DSN8CLPV, which demonstrates how to manipulate BLOB data (employee photo images).

This job is not edited by the CLIST unless the GDDM MACLIB and GDDM LOAD MODULES fields on panel DSNTIPW are non-blank.

DSN8CLPV is a COBOL language program linked with ISPF, GDDM and the DB2 Call Attach Facility. The package name and plan name are both DSN8CP*vr*, where *vr* is the DB2 version and release. To run DSN8CLPV you must include the GDDM load module library (SADMMOD) in the logon procedure or in the ISPLLIB concatenation.

If DSNTEJ78 runs successfully, it produces the return codes that are shown in the table below.

Table 153. DSNTEJ78 return codes

JOBSTEP	PROCSTEP	Return Code
PH078S01		
	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH078S02		0000 or 0004

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 7"

Starting an application in an ISPF/TSO environment in phase 7

You must have access to ISPF load module libraries to run the Employee Resume and Photo sample applications.

Procedure

To start the application:

1. Enter a CALL command from option 6 of the ISPF primary option menu.
2. To start the Employee Resume and Photo sample applications, enter the following:

   ```
   CALL 'prefix.RUNLIB.LOAD(DSN8SDM)'
   ```

   After you enter this command, DB2 displays the sample applications panel.

   Choosing option 4 or option 6 on the sample applications panel during Phase 7 invokes the "Employee Resume" sample application, which processes CLOB data. Choosing option 5 or option 7 on the sample applications panel during Phase 7 invokes the "Employee Photo" sample application, which processes BLOB data.

Related tasks:

"Working with the employee resume and photo scenarios" on page 826

"Starting an application in an ISPF/TSO environment" on page 768

"Make DB2I panels, messages, and load modules available to ISPF and TSO" on page 586

The sample applications

After you run the installation verification jobs, you can use the sample applications to test and verify your new installation or migration.

The names of the sample applications have changed for Version 10. Check to make sure you have the authority to run the Version 10 sample programs.

Brief scenarios describe how to display, update, add, and delete information using the sample applications. Another scenario describes how to view or change information using a combination of organization and project applications. This scenario contains problem-solving exercises based upon creating and staffing a new department with new projects.

The output from the install verification steps discussed here appears in your *prefix*.SDSNIVPD data set.

Related tasks:

Managing explicit privileges (Managing Security)

Printing options for the sample application listings

The source statements that are contained in *prefix*.SDSNSAMP can be printed using ISPF facilities, IEBPTPCH, or local facilities.

Most of the DB2 sample applications are contained in *prefix*.SDSNSAMP. The modules making up the SQLCA formatter routine (DSNTIAR, DSNTIAC, DSNTIA1, and DSNTIAM) are not in the *prefix*.SDSNSAMP library. They are provided in object form in *prefix*.SDSNLOAD.

You might not want to print all members of *prefix*.SDSNSAMP because some of the members are large and contain unprintable data. An alternative is to precompile and compile the wanted program by specifying a cross-reference to the precompiler and compiler. This provides a cross-reference for program variables and is current.

The sample application panels

When you run the interactive sample applications, the sample application panels require you to enter information such as the action you want to complete.

You are prompted for the following information when you run the interactive sample applications.

Sample Applications	Distributed Sample Applications
• MAJOR SYSTEM • ACTION • OBJECT • SEARCH CRITERIA • DATA	• ACTION • OBJECT • SEARCH CRITERIA • LOCATION • DATA

Regard these categories as a family of values that are used together to specify a task that is to be performed. For MAJOR SYSTEM, ACTION, OBJECT, and SEARCH CRITERIA, a character code of one or two characters is used as a form of shorthand to indicate the desired criteria. The system provides a list of these codes with their meanings. A valid location name of 1 to 16 characters must be used for LOCATION. The value for DATA must be consistent with the data type and length of SEARCH CRITERIA.

MAJOR SYSTEM
: **MAJOR SYSTEM** specifies the major application area. In the sample application, there are two major systems: organization and project. These major systems are implemented in separate transactions to keep the plan sizes reasonable. If you are running the DB2 distributed sample program, organization is the only system; therefore, this criterion is not used.

ACTION
: **ACTION** specifies what you want to do with the object (specified on another line of the panel). You can display, update, add (insert), or erase (delete) information about the specified object.

OBJECT
: **OBJECT** specifies the object about which you want information. Normally, the

action is associated with the object. Examples of objects are information about an employee (EM) or information about the relationship among departments (DS).

Objects can be specified with the following codes for the organization application:

DE Department—general department and manager information for department specified

DS Department structure—hierarchy information for department specified

EM Employee—information concerning employee specified.

Objects can be specified with the following codes for the project application:

PS Project structure—information on projects and subprojects

AL Activity listing—information concerning the different activities that makes up a project

PR Project—general project information

AS Activity staffing—information about the employees staffed for activities of specified projects

AE Activity estimate—information concerning the estimated staffing and time requirements of specified projects.

You are able only to add, update, or erase information about the selected object, although you can search and display based on other criteria. Items that are added or updated can be changed on the screen. Other fields are protected.

SEARCH CRITERIA

SEARCH CRITERIA helps to locate the specific item of information upon which to act.

The following codes can be specified for the search criteria field for the organization application:

DI Department number

DN Department name

EI Employee number

EN Employee name

MI Manager number

MN Manager name.

The following codes can be specified for the search criteria for the project application:

DI Department number

DN Department name

EI Employee number

EN Employee name

PI Project number

PN Project name

RI Responsible person number

RN Responsible person name.

LOCATION

LOCATION is used only for the distributed application. It describes the location where the action is to take place. If this criterion is left blank, then the local location is assumed.

DATA

DATA further identifies the search criteria target. The data value specified must be consistent with the data type and length of the search criteria code. If the search criterion is an employee name (**EN**), manager name (**MN**), or responsible person name (**RN**), the value of data must be a person's last name.

Data values can be specified by using either primary selection or secondary selection. *Primary selection* is the data value itself. Only one set of data values fulfills the request. *Secondary selection* allows multiple sets of data values to fulfill the request. A brief summary of the sets of data values appear on the screen. Each summary has an associated line number. To display additional information about a certain line, enter the line number in the DATA field. Secondary selection allows the application to display a set of values and then provides a prompt to select a specific DATA value. For example, you can display information about a department (**DE**) (the OBJECT) with a department number (**DI**) (the SEARCH CRITERIA) with a **DATA** value of D11.

Related concepts:

"DATA field values on the sample application panels"

Related tasks:

"Connecting distributed database systems" on page 877

Allowable combinations of values on the sample application panels

The codes on the sample application panels cannot be combined indiscriminately.

For instance, manager number (MI) is a valid search criterion for a department (DE), but employee number (EI) and project number (PI) cannot be used to locate a department.

You can retrieve data by having the panels prompt you for the proper values. It is not necessary to enter the values one line at a time. If you already know all the values you want, they can be entered at the same time. If the values are only partially entered, you must start with ACTION and enter each value in sequence, not skipping over any values. For example, if you know all the values except OBJECT, only ACTION can be entered. You are prompted for OBJECT. Then you can enter OBJECT, SEARCH CRITERIA, and DATA.

DATA field values on the sample application panels

When you run the interactive sample applications, you must use the sample application panels to enter information about the task that you want to complete. Information that is entered in the DATA field on the sample application panels is used with the SEARCH CRITERIA field to define a search.

The values available for DATA are not limited to a select few as are the values for ACTION and OBJECT. There is a wider choice of DATA values and a variety of ways to express them.

If you know only part of a DATA value (for example, you know the department number begins with D), you can specify it as a *pattern*. The pattern can contain any character string with a special meaning, such as:

- The underscore character, _, represents any single character.

- The percent character, %, represents any string of zero or more characters.

These two special characters can be used in conjunction with other characters to specify a DATA value. The following table demonstrates three ways to use these characters to create a DATA value.

Table 154. Searching for data values

Data Value	Search Criteria	Description
%SMITH%	EN (Employee name)	Searches for any last name that contains the word SMITH; for example, BLACKSMITH, SMITHSONIAN, or NESMITHA
E_1	DI (Department number)	Searches for any department number with E in position 1 and 1 in position 3; for example, E71, E21, or EB1
%	Any	All values qualify

The values entered on the SEARCH CRITERIA and DATA fields can choose only one item to be displayed. However, the more usual case is that several items are displayed as a list. When this is the case, a secondary selection can be made by choosing the line number of the item of interest.

Function keys on the sample application panels

The bottom line of each sample application panel displays the function keys that are active for that panel.

Function key 2—Resend: If the panel is blanked out (for example, you pressed the CLEAR key) or you want to refresh the panel, press function key 2 to return (resend) the display you were viewing to the terminal.

Function key 3—End To terminate the application, press function key 3 to clear the screen and continue with other transactions.

Function key 8—Next: Sometimes a display of information is too large to fit on one panel. Press function key 8 to scroll forward (the lines move upward).

Function key 10—Left: Press function key 10 to move the field of vision up one level in the department structure. For instance, in the organization application, Department E01 is shown on the left, and its subdepartments are shown on the right. When you press function key 10, the screen scrolls so that Department E01 is moved from the left side of the panel to the right side and the department to which it reports appears on the left. All other departments that report to the department now on the left also appear along with Department E01 on the right. Function key 10 performs this function only for IMS and CICS samples.

Related tasks:

"Working with the organization application scenario" on page 810

Working with scenarios

Work with the scenarios for using the sample applications. The method for invoking these applications depends on the environment that you are working in.

About this task

When an application runs, many areas on the display panel might be highlighted. The data that you enter might not be highlighted, depending on the type of panel that is displayed.

Related tasks:

"Starting an application in an ISPF/TSO environment in phase 6" on page 781

"Starting an application in a CICS environment" on page 776

"Starting an application in an IMS environment" on page 772

"Starting an application in an ISPF/TSO environment" on page 768

Working with the project application scenario

The project application scenario demonstrates how to view details of a project and update activities that are associated with a project.

About this task

For example, you can find the person responsible for a project and list the activities that are assigned to one of its subprojects. Phase 4 (IMS) and Phase 5 (CICS) prepare the programs that you execute.

Procedure

To view details of a project with the sample application:

Enter the following values on the first panel of the project application:

- On the MAJOR SYSTEM line, enter `P` for project.
- On the ACTION line, enter `D` for display.
- On the OBJECT line, enter `PS` for project structure.
- On the SEARCH CRITERIA line, enter `PI` for project ID.
- On the DATA line, enter `MA2100` as the project ID.

The following example panel shows the selected project and its corresponding subprojects.

```
              PROJECT STRUCTURE
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........: D          DISPLAY (SHOW)
OBJECT .........: PS         PROJECT STRUCTURE
SEARCH CRITERIA.: PI         PROJECT ID
DATA ...........: MA2100

PROJECT ID & NAME                      SUBPROJECT ID & NAME
 RESPONSIBLE ID & NAME                  RESPONSIBLE ID & NAME

MA2100  WELD LINE AUTOMATION            MA2110  W L PROGRAMMING
 000010  CHRISTINE I HAAS                000060  IRVING F STERN

                                        PL2100  WELD LINE PLANNING
                                         000020  MICHAEL L THOMPSON


PFK: 02=RESEND 03=END 08=NEXT 10=LEFT
```

Figure 87. Project application—viewing a project structure

Updating an activity

You can use the project application to update an activity information for a project, such as the estimated start date.

About this task

Suppose that you want to update activity information for a project with ID IF1000.

Procedure

To update the activity:

1. Enter the following values:
   - On the MAJOR SYSTEM, enter `P` for project.
   - On the ACTION line, enter `U` for update.
   - On the OBJECT line, enter `AE` for activity estimate.
   - On the SEARCH CRITERIA line, enter `PI` for project ID.
   - On the DATA line, enter `IF1000` as the project ID.
2. Press the ENTER key, and a list of project IF1000 activities appears on the panel.
3. Choose the activity to be updated. For instance, if you want to update the first activity listed, enter 1 as the DATA value and press the ENTER key. The next panel shows information about the estimated mean staffing requirements of this activity as well as the start and completion dates.
4. To change information about the estimated end date, enter data over the existing information displayed on that input line.
5. After you have verified the change, press ENTER. The next panel displays the updated information, as shown in the following figure:

```
              UPDATING OF AN ACTIVITY ESTIMATE
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........: U          UPDATE (CHANGE)
OBJECT .........: AE         ACTIVITY ESTIMATE
SEARCH CRITERIA : PI         PROJECT ID
DATA ...........: 01
DSN8024I  DSN8MPX - ACTIVITY SUCCESSFULLY UPDATED
  PROJECT  ID                       :  IF1000
           NAME                     :  QUERY SERVICES

  ACTIVITY ID                       :         90
              KEYWORD               :  ADMQS
              DESCRIPTION           :  ADM QUERY SYSTEM
           EST MEAN STAFFING        :      2.00
           EST START DATE           :  1982-01-01
           EST END DATE             :  1983-04-15

PFK: 02=RESEND 03=END
```

Figure 88. Project application—changes accepted

6. To terminate the project application, press the PF3 key. The APPLICATION TERMINATED message is displayed.

What to do next

If you are using CICS, clear the screen and enter a new transaction code. If you are using IMS, clear the screen and enter a new transaction code or a /FORMAT command.

Working with the organization application scenario

The organization application scenario shows how to display a list of departments within an organization and the structure of one of these departments.

About this task

This application is executed in Phase 4 for IMS and Phase 5 for CICS.

Procedure

To display a list of departments and the structure of one of the departments:

1. After you enter the appropriate transaction code, you see the first panel of the project application. Enter the following values:
   - On the MAJOR SYSTEM, enter O for organization.
   - On the ACTION line, enter D for display.
   - On the OBJECT line, enter DS for department structure.
   - On the SEARCH CRITERIA line, enter DI for department number.
   - On the DATA line, enter %, which enables you to display a list of all the departments.

   Each department entry is numbered on the far left side of the panel as shown in the figure below.

```
              DEPARTMENT ADMINISTRATIVE STRUCTURE SELECTION
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........: D          DISPLAY (SHOW)
OBJECT .........: DS         DEPARTMENT STRUCTURE
SEARCH CRITERIA.: DI         DEPARTMENT ID
DATA ...........: %
SELECT A DEPARTMENT FROM FOLLOWING LIST BY SPECIFYING THE LINE NUMBER
NO  D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME
01  A00   SPIFFY COMPUTER SERVICES DIV.        000010   CI HAAS
02  B01   PLANNING                               000020   ML THOMPSON
03  C01   INFORMATION CENTER                     000030   SA KWAN
04  D01   DEVELOPMENT CENTER
05  D11   MANUFACTURING SYSTEMS                  000060   IF STERN
06  D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
07  E01   SUPPORT SERVICES                       000050   JB GEYER
08  E11   OPERATIONS                             000090   EW HENDERSON
09  E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
PFK:  02=RESEND 03=END 08=NEXT
```

Figure 89. Organization application—viewing a list of departments

2. To retrieve further information, specify a line number as a data value. This method is called secondary selection. Secondary selection provides prompts to aid in finding the information to be displayed, added, erased, or updated. If only one entry possibility exists, secondary selection is not offered. For example, to view the structure of Department E01, specify a data value of 7 on the DATA entry line (7 is the line number of the entry for Department E01). The result of entering the data value of 7 is a display of Department E01 and its departments as shown in the figure below. The department manager for E01 is listed on the left, and the departments of E01 are listed on the right. Employees of E01 are listed below the subdepartments of E01.

```
              DEPARTMENT ADMINISTRATIVE STRUCTURE
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........: D          DISPLAY (SHOW)
OBJECT .........: DS         DEPARTMENT STRUCTURE
SEARCH CRITERIA.: DI         DEPARTMENT ID
DATA ...........: 07

DEPARTMENT ID & NAME                    SUBDEPARTMENT ID, NAME & MANAGER
 MANAGER ID & NAME                      EMPLOYEE ID & NAME
E01  SUPPORT SERVICES                    E11  OPERATIONS
 000050  JOHN B GEYER                     000090  EILEEN W HENDERSON

                                         E21  SOFTWARE SUPPORT
                                          000100  THEODORE Q SPENSER

                                          000050  JOHN B GEYER

PFK:  02=RESEND 03=END 08=NEXT 10=LEFT
```

Figure 90. Organization application—viewing a department structure

Starting a new operation

You can use the organization application to start a new operation such as displaying or updating a department structure.

Procedure

To start a new operation:

Choose one of the following actions:

- Move the cursor to the D on the ACTION line and retain the D or change it to a different action (add, erase, or update). Follow the displayed options to perform your selected action.
- Alternatively, you can leave the organization application by pressing the PF3 key. If you are using CICS, enter the transaction code. If you are using IMS, clear the screen and enter the /FORMAT command to select the project application. In either case, to proceed with a different operation, select a different ACTION, OBJECT, and so forth.

Adding a new department

You can use the organization application to add a new department and specify information about the department.

Procedure

To add a department to the organization:

1. Start the organization application and enter the following values:
   - On the MAJOR SYSTEM line, enter O for organization
   - On the ACTION line, enter A for add (insert)
   - On the OBJECT line, enter DE for the department that is to be added
   - On the SEARCH CRITERIA line, enter DI for department ID
   - On the DATA line, enter C11, the specific department number
2. Next, you can enter the details of the new department. The four department fields are department number, department name, manager number, and administration department number. Enter the following values:
   - INFORMATION SERVICES for department name
   - 000130 for manager number
   - C01 for the administration department number

Press ENTER to display the panel shown in the figure below. The panel shows the successful addition of the new department.

```
              ADDING A NEW DEPARTMENT
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........: A          ADD (INSERT)
OBJECT .........: DE         DEPARTMENT
SEARCH CRITERIA.: DI         DEPARTMENT ID
DATA ...........: C11
DSN8012I  DSN8MPE - DEPARTMENT SUCCESSFULLY ADDED
  DEPARTMENT ID                    :  C11
             NAME                  :  INFORMATION SERVICES
             MANAGER ID            :  000130
             ADMIN DEP ID          :  C01
  MANAGER    ID                    :  000130
             FIRST NAME            :  DOLORES
             MIDDLE INITIAL        :  M
             LAST NAME             :  QUINTANA
             WORK DEPT ID          :  C01

PFK: 02=RESEND 03=END
```

Figure 91. Organization application—adding a department

Related tasks:

"Starting a new operation" on page 811

Deleting an entry

You can use the organization application to delete an entry from the department table.

About this task

Deleting an entry from the department table is a function of the organization major system.

Procedure

To delete an entry from the department table:

1. Replace the following values on the panel that is on your screen:

   ,

   - On the MAJOR SYSTEM line, enter O for organization.
   - On the ACTION line, enter E for erase.
   - On the OBJECT line, enter DE for department.
   - On the SEARCH CRITERIA line, enter DI for department ID.
   - On the DATA line, enter C11 for department name.
2. Press ENTER to display the panel shown in the figure below.

```
              ERASING A DEPARTMENT
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........: E          ERASE (REMOVE)
OBJECT .........: DE         DEPARTMENT
SEARCH CRITERIA.: DI         DEPARTMENT ID
DATA ...........: C11
PRESS ENTER TO ERASE A DEPARTMENT
  DEPARTMENT ID                    :  C11
             NAME                  :  INFORMATION SERVICES
             MANAGER ID            :  000130
             ADMIN DEP ID          :  C01
  MANAGER    ID                    :  000130
             FIRST NAME            :  DOLORES
             MIDDLE INITIAL        :  M
             LAST NAME             :  QUINTANA
             WORK DEPT ID          :  C01

PFK: 02=RESEND 03=END
```

Figure 92. Organization application—deletion successful

3. Press ENTER again to verify the erase action. The following message appears on the panel:

   `DSN8013I  csect DEPARTMENT SUCCESSFULLY ERASED`

Related tasks:

"Starting a new operation" on page 811

Transferring an employee

The procedure for transferring one employee to another department and replacing that employee involves several steps.

About this task

In this scenario, John B. Geyer (manager of the department for Support Services) is transferred to the staff of Spiffy Computer Service Division. Bruce Adamson is assigned as manager of Support Services.

To move Adamson into his new position as manager of Support Services, you must determine his employee number. Transferring an employee is a function of the organization major system.

Procedure

To transfer an employee to another department:

1. Start the organization application and enter the following values:
   - On the MAJOR SYSTEM, enter O for organization.
   - On the ACTION line, enter D for display.
   - On the OBJECT line, enter EM for employee.
   - On the SEARCH CRITERIA line, enter EN for employee name.
   - On the DATA line, enter ADAMSON as the specific employee name.
2. Press ENTER to display the panel showing that Adamson's employee number is 000150.
3. The next step is for you to change the manager number for the Support Services department to Adamson's number, 000150. But first you must find the Support Services department. To do this, change ACTION to U (update), OBJECT to DE (department), and SEARCH CRITERIA to DN (department name). Change DATA to %SUPPORT% to specify any department with the word SUPPORT in it.

4. Press ENTER, and a list of departments with support in their name is displayed. Support Services has line number 01. Enter this number at DATA. (The leading zero is not needed.)
5. Press ENTER to display the next panel. The only values that can be changed are department name, manager ID, and administration department ID. Enter Adamson's employee number in the Support Services department after MANAGER ID. At this point, the data on the manager still pertains to Geyer.
6. Press ENTER to display the panel that shows Adamson as manager of Support Services. The work department ID shown (D11) is still Adamson's old number. To change Adamson's work department ID, enter EM for OBJECT, enter EI for SEARCH CRITERIA, and change the employee number to 000150 for DATA.
7. Press ENTER to display the employee information on Adamson. Now that information on Adamson can be updated. The fields that can be changed are employee first name, middle initial, last name, and work department ID. Enter the middle initial for Adamson, which was not in the database, and the department number E01. Press ENTER, and the information on Adamson is updated, including his new department number.
8. The final step is to move Geyer to the correct department. Change the SEARCH CRITERIA and DATA to EN and GEYER, respectively. Press ENTER to obtain the next panel. The employee ID, name, and work department ID can be changed on this panel. However, the only change necessary in this case is to change Geyer's work department ID to his new one, A00. The panel in the figure below shows the completed entry.

```
              UPDATING AN EMPLOYEE
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........: U          UPDATE (CHANGE)
OBJECT .........: EM         EMPLOYEE
SEARCH CRITERIA.: EN         EMPLOYEE NAME
DATA ...........: GEYER
DSN8004I  DSN8MPF - EMPLOYEE SUCCESSFULLY UPDATED
  DEPARTMENT ID                    :  A00
             NAME                  :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID            :  000010
             ADMIN DEP ID          :  A00
  EMPLOYEE   ID                    :  000050
             FIRST NAME            :  JOHN
             MIDDLE INITIAL        :  B
             LAST NAME             :  GEYER
             WORK DEPT ID          :  A00

PFK: 02=RESEND 03=END
```

Figure 93. Organization application—employee data update completed

9. To terminate the application and return to the beginning of the operation, press the PF3 key.

Related tasks:

"Starting a new operation" on page 811

Working with the phone application scenario

The phone application retrieves information from a phone directory and updates employee phone numbers.

About this task

The phone application is used in phase 2 (batch mode), phase 3 (CAF), and interactively in phase 4 (IMS) and phase 5 (CICS).

The phone directory consists of data from a combination (join) of the employee table (DSN81010.EMP) and the department table (DSN81010.DEPT). This joined view is called VPHONE. The program also uses a second view called VEMPLP to update the employee table, which does not affect a view that joins tables.

The phone application is designed to operate in batch and interactively in ISPF/TSO, IMS, and CICS. The following table describes the environments in which each phone application operates and the language in which each is written.

Table 155. Phone programs

Environment	Language	Name
ISPF/TSO	COBOL	DSN8SC3
ISPF/TSO	PL/I	DSN8SP3
IMS	PL/I	DSN8IP3
CICS	PL/I	DSN8CP3
batch	COBOL	DSN8BC3
batch	Fortran	DSN8BF3
batch	PL/I	DSN8BP3
batch	C	DSN8BD3

Related tasks:

"Starting an application in an ISPF/TSO environment" on page 768

Viewing or changing an employee's telephone number

A couple of panels are used to view or change an employee's telephone number in the sample telephone application. These panels are the same, regardless of whether IMS or CICS is used.

Procedure

To view or change an employee's telephone number:

1. On the first panel of the telephone application, enter the first and last name of the employee whose telephone number you want to view or change. To see an entire listing of employee numbers, put an * next to the LAST NAME input line. If only part of a first or last name is known, use the percent character (%) to qualify the list of names to appear in the directory. For example, entering K% on the LAST NAME input line calls a list of the telephone numbers of all employees whose last name begins with a K. Similarly, the first name can be qualified.

   To keep this sample program as simple as possible and to allow updating, scrolling is not used with the IMS and CICS versions. Scrolling is used with the ISPF/CAF version. Only the first panel of selected names and phone numbers can be displayed. The second panel is the Telephone Directory itself. The employee telephone number is highlighted. To update an employee telephone number, type over the highlighted number and press ENTER. To update a phone number listed under the name Heather A Nicholls, specify NICHOL% when you are not sure if there are one or two ls in Nicholls.

```
------------------------- TELEPHONE DIRECTORY --------------------


                            LAST  NAME ==> _

                            FIRST NAME ==>




LAST NAME              *  FOR LIST OF ENTIRE DIRECTORY
                       %  FOR GENERIC LIST (EX. K% = ALL  K - NAMES)
FIRST NAME(OPTIONAL)   %  FOR GENERIC LIST
```

Figure 94. Telephone application—first display

2. Press ENTER to display the panel on which the phone number is highlighted.
3. If you want to change a phone number, type over the number to be changed. You can type over as many numbers as are listed in the display. For example, suppose that you want to change the phone number from 1793 to 1795. After you press ENTER, you get a message confirming the updated phone number. The panel in the following figure shows the updated panel.

```
--------------------------- TELEPHONE DIRECTORY --------------------

 FIRST NAME MID LAST NAME  PHONE  EMPL WORK WORKDEPT
           INIT              NO    NO  DEPT NAME

HEATHER  A      NICHOLLS     1795 000140 C01 INFORMATION CENTER
.
.
.
```

Figure 95. Telephone application—updated display

Using the phone application under batch

The sample batch phone applications are provided in Fortran (DSN8BF3), COBOL (DSN8BC3), C (DSN8BD3), and PL/I (DSN8BP3).

Procedure

To update or list an employee phone number:

Create a data set that contains information about the phone number that is to be updated. This data set works in combination with another data set that contains JCL for processing information. The first data set consists of card images in the format that is shown in the following table.

Table 156. Format of phone application data set

Column	Description
1	ACTION—U for update, L for list
2	Employee last name
17	Employee first name
29	Employee number
35	New phone number

The ACTION code in this card image indicates whether an employee number is to

be updated (U) or listed (L). When updating an employee phone number, only the employee number and the new phone number are specified in the data set. When listing phone numbers, the last name must be specified. Specifying the first name is optional. The * and % can be used with the ACTION code just as they are used with the panels.

Example

The following figure shows an example of an update data set and a list data set. Each time a number is listed or updated, a new data set is created containing a card image like the one in the following figure. The first card in the data set shows the phone number of employee number 000140 being updated (U) to 6767. The second card shows a list (L) for Heather Nicholls. The last card shows a list (L) of all employees whose first names begin with the letters MAR. The example shows the letters MAR followed by a % in the first name column to indicate that only those employees whose first names begin with MAR are to be listed.

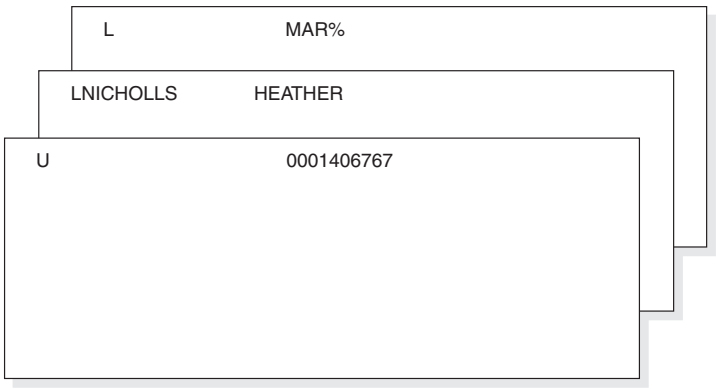


Figure 96. Example of a card image data set

The data set that contains the JCL for processing information is supplied with DB2 and is contained in DSNTEJ2P, which is part of *prefix*.SDSNSAMP. The figure below shows the data set that contains the JCL with the card image data sets embedded.

```
//PH02PS05 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//REPORT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//CARDIN DD *
U                          0001406767
LNICHOLLS       HEATHER
L               MAR%
//SYSTSIN DD *
 DSN SYSTEM(DSN)
 RUN  PROGRAM(DSN8BP3) PLAN(DSN8BP10) LIB('prefix.RUNLIB.LOAD')
 END
```

Figure 97. The job control language data set

The complete data set can be submitted to the system either through a card reader or from a terminal through TSO. The figure below contains an example of the batch output.

```
---------------------------- TELEPHONE DIRECTORY -----------------------

LAST NAME FIRST NAME INITIAL PHONE  EMPLOYEE WORK WORK
                             NUMBER NUMBER   DEPT DEPT NAME

QUINTANA  DOLORES       M    6767   000130   C01  INFORMATION CENTER
NICHOLLS  HEATHER       A    1793   000140   C01  INFORMATION CENTER
SCOUTTEN  MARILYN       S    1682   000180   D11  MANUFACTURING SYSTEMS
PEREZ     MARIA         L    9001   000270   D21  ADMINISTRATIVE SYSTEMS
```

Figure 98. Example of the phone application batch output

Working with the distributed organization application scenario

The distributed organization application scenario shows how to work with department and employee data in a distributed environment.

About this task

The scenario shows how to display a department structure, display department information, and update a department at a local location. It also shows how to erase and add an employee at a remote location. This application is executed in Phase 6. The application accesses distributed data with DRDA access.

The department information (DEPT table) is shared by all locations. If you make changes to DEPT table at one location, the DEPT tables at the other locations are updated at the same time. The employee information (EMP table) is unique to each location, containing only the employees that work at that particular location.

After you enter the appropriate transaction code, you see the first panel of the organization application.

Displaying department structure at the local location

You can use the distributed organization sample application to display department structure at the local location.

Procedure

To display a department structure:

1. Enter the following values in the distributed organization sample application:
   - On the ACTION line, enter `D` for display.
   - On the OBJECT line, enter `DS` for department structure.
   - On the SEARCH CRITERIA line, enter `DI` for department number.
   - On the LOCATION line, leave blank, indicating local location.
   - On the DATA line, enter `A00` for department number.

```
                        DB2 ORGANIZATION APPLICATION
===>_

 ACTION .........:d       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:ds      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:       (Blank implies local location)

 DATA ...........:a00

 PRESS: ENTER to process    END to exit
```

Figure 99. Starting the distributed organization application

2. Press the ENTER key. The panel below shows the structure of the department requested.

```
                        DB2 ORGANIZATION APPLICATION          ROW 1 of 5
===>_
 PRESS:  ENTER TO PROCESS    END TO EXIT

 DEPARTMENT STRUCTURE FOR:

     ----- DEPARTMENT ID AND NAME------  ----- MANAGER ID AND NAME-------------

     A00 SPIFFY COMPUTER SERVICE DIV.  000010 CHRISTINE     I  HAAS

 SUBDEPARTMENTS:

     A00  SPIFFY COMPUTER SERVICE DIV.   000010  CHRISTINE    I  HAAS
     B01  PLANNING                       000020  MICHAEL      L  THOMPSON
     C01  INFORMATION CENTER             000030  SALLY        A  KWAN
     D01  DEVELOPMENT CENTER
     E01  SUPPORT SERVICES               000050  JOHN         B  GEYER
```

Figure 100. Displaying department structure

3. Press ENTER or END to exit.

Displaying department information at the local location

You can use the distributed organization sample application to display department structure at the local location.

Procedure

To display department information:

1. Enter the following values in the distributed organization sample application:
   - On the ACTION line, enter `D` for display.
   - On the OBJECT line, enter `DE` for department.
   - On the SEARCH CRITERIA line, enter `DI` for department number.
   - On the LOCATION line, leave blank, indicating local location.
   - On the DATA line, enter `A00` for department number.

```
                        DB2 ORGANIZATION APPLICATION
===>_

 ACTION .........:d       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:de      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:        (Blank implies local location)

 DATA ...........:a00

 PRESS: ENTER to process    END to exit
```

Figure 101. Starting the distributed organization application

2. Press the ENTER key. The panel below shows the department information that was requested.

```
                        DB2 ORGANIZATION APPLICATION
===>_

                             DISPLAY A DEPARTMENT

  DEPARTMENT ID                    :  A00
             NAME                  :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID            :  000010
             ADMIN DEP ID          :  A00
             LOCATION              :

  MANAGER    ID                    :  000010
             FIRST NAME            :  CHRISTINE
             MIDDLE INITIAL        :  I
             LAST NAME             :  HAAS
             WORK DEPT ID          :  A00


  PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 102. Displaying department information

3. Press ENTER or END to exit.

Updating a department at the local location

You can use the distributed organization sample application to update department information at the local location.

Procedure

To update department information:

1. Enter the following values in the distributed organization sample application:
   - On the ACTION line, enter `u` for update.
   - On the OBJECT line, enter `de` for department.
   - On the SEARCH CRITERIA line, enter `di` for department number.
   - Leave the LOCATION line blank, indicating local location.
   - On the DATA line, enter `%`, which displays a list of all the departments.

```
                        DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:u       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:de      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:        (Blank implies local location)

 DATA ...........:%

 PRESS: ENTER to process    END to exit
```

Figure 103. Starting the distributed organization application

2. Press the ENTER key. The panel below lists the departments that can be updated. Select the department to be updated by inserting an S in the left margin by the department number.

```
                        DB2 ORGANIZATION APPLICATION      ROW 1 to 14 of 14
===>_
    ACTION:  UPDATE A DEPARTMENT

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE DEPARTMENT
    PRESS ENTER TO PROCESS OR END TO EXIT

    D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME

    A00   SPIFFY COMPUTER SERVICE DIV.           000010   CI HAAS
    B01   PLANNING                               000020   ML THOMPSON
    C01   INFORMATION CENTER                     000030   SA KWAN
    D01   DEVELOPMENT CENTER
    D11   MANUFACTURING SYSTEMS                  000060   IF STERN
    D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
S   E01   SUPPORT SERVICES                       000050   JB GEYER
    E11   OPERATIONS                             000090   EW HENDERSON
    E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
    F22   BRANCH OFFICE F2
    G22   BRANCH OFFICE G2
    H22   BRANCH OFFICE H2
    I22   BRANCH OFFICE I2
    J22   BRANCH OFFCIE J2
```

Figure 104. Selecting a department to be updated

3. Press the ENTER key. The panel below displays the information relevant to the selected department. Enter the information you want to update on this panel; in this case, enter the name of the department.

```
                      DB2 ORGANIZATION APPLICATION
===>_

                            UPDATE A DEPARTMENT

  DEPARTMENT ID                    :  E01
             NAME                  :  hardware support service
             MANAGER ID            :  000050
             ADMIN DEP ID          :  A00
             LOCATION              :

  MANAGER    ID                    :  000050
             FIRST NAME            :  JOHN
             MIDDLE INITIAL        :  B
             LAST NAME             :  GEYER
             WORK DEPT ID          :  E01

 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 105. Updating a department

4. Press the ENTER key to process the updated information. A message appears on this panel that states the update was successful.

```
                      DB2 ORGANIZATION APPLICATION
===>_
DSN8014I   DSN8HC3-DEPARTMENT SUCCESSFULLY UPDATED
                            UPDATE A DEPARTMENT

  DEPARTMENT ID                    :  E01
             NAME                  :  HARDWARE SUPPORT SERVICE
             MANAGER ID            :  000050
             ADMIN DEP ID          :  A00
             LOCATION              :

  MANAGER    ID                    :  000050
             FIRST NAME            :  JOHN
             MIDDLE INITIAL        :  B
             LAST NAME             :  GEYER
             WORK DEPT ID          :  E01

 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 106. Update successfully processed

5. Press ENTER to return to the previous panel or END to exit. If you return to the previous panel, you can now select another department to update, or press ENTER or END to exit. The same message appears on this panel, indicating the update was successful.

```
                        DB2 ORGANIZATION APPLICATION      ROW 1 to 14 of 14
===>_
DSN8014I   DSN8HC3-DEPARTMENT SUCCESSFULLY UPDATED
    ACTION:  UPDATE A DEPARTMENT

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE DEPARTMENT
    PRESS ENTER TO PROCESS OR END TO EXIT

    D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME

    A00   SPIFFY COMPUTER SERVICE DIV.           000010   CI HAAS
    B01   PLANNING                               000020   ML THOMPSON
    C01   INFORMATION CENTER                     000030   SA KWAN
    D01   DEVELOPMENT CENTER
    D11   MANUFACTURING SYSTEMS                  000060   IF STERN
    D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
    E01   HARDWARE SUPPORT SERVICE               000050   JB GEYER
    E11   OPERATIONS                             000090   EW HENDERSON
    E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
    F22   BRANCH OFFICE F2
    G22   BRANCH OFFICE G2
    H22   BRANCH OFFICE H2
    I22   BRANCH OFFICE I2
    J22   BRANCH OFFICE J2
```

Figure 107. Department successfully updated

Adding an employee at a remote location

You can use the distributed organization sample application to add information for employees at remote locations.

Procedure

To add an employee at a remote location:

1. Enter the following values:
   - On the ACTION line, enter `a` for add.
   - On the OBJECT line, enter `em` for employee.
   - On the SEARCH CRITERIA line, enter `ei` for employee number.
   - On the LOCATION line, enter your server location for the remote location.
   - On the DATA line, enter the employee ID of the employee to be added (`sj0100` in the example below).

```
                        DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:a       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:em      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :ei      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:your server location  (Blank implies local location)

 DATA ...........:sj0100

 PRESS: ENTER to process    END to exit
```

Figure 108. Starting the distributed organization application

2. Press the ENTER key. The panel below allows you to enter information only about the employee. The department information on the panel is protected. Enter the necessary information about the employee.

```
                        DB2 ORGANIZATION APPLICATION
===>_

                             ADD     AN EMPLOYEE

  DEPARTMENT ID                     :
             NAME                   :
             MANAGER ID             :
             ADMIN DEP ID           :
             LOCATION               :

  EMPLOYEE   ID                     :  SJ0100
             FIRST NAME             :  W
             MIDDLE INITIAL         :
             LAST NAME              :  WALTERS
             WORK DEPT ID           :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 109. Employee to be added

3. Press the ENTER key to process or END to exit. If you press the ENTER key, a message appears on the panel indicating that the employee has been added.

```
                        DB2 ORGANIZATION APPLICATION
===>_
DSN8002I    DSN8HC3-EMPLOYEE SUCCESSFULLY ADDED
                             ADD     AN EMPLOYEE

  DEPARTMENT ID                     :  F22
             NAME                   :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID             :
             ADMIN DEP ID           :  E01
             LOCATION               :  SAN_JOSE

  EMPLOYEE   ID                     :  SJ0100
             FIRST NAME             :  W
             MIDDLE INITIAL         :
             LAST NAME              :  WALTERS
             WORK DEPT ID           :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 110. Employee successfully added

4. Press the ENTER key to return to the selection panel or END to exit.

Erasing an employee at a remote location

You can use the distributed organization sample application to erase information for employees at remote locations.

Procedure

To erase an employee at a remote location:

1. Enter the following values in the distributed organization sample application:
   - On the ACTION line, enter `e` for erase.
   - On the OBJECT line, enter `em` for employee.
   - On the SEARCH CRITERIA line, enter `ei` for employee number.

- On the LOCATION line, enter your server location for the remote location.
- On the DATA line, enter %, which displays a list of all the employees.

```
                      DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:e       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:em      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :ei      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:your server location (Blank implies local location)

 DATA ...........:%

 PRESS: ENTER to process    END to exit
```

Figure 111. Starting the distributed organization application

2. Press the ENTER key. The panel below lists the employees that can be erased. Select the employee to be erased by inserting an S in the left margin by the employee ID.

```
                      DB2 ORGANIZATION APPLICATION       ROW 1 OF 4
===>_

    ACTION: ERASE   AN EMPLOYEE                  LOCATION: SAN_JOSE

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE EMPLOYEE
    PRESS ENTER TO PROCESS OR END TO EXIT

    E/ID     EMPLOYEE NAME                       D/ID     DEPARTMENT NAME

S   SJ0100   W  WALTERS                          F22      BRANCH OFFICE F22
    SJ0020   S  O'SHEA                           F22      BRANCH OFFICE F22
    SJ0030   D  COOPER                           F22      BRANCH OFFICE F22
    SJ0040   A  HAYES                            F22      BRANCH OFFICE F22
    SJ0050   L  ASHER                            F22      BRANCH OFFICE F22
```

Figure 112. Selecting an employee at a remote location

3. Press the ENTER key. The panel below displays the information relevant to the selected employee that is to be erased.

```
                      DB2 ORGANIZATION APPLICATION
===>_

                           ERASE   AN EMPLOYEE

  DEPARTMENT ID                    :  F22
             NAME                  :  BRANCH OFFICE F22
             MANAGER ID            :
             ADMIN DEP ID          :  E01
             LOCATION              :  SAN_JOSE

  EMPLOYEE   ID                    :  SJ0100
             FIRST NAME            :  W
             MIDDLE INITIAL        :
             LAST NAME             :  WALTERS
             WORK DEPT ID          :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 113. Employee to be erased

4. Press the ENTER key to erase the employee information. A message appears on the panel stating the employee has been successfully erased. You can now erase another employee or press ENTER or END to exit.

```
                      DB2 ORGANIZATION APPLICATION       ROW 1 OF 3
===>_
DSN8003I   DSN8HC3-EMPLOYEE SUCCESSFULLY ERASED
    ACTION:   ERASE AN EMPLOYEE                  LOCATION: SAN_JOSE

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE EMPLOYEE
    PRESS ENTER TO PROCESS OR END TO EXIT

    E/ID     EMPLOYEE NAME                       D/ID     DEPARTMENT NAME

    SJ0020   S  O'SHEA                           F22      BRANCH OFFICE F22
    SJ0030   D  COOPER                           F22      BRANCH OFFICE F22
    SJ0050   L  ASHER                            F22      BRANCH OFFICE F22
```

Figure 114. Employee at a remote location erased

Working with the employee resume and photo scenarios

The LOB sample application extends the existing DB2 sample employee database by adding a new table for storing employee resumes and photographs as CLOB and BLOB entries.

About this task

The supporting JCL, application objects, and input data for this table are provided.

The purpose of the LOB sample application is to:

- Provide an IVP for LOB functions.
- Demonstrate how to:
  - Use DDL to create LOB objects.
  - Use the DB2 LOAD utility to populate LOB columns of 32K bytes or less.
  - Create an application program to populate LOB columns of greater than 32K bytes.

– Use LOB locators and related functions to manipulate LOBs without materializing the data.

The LOB sample application consists of a batch portion and an optional online portion. The batch portion verifies that LOB objects can be created, populated, and successfully read by using locators and supporting functions. The online portion demonstrates further techniques for using and manipulating LOB data.

The batch portion of the LOB sample application consists of the following jobs: DSNTEJ7, DSNTEJ71, DSNTEJ73, DSNTEJ75, DSNTEJ76, DSNTEJ77, and DSNTEJ78.

The optional online portion of the LOB sample application adds two additional scenarios to the existing three scenarios that are provided in IVP phase 3. These scenarios include the following user activities:

1. Invoking the DB2 sample connection manager to display the DB2 ISPF sample application menu, DSN8SSM.
2. Selecting and viewing sample employee resumes by using option 4 of DSN8SSM.
3. Selecting and viewing sample employee photo images by using option 5 of DSN8SSM.

Application programs for the LOB sample are written in the C and COBOL languages. The resume viewer requires ISPF. The photo viewer requires ISPF and GDDM.

Related concepts:

"Phase 7: Accessing LOB data" on page 798

Using the resume application in the LOB application scenario

You can view employee resumes with the resume application in the LOB application scenario.

About this task

The LOB application begins with the initial panel of the ISPF/CAF application.

Procedure

To view employee resumes:

1. From the DB2 Sample Application Menu, DSN8SSM, select one of the following options:
   - If you ran job DSNTEJ73, select option 4.
   - If you ran job DSNTEJ77, select option 6.

   DSN8DLRV and DSN8CLRV show the following panels for selecting and viewing sample employee resumes until you press the END key to exit ISPF.

```
DSN8SSE                  DB2 EMPLOYEE SELECTION PANEL
===>_



SELECT ONE OF THE EMPLOYEES AND PRESS ENTER.

    1. 000130 - DELORES M. QUINTANA
    2. 000130 - HEATHER A. NICHOLLS
    3. 000130 - BRUCE ADAMSON
    4. 000130 - JAMES H. WALKER


PRESS:    END TO EXIT
```

Figure 115. DB2 Employee Selection Panel

2. Select from the list of available resumes that are displayed on ISPF panel DSN8SSE.

   The employee serial numbers and names are hard-coded into the panel because the data for this sample is predetermined. The formatted resume is displayed on ISPF panel DSN8SSR:

```
DSN8SSR                  DB2 EMPLOYEE RESUME APPLICATION
===>_

PERSONAL INFORMATION:                 DEPARTMENT INFORMATION:
- NAME: DELORES M. QUINTANA           - EMPLOYEE NO. : 000130
- HOME: 1150 EGLINTON AVE             - DEPARTMENT NO.: C01
        MELLONVILLE, IDAHO 83757      - MANAGER       : SALLY KWAN
        (208) 555-9933                - POSITION      : ANALYST
- BORN: SEPTEMBER 15, 1925            - PHONE         : (208) 555-4578
- SEX: FEMALE  HT:5'2"   WT:120 LBS. - HIRE DATE     : 1971-07-28
- MARITAL STATUS: MARRIED

EDUCATION:
1965 MATH AND ENGLISH B.A.          1960 DENTAL TECHNICIAN
     ADELPHI UNIVERSITY                    FLORIDA INSTITUTE OF TECHNOLOGY

WORK HISTORY:
10/91 - PRESENT ADVISORY SYSTEMS ANALYST
                PRODUCING DOCUMENTATION TOOLS FOR ENGINEERING DEPARTMENT
12/85 - 9/91    TECHNICAL WRITER
                WRITER, TEXT PROGRAMMER, AND PLANNER
 1/79 - 11/85   COBOL PAYROLL PROGRAMMER
                WRITING PAYROLL PROGRAMS FOR A DIESEL FUEL COMPANY
```

Figure 116. DB2 Employee Resume Application

This panel is designed around the sample data. It is assumed that all resume information for an employee will fit predictably and no handling for special cases is provided. The "Interests" section of the resume is not presented because of space constraints on the panel.

Related tasks:

"Starting an application in an ISPF/TSO environment" on page 768

Using the photo application in the LOB application scenario

You can view employee photos with the photo application in the LOB application scenario.

About this task

The LOB application begins with the initial panel of the ISPF/CAF application. This LOB application requires that you include the GDDM load module library (SADMMOD) in your logon procedure or in the ISPLLIB concatenation.

Procedure

1. From the DB2 Sample Application Menu, DSN8SSM, select one of the following options to look at employee photos through GDDM:
   - If you ran job DSNTEJ75, select option 5.
   - If you ran job DSNTEJ78, select option 8.

   DSN8DLPV and DSN8CLPV show the following panel for selecting and viewing sample employee photos until you press the END key to exit ISPF.

```
DSN8SSE                  DB2 EMPLOYEE SELECTION PANEL
===>_



SELECT ONE OF THE EMPLOYEES AND PRESS ENTER.

    1. 000130 - DELORES M. QUINTANA
    2. 000130 - HEATHER A. NICHOLLS
    3. 000130 - BRUCE ADAMSON
    4. 000130 - JAMES H. WALKER


PRESS:    END TO EXIT
```

Figure 117. DB2 Employee Selection Panel

2. Select from the list of available photos by that are displayed on ISPF panel DSN8SSE.

Related tasks:

"Starting an application in an ISPF/TSO environment" on page 768

Sample LOB table: EMP_PHOTO_RESUME

The LOB sample application uses the EMP_PHOTO_RESUME table. The sample jobs create, load, and manipulate the table.

Table 157. EMP_PHOTO_RESUME table

Column Name:	EMPNO	EMP_ROWID	PSEG_PHOTO	BMP_PHOTO	RESUME
Type:	CHAR(6) NOT NULL	ROWID	BLOB(500K)	BLOB(100K)	CLOB(5K)
Description:	Employee number	Row identifier	Employee photo (PSEG)	Employee photo (BMP)	Employee resume
Values:	000130	#####	Delores M. Quintana	Delores M. Quintana	Delores M. Quintana
Values:	000140	#####	Heather A Nicholls	Heather A Nicholls	Heather A Nicholls
Values:	000150	#####	Bruce Adamson	Bruce Adamson	Bruce Adamson
Values:	000190	#####	James H. Walker	James H. Walker	James H. Walker

Edit exit routine

The edit exit routine encodes the salary column to protect the confidentiality of the employee's salary.

The edit exit routine is prepared in Phase 1. It works with the employee table (DSN81010.EMP) and is written in assembler language.

The name of the edit exit routine is DSN8EAE1. When the employee table (DSN81010.EMP) is changed by either an update or an add, the edit exit routine encodes the salary amount that goes into the SALARY column. When the SALARY column is read from the employee table, the amount is decoded. The encoding and decoding of the salary column protects the confidentiality of the employee's salary.

Recommendation: From V8 If you are migrating from DB2 Version 8 and you set the RRF subsystem parameter to ENABLE, use DSN8EAE9 rather than DSN8EAE1 as the name of the sample edit exit routine and module. Setting the RRF subsystem parameter to ENABLE specifies that newly created table spaces are to store data in reordered row format. If DB2 has already loaded the DSN8EAE1 edit exit routine for Version 8, sample job DSNTEJ1 might fail in job step PH01S13 with ABEND0C4 reason code 11. The failure occurs because the Version 8 DSN8EAE1 module supports only the basic row format. DSN8EAE1 is automatically changed to DSN8EAE9 in job DSNTEJ1 when you run the installation CLIST in ENFM mode. From V8

Huffman compression exit routine

IBM supplies a sample edit routine that compresses data using the Huffman algorithm.

Before using any data compression routine, understand its limitations and consider tailoring it to your particular table. For the restrictions and concerns that apply to the IBM sample, see the comments provided with the code. The routine is called DSN8HUFF and resides in library *prefix*.SDSNSAMP. IBM supplies a sample edit routine that compresses data using the Huffman algorithm (first described in *Proceedings of the IRE* September, 1952).

Sample field procedure

A sample field procedure is prepared in Phase 1 of the installation or migration verification process. This procedure causes values in a CHAR(6) column to be ordered in the ASCII sorting sequence.

Dynamic SQL statements: DSNTESA, DSNTESQ

prefix.SDSNSAMP library members DSNTESA and DSNTESQ contain dynamic SQL statements to help verify the success of an installation or migration.

DSNTESA

The SQL statements in DSNTESA are run dynamically by SPUFI. DSNTESA is used in Phase 3 of the verification process.

The first group of statements in DSNTESA create a temporary work file table space and defines a created temporary table. The INSERT statements fill the table with

names, midterm scores, and final examination results, and the SELECT statement then does a check of the averages. The UPDATE statements assign a grade according to the formula in the first UPDATE statement: 60% for the final and 40% for the midterm. The next SELECT statement produces the entire table. The ROLLBACK statement removes the table space and the table within it.

GUPI

The following statements make some administrative queries on the system tables:

- The following SELECT statements find all the plans and packages that are owned by the current user, and the date they were bound.

```
SELECT NAME, BINDDATE
    FROM SYSIBM.SYSPLAN
    WHERE CREATOR = USER;
```

```
SELECT COLLID, NAME, VERSION, BINDTIME
    FROM SYSIBM.SYSPACKAGE
    WHERE OWNER = USER;
```

- The following SELECT statements find the plans and packages that require a bind or rebind before they can be run, and the plans and packages that are automatically rebound the next time they are run.

```
SELECT NAME, CREATOR, BINDDATE, VALID, OPERATIVE
    FROM SYSIBM.SYSPLAN
    WHERE OPERATIVE = 'N' OR VALID = 'N';
```

```
SELECT COLLID, NAME, VERSION, BINDTIME, VALID
    FROM SYSIBM.SYSPACKAGE
    WHERE OPERATIVE = 'N' OR VALID = 'N';
```

- The following SELECT statements find all objects required for the current user's programs.

```
SELECT DNAME, BTYPE, BCREATOR, BNAME
    FROM SYSIBM.SYSPLANDEP
    WHERE BCREATOR = USER
    ORDER BY DNAME, BTYPE, BCREATOR, BNAME;
```

```
SELECT DCOLLID, DNAME, BTYPE, BQUALIFIER, BNAME
    FROM SYSIBM.SYSPACKDEP
    WHERE BQUALIFIER = USER
    ORDER BY DCOLLID, DNAME, BTYPE, BQUALIFIER, BNAME;
```

- The second SELECT from SYSTABLES provides information about all the DEPT tables regardless of the owner.

```
SELECT *
   FROM SYSIBM.SYSTABLES
   WHERE NAME = 'DEPT';
```

- The SELECT from SYSCOLUMNS supplies a description of the fields of the DSN81010.DEPT table. This information can also be provided by DCLGEN, and, within a program, the DESCRIBE statement gives this same information.

```
SELECT NAME, COLTYPE, LENGTH, SCALE, NULLS, REMARKS, COLNO
   FROM SYSIBM.SYSCOLUMNS
   WHERE TBNAME= 'DEPT' AND TBCREATOR = 'DSN8610'
   ORDER BY COLNO;
```

- The following SELECT statements find the kinds of authority a user can have. Determining which tables a specific user can access is relatively complicated because of the various authorities. If the user has SYSADM authority, any table can be accessed.

```
SELECT * FROM SYSIBM.SYSPLANAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSPACKAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSUSERAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSDBAUTH WHERE GRANTEE = USER;
```

```
SELECT * FROM SYSIBM.SYSTABAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSCOLAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSRESAUTH WHERE GRANTEE = USER;
```

- The final four SELECT statements show the tables and views that can be accessed directly by the current user, those that can be accessed using a plan, and those that are accessed using the database authority.

```
SELECT TCREATOR, TTNAME, STNAME, GRANTOR
   FROM SYSIBM.SYSTABAUTH
   WHERE GRANTEE = USER;
SELECT BNAME, BTYPE, GRANTOR, NAME
   FROM SYSIBM.SYSPLANAUTH, SYSIBM.SYSPLANDEP
   WHERE GRANTEE = USER
      AND NAME = DNAME
      AND EXECUTEAUTH ¬= ' '
      AND (BTYPE = 'T' OR BTYPE = 'V');
SELECT DCOLLID, BNAME, BTYPE, BQUALIFIER, BNAME
   FROM SYSIBM.SYSPACKAUTH, SYSIBM.SYSPACKDEP
      WHERE GRANTEE = USER
         AND COLLID = DCOLLID
         AND NAME = DNAME
         AND EXECUTEAUTH ¬= ' '
         AND (BTYPE = 'T' OR BTYPE = 'V');
SELECT NAME, CREATOR, TYPE, DBNAME, TSNAME
   FROM SYSIBM.SYSTABLES
      WHERE DBNAME IN
        (SELECT NAME FROM SYSIBM.SYSDBAUTH
           WHERE GRANTEE = USER
           AND DBADMAUTH ¬= ' ');
```

GUPI

DSNTESQ

DSNTESQ contains a set of queries to check consistency between catalog tables.

The SQL statements are in a format available for input to SPUFI and DSNTEP2. If SPUFI is not bound when you want to execute these queries, you can use DSNTEP2 from your previous DB2 version.

Before running these queries, you should run the DSN1CHKR utility to make sure the physical structure of the catalog is correct. You should also run the CHECK INDEX utility.

DSNTESQ contains SQL that creates copies of the catalog using segmented table spaces. In some cases, the queries in DSNTESQ run faster when run on copies of the catalog instead of the actual catalog because the copies have additional indexes. If you plan to use the copies of the catalog, use the comment lines in DSNTESQ for guidance.

Dynamic SQL programs: DSNTIAD, DSNTEP2, DSNTIAUL

DSNTIAD, DSNTEP2, and DSNTIAUL are sample dynamic SQL programs and must be compiled, link-edited, and bound as usual.

SPUFI is a part of the distributed product. An installation job is used to bind it. It can be used only with ISPF.

Related concepts:

DB2 sample applications (DB2 Application programming and SQL)

Chapter 8. Working with additional capabilities for DB2

After you complete the installation or migration process, you can work with additional capabilities of DB2 such as stored procedures and remote communication.

About this task

DB2-supplied stored procedures and user-defined functions

DB2 supplies a number of stored procedures and user defined functions that job DSNTIJRT defines and binds during installation or migration.

For DB2-supplied routines, job DSNTIJRW creates the core WLM environments that are listed in the following table. If you decide to set up your own WLM application environments, use the following table to help you group the DB2-supplied routines into the appropriate environments. If your system resources are constrained, you might need to choose a lower NUMTCB value than the recommended one.

DB2-supplied routine	Schema name	Recommended value of NUMTCB	Core WLM environment	Does the routine have to run in an APF-authorized environment?	Does the routine have to be program-controlled?
ADMIN_COMMAND_DB2	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_COMMAND_DSN	SYSPROC	1	DSNWLM_REXX	No	No
ADMIN_COMMAND_UNIX	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	No	Yes
ADMIN_DS_BROWSE	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_DS_DELETE	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_DS_LIST	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_DS_RENAME	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_DS_SEARCH	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_DS_WRITE	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_INFO_HOST	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_INFO_SMS	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
ADMIN_INFO_SQL	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_INFO_SSID	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_INFO_SYSLOG	SYSPROC	1	DSNWLM_REXX	No	No
ADMIN_INFO_SYSPARM	SYSPROC	1	DSNWLM_NUMTCB1	No	No
ADMIN_JOB_CANCEL	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	Yes	Yes
ADMIN_JOB_FETCH	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	Yes	Yes
ADMIN_JOB_QUERY	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	Yes	Yes
ADMIN_JOB_SUBMIT	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	No	Yes
ADMIN_TASK_ADD	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_CANCEL	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_LIST	DSNADM	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_OUTPUT	DSNADM	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_REMOVE	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_STATUS	DSNADM	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_TASK_UPDATE	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No

© Copyright IBM Corp. 1982, 2012

DB2-supplied routine	Schema name	Recommended value of NUMTCB	Core WLM environment	Does the routine have to run in an APF-authorized environment?	Does the routine have to be program-controlled?
ADMIN_UTL_EXECUTE	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	Yes	Yes
ADMIN_UTL_MODIFY	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_UTL_MONITOR	SYSPROC	40 - 60	DSNWLM_PGM_CONTROL	Yes	Yes
ADMIN_UTL_SCHEDULE	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ADMIN_UTL_SORT	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
ALTER_JAVA_PATH	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
CREATE_SESSION	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
DB2_INSTALL_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
DB2_REMOVE_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
DB2_REPLACE_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
DB2_UPDATEJARINFO	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
DBG_ENDSESSIONMANAGER	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_INITIALIZECLIENT	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_LOOKUPSESSIONMANAGER	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_PINGSESSIONMANAGER	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_RECVCLIENTREPORTS	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_RUNSESSIONMANAGER	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DBG_SENDCLIENTCOMMANDS	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_SENDCLIENTREQUESTS	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DBG_TERMINATECLIENT	SYSPROC	5 - 20	DSNWLM_DEBUGGER	No	No
DEBUGGERLEVEL	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
DESTROY_SESSION	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
DSN_WLM_APPLENV	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DSNACCOR	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DSNACCOX	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DSNACICS	SYSPROC	40	DSNWLM_CICS	Yes	No
DSNAEXP	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DSNAHVPM	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
DSNAIMS	SYSPROC	40	DSNWLM_GENERAL	Yes	No
DSNAIMS2	SYSPROC	40	DSNWLM_GENERAL	Yes	No
DSNTBIND	SYSPROC	1	DSNWLM_REXX	No	No
DSNLEUSR	SYSPROC	1	DSNWLM_NUMTCB1	Yes	No
DSNTPSMP	SYSPROC	1	DSNWLM_REXX	No[1]	No
DSNUTILS	SYSPROC	1	DSNWLM_UTILS	Yes	No
DSNUTILU	SYSPROC	1	DSNWLM_UTILS	Yes	No
DSNWSPM	SYSPROC	1	DSNWLM_NUMTCB1	Yes	No
DSNWZP	SYSPROC	1	DSNWLM_NUMTCB1	Yes	No
GET_CONFIG	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
GET_MESSAGE	SYSPROC	40 - 60	DSNWLM_GENERAL	No	No
GET_REPORT	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
GET_SYSTEM_INFO	SYSPROC	1	DSNWLM_NUMTCB1	Yes	No
INSTALL_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
LIST_SESSION	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
MQREAD	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQREADALL	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQREADALLCLOB	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQREADCLOB	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQRECEIVE	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No

DB2-supplied routine	Schema name	Recommended value of NUMTCB	Core WLM environment	Does the routine have to run in an APF-authorized environment?	Does the routine have to be program-controlled?
MQRECEIVEALL	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQRECEIVEALLCLOB	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQRECEIVECLOB	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
MQSEND	DB2MQ	≥ 10	DSNWLM_MQSERIES	No	No
PUT_COMMAND	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
QUERY_SESSION	DB2DEBUG	5 - 20	DSNWLM_DEBUGGER	No	No
REMOVE_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
REPLACE_JAR	SQLJ	40 - 60	DSNWLM_GENERAL	No	No
SOAPHTTPC	DB2XML	≥ 10	DSNWLM_WEBSERVICES	No	No
SOAPHTTPNC	DB2XML	≥ 10	DSNWLM_WEBSERVICES	No	No
SOAPHTTPNV	DB2XML	≥ 10	DSNWLM_WEBSERVICES	No	No
SOAPHTTPV	DB2XML	≥ 10	DSNWLM_WEBSERVICES	No	No
SQLCAMESSAGE	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLCOLPRIVILEGES	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLCOLUMNS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLFOREIGNKEYS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLFUNCTIONCOLS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLFUNCTIONS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLGETTYPEINFO	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLPRIMARYKEYS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLPROCEDURECOLS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLPROCEDURES	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLPSEUDOCOLUMNS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLSPECIALCOLUMNS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLSTATISTICS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLTABLEPRIVILEGES	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLTABLES	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SQLUDTS	SYSIBM	40 - 60	DSNWLM_GENERAL	No	No
SYSTS_ALTER	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_CREATE	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_DROP	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_ENCRYPT	SYSFUN	2 - 8	DSNWLM_JAVA	No	No
SYSTS_RESTORE	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_START	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_STOP	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_TAKEOVER	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
SYSTS_UPDATE	SYSPROC	10 - 40	DSNWLM_GENERAL	No	No
WLM_REFRESH	SYSPROC	40 - 60	DSNWLM_GENERAL	Yes	No
WLM_SET_CLIENT_INFO	SYSPROC	> 1	DSNWLM_GENERAL	No	No
XSR_ADDSCHEMADOC	SYSPROC	40 - 60	DSNWLM_XML	Yes	No
XSR_COMPLETE	SYSPROC	2 - 8	DSNWLM_JAVA	No	No
XSR_REGISTER	SYSPROC	40 - 60	DSNWLM_XML	Yes	No
XSR_REMOVE	SYSPROC	40 - 60	DSNWLM_XML	Yes	No

[1]This routine cannot run in an APF-authorized environment.

Routine setup information in DSNTID*xx*

To install the DB2-supplied routines, job DSNTIJRT uses setup information that is stored in the CLIST defaults input member (DSNTID*xx*). Each DB2-supplied routine is represented by a different name in DSNTID*xx*. The following table shows the mapping between each DSNTID*xx* name and DB2-supplied routine. The left column contains the DSNTID*xx* name and the right column contains the routine name. The *xxx* in the DSNTID*xx* names represent the letters that specify the WLM environment name, the grant execute list, and the package owner. For example, RTB01*xxx* represents RTB01WLM, RTB01ACL, and RTB01PKO.

Table 158. Mapping between DSNTIDxx variables and DB2-supplied routines

DSNTID*xx* variable	**Stored procedure name**
RTB01*xxx*	DSNWSPM
RTB02*xxx*	DSNWZP
RTC01*xxx*	DSNUTILS
RTC02*xxx*	DSNUTILU
RTD01*xxx*	DSNACCOR
RTD02*xxx*	DSNACCOX
RTE01*xxx*	DSNLEUSR
RTF01*xxx*	DSNTBIND
RTF02*xxx*	DSNTPSMP
RTF03*xxx*	WLM_REFRESH
RTG01*xxx*	ALTER_JAVA_PATH
RTG02*xxx*	DB2_INSTALL_JAR
RTG03*xxx*	DB2_REMOVE_JAR
RTG04*xxx*	DB2_REPLACE_JAR
RTG05*xxx*	DB2_UPDATEJARINFO
RTG06*xxx*	INSTALL_JAR
RTG07*xxx*	REMOVE_JAR
RTG08*xxx*	REPLACE_JAR
RTH01*xxx*	CREATE_SESSION
RTH02*xxx*	DEBUGGERLEVEL
RTH03*xxx*	DESTROY_SESSION
RTH04*xxx*	GET_REPORT
RTH05*xxx*	LIST_SESSION
RTH06*xxx*	PUT_COMMAND
RTH07*xxx*	QUERY_SESSION
RTH08*xxx*	DBG_ENDSESSIONMANAGER
RTH09*xxx*	DBG_INITIALIZECLIENT
RTH10*xxx*	DBG_LOOKUPSESSIONMANAGER
RTH11*xxx*	DBG_PINGSESSIONMANAGER
RTH12*xxx*	DBG_RECVCLIENTREPORTS
RTH13*xxx*	DBG_RUNSESSIONMANAGER
RTH14*xxx*	DBG_SENDCLIENTCOMMANDS

Table 158. Mapping between DSNTIDxx variables and DB2-supplied routines (continued)

DSNTID*xx* variable	**Stored procedure name**
RTH15*xxx*	DBG_SENDCLIENTREQUESTS
RTH16*xxx*	DBG_TERMINATECLIENT
RTI04*xxx*	XSR_ADDSCHEMADOC
RTI05*xxx*	XSR_COMPLETE
RTI06*xxx*	XSR_REGISTER
RTI07*xxx*	XSR_REMOVE
RTJ01*xxx*	SYSTS_CREATE
RTJ02*xxx*	SYSTS_DROP
RTJ03*xxx*	SYSTS_ENCRYPT
RTJ04*xxx*	SYSTS_RESTORE
RTJ05*xxx*	SYSTS_START
RTJ06*xxx*	SYSTS_STOP
RTJ07*xxx*	SYSTS_TAKEOVER
RTJ08*xxx*	SYSTS_UPDATE
RTJ09*xxx*	SYSTS_ALTER
RTK01*xxx*	SQLCAMESSAGE
RTK02*xxx*	SQLCOLPRIVILEGES
RTK03*xxx*	SQLCOLUMNS
RTK04*xxx*	SQLFOREIGNKEYS
RTK05*xxx*	SQLFUNCTIONCOLS
RTK06*xxx*	SQLFUNCTIONS
RTK07*xxx*	SQLGETTYPEINFO
RTK08*xxx*	SQLPRIMARYKEYS
RTK09*xxx*	SQLPROCEDURECOLS
RTK10*xxx*	SQLPROCEDURES
RTK11*xxx*	SQLSPECIALCOLUMNS
RTK12*xxx*	SQLSTATISTICS
RTK13*xxx*	SQLTABLEPRIVILEGES
RTK14*xxx*	SQLTABLES
RTK15*xxx*	SQLUDTS
RTL01*xxx*	ADMIN_COMMAND_DB2
RTL02*xxx*	ADMIN_COMMAND_DSN
RTL03*xxx*	ADMIN_COMMAND_UNIX
RTL04*xxx*	ADMIN_DS_BROWSE
RTL05*xxx*	ADMIN_DS_DELETE
RTL06*xxx*	ADMIN_DS_LIST
RTL07*xxx*	ADMIN_DS_RENAME
RTL08*xxx*	ADMIN_DS_SEARCH
RTL09*xxx*	ADMIN_DS_WRITE
RTL10*xxx*	ADMIN_INFO_HOST

Table 158. Mapping between DSNTIDxx variables and DB2-supplied routines (continued)

DSNTID*xx* variable	**Stored procedure name**
RTL11*xxx*	ADMIN_INFO_SSID
RTL12*xxx*	ADMIN_JOB_CANCEL
RTL13*xxx*	ADMIN_JOB_FETCH
RTL14*xxx*	ADMIN_JOB_QUERY
RTL15*xxx*	ADMIN_JOB_SUBMIT
RTL17*xxx*	ADMIN_TASK_ADD
RTL18*xxx*	ADMIN_TASK_LIST
RTL19*xxx*	ADMIN_TASK_REMOVE
RTL20*xxx*	ADMIN_TASK_STATUS
RTL21*xxx*	ADMIN_UTL_SCHEDULE
RTL22*xxx*	ADMIN_UTL_SORT
RTL23*xxx*	ADMIN_TASK_CANCEL
RTL24*xxx*	ADMIN_TASK_OUTPUT
RTL25*xxx*	GET_CONFIG
RTL26*xxx*	GET_MESSAGE
RTL27*xxx*	GET_SYSTEM_INFO
RTL28*xxx*	ADMIN_INFO_SYSPARM
RTL29*xxx*	WLM_SET_CLIENT_INFO
RTL30*xxx*	ADMIN_UTL_EXECUTE
RTL31*xxx*	ADMIN_UTL_MODIFY
RTL32*xxx*	ADMIN_UTL_MONITOR
RTL33*xxx*	DSN_WLM_APPLENV
RTL34*xxx*	ADMIN_TASK_UPDATE
RTL35*xxx*	ADMIN_INFO_SMS
RTL36*xxx*	ADMIN_INFO_SQL
RTL37*xxx*	ADMIN_INFO_SYSLOG
RTM01*xxx*	DSNAEXP
RTM05*xxx*	DSNAHVPM
RTN01*xxx*	DSNACICS
RTN02*xxx*	DSNAIMS
RTN03*xxx*	DSNAIMS2
RTO02*xxx*	MQREAD
RTO03*xxx*	MQREADALL
RTO04*xxx*	MQREADALLCLOB
RTO05*xxx*	MQREADCLOB
RTO06*xxx*	MQRECEIVE
RTO07*xxx*	MQRECEIVEALL
RTO08*xxx*	MQRECEIVEALLCLOB
RTO09*xxx*	MQRECEIVECLOB
RTO10*xxx*	MQSEND

Table 158. Mapping between DSNTIDxx variables and DB2-supplied routines (continued)

DSNTID*xx* variable	**Stored procedure name**
RTP01*xxx*	SOAPHTTPC
RTP02*xxx*	SOAPHTTPNC
RTP03*xxx*	SOAPHTTPNV
RTP04*xxx*	SOAPHTTPV

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

WLM management of stored procedures

Workload Manager (WLM) is a component of z/OS that manages incoming work requests for the operating system and allocates available system resources to meet these requests. DB2 for z/OS uses WLM to allocate workload requests for DB2 stored procedures.

When DB2 receives a request to run a stored procedure, WLM determines whether additional resources are needed to process the request. For example, WLM might determine that a new WLM-managed address space is needed. WLM makes this determination by evaluating whether the work for a service class is meeting the goals that you set. WLM also evaluates whether the work queues are becoming too large. WLM then examines the system resource utilization as part of the algorithm to decide if another address space should be started. In many cases, a sudden increase in incoming work is handled by existing WLM address spaces, and task control blocks (TCBs) remove work from the queues. Consequently, an address space can handle more concurrent stored procedures or user-defined functions than the number that is specified by NUMTCB in the application environment definition.

Requirement: All DB2 stored procedures (except native SQL procedures) must run in WLM-managed address spaces.

A *WLM application environment* starts and manages WLM address spaces that are to be used by stored procedures. You can either create your own WLM application environments or use existing ones, and each stored procedure must be assigned to a WLM application environment. You can assign more than one stored procedure to each WLM application environment, but you might want to separate them for various resource management reasons. The stored procedures that are assigned to a WLM application environment do not need to be written in the same high-level language. However, for different performance reasons, you might decide to separate stored procedures into different application environments by language.

You define application environments for stored procedures as part of your WLM service policy. You also need to define a JCL procedure that WLM is to use to start the address spaces. You store this procedure in PROCLIB. The application environment definition in your policy contains the name of this startup procedure in PROCLIB. If too few address spaces are available to meet the policy goal, WLM starts another server address space if the service policy definition allows it.

Related concepts:

MVS Planning: Workload Management

General performance considerations (DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond)

Related tasks:

"Setting up a WLM application environment for stored procedures during installation" on page 543

"Setting up a WLM application environment for stored procedures during migration" on page 617

Setting up and managing Workload Manager (DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond)

Setting the priority of stored procedures (DB2 Administration Guide)

Core WLM environments for DB2-supplied routines

Definitions for core WLM environments are provided as recommended initial environments for running DB2-supplied routines. You can use DB2 core WLM environments to model additional WLM environments as you tune your subsystem for maximum performance of the DB2-supplied routines that you use.

Use job DSNTIJRW to define each of these environments with the parameters below, including any changes that are applicable to your site. Alternatively, you can use the provided IWMARIN0 ISPF tool to define these environments. These DB2 core WLM environments are intended as a starting point, not as the optimal environments in which to run the DB2-supplied routines. Over time, you will identify routines for your site that function best in a separate WLM environment.

Installation job DSNTIJMV creates an address space procedure for each of the core WLM environments. The WLM environment address space names have the form *ssnm*WLM*suffix*, where *ssnm* is the DB2 subsystem name specified on panel DSNTIPM in the SUBSYSTEM NAME field, and *suffix* is a character that differentiates the various WLM environments. Running DSNTIJMV adds these procedures to the system PROCLIB that you specified on panel DSNTIPW in the SYSTEM PROCEDURES field.

In the description of each environment below, the following terms are used:

NUMTCB
: The maximum number of procedures or functions that can run concurrently in the WLM-established stored procedure address space. The NUMTCB setting for the environment is the maximum appropriate setting for at least one of the routines that should be run in that environment. Other routines that run in the environment may tolerate a higher NUMTCB setting.

APF-authorized
: Whether the external module for one or more routines that are recommended to run in the environment must run under control of the z/OS authorized program facility (APF). "Yes" means that all libraries allocated by the STEPLIB DD must be APF-authorized. "No" means that at least one library allocated by the STEPLIB DD is not APF-authorized. Not all routines that are recommended to run in an APF-authorized environment necessarily require APF authorization. DSNTPSMP must run in an environment that is not APF-authorized.

Special DDs

Any special data sets, libraries, or work files that are required by at least one of the routines that are recommended to run in the environment. These are in addition to the standard requirement of STEPLIB DD.

WLM environments

DSNWLM_DEBUGGER

This environment is appropriate for stored procedures that are provided for the Unified Debugger tool. Job DSNTIJMV creates an address space procedure for DSNWLM_DEBUGGER that is called DSNWLMD. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMD.

DSNWLM_DEBUGGER has the following characteristics:

NUMTCB	10
APF-authorized	No
Special DDs:	This DD is optional: • PSMDEBUG which is commented out in normal use. Uncomment it to activate debug tracing.
Other considerations:	• This environment might have a high refresh rate when debug diagnostics are activated. • The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_DSNACICS

This environment is appropriate for DSNACICS, the DB2 CICS transaction invocation stored procedure. Job DSNTIJMV creates an address space procedure for DSNWLM_DSNACICS that is called DSNWLMC. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMC.

DSNWLM_DSNACICS has the following characteristics:

NUMTCB	40
APF-authorized	Yes
Special DDs:	This DD is optional: • DSNDUMP, which is commented out in normal use. Uncomment it to cause DSNACICS to generate an SVC dump when DSNACICS will generate an error message. When this DD is used, allocate it to output class for system dumps.
Other considerations:	• The CICS SDFHEXCI library must be allocated by the STEPLIB DD. • The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_GENERAL

This environment is appropriate for most DB2-supplied routines. Job DSNTIJMV creates an address space procedure for DSNWLM_GENERAL that is called DSNWLMG. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMG.

DSNWLM_GENERAL has the following characteristics:

NUMTCB	40
APF-authorized	Yes
Special DDs:	None
Other considerations:	The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_JAVA

This environment is appropriate for most DB2-supplied Java routines. Job DSNTIJMV creates an address space procedure for DSNWLM_JAVA that is called DSNWLMJ. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMJ.

DSNWLM_JAVA has the following characteristics:

NUMTCB	5
APF-authorized	No

Special DDs:	These DDs are required:
	• JAVAENV allocates a data set that contains Language Environment run-time options for Java stored procedures. The presence of this DD statement indicates to DB2 that the WLM environment is for Java routines. For an interpreted Java routine, this data set must contain the environment variable JAVA_HOME. This environment variable indicates to DB2 that the WLM environment is for interpreted Java routines. JAVA_HOME also specifies the highest-level directory in the set of directories that contain the Java SDK.
	• JSPDEBUG allocates a data set in which Language Environment puts run-time diagnostics.
	Example:
	`//JAVAENV DD DISP=SHR,DSN=WLMIJAV.JSPENV`
	`//JSPDEBUG DD SYSOUT=*`

Other considerations:	The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_MQSERIES

This environment is appropriate for running DB2 MQSeries functions. Job DSNTIJMV creates an address space procedure for DSNWLM_MQSERIES that is called DSNWLMM. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMM.

DSNWLM_MQSERIES has the following characteristics:

NUMTCB	10
APF-authorized	No
Special DDs:	None

Other considerations:

- MQSeries runtime libraries must be allocated by the STEPLIB DD if they are not included in the system link list. Example:

```
//STEPLIB DD DISP=SHR,DSN=prefix.RUNLIB.LOAD
//        DD DISP=SHR,DSN=prefix.SCEERUN
//        DD DISP=SHR,DSN=prefix.SDSNEXIT
//        DD DISP=SHR,DSN=prefix.SDSNLOAD
//* Uncomment the following if not included in the LINKLIST
//*       DD DISP=SHR,DSN=prefix.SCSQLOAD
//*       DD DISP=SHR,DSN=prefix.SCSQAUTH
//*       DD DISP=SHR,DSN=prefix.SCSQANL
```

- The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_NUMTCB1

This environment is for DB2-supplied routines that must run serially. If a routine must run serially, it cannot share a task control block (TCB) with another routine. Routines that should run in this environment require a maximum TCB setting of 1 but have no other special DD requirements. Job DSNTIJMV creates an address space procedure for DSNWLM_NUMTCB1 that is called DSNWLM1. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLM1.

DSNWLM_NUMTCB1 has the following characteristics:

NUMTCB	1
APF-authorized	Yes
Special DDs:	None
Other considerations:	The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_PGM_CONTROL

This environment is appropriate for DB2-supplied routines that are registered to RACF program control or an equivalent and can therefore be loaded only with other similar routines. Job DSNTIJMV creates an address space procedure for DSNWLM_PGM_CONTROL that is called DSNWLMP. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMP.

DSNWLM_PGM_CONTROL has the following characteristics:

NUMTCB	40
APF-authorized	Yes
Special DDs:	None
Other considerations:	The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_REXX

This environment is primarily appropriate for the DB2 SQL Procedure Processor routine, DSNTPSMP. However, it can be shared with the other DB2-supplied REXX stored procedures, DSNTBIND and ADMIN_COMMAND_DSN. Job DSNTIJMV creates an address space procedure for DSNWLM_REXX that is called DSNWLMR. The name and

library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMR.

DSNWLM_REXX has the following characteristics:

NUMTCB	1
APF-authorized	No

Special DDs: These DDs are required:

- SYSEXEC allocates the *prefix*.SDSNCLST library. This data set is where the DB2-supplied REXX routines reside.
- SYSTSPRT allocates an output data destination for messages from the TSO/E command processor.
- SQLDBRM allocates the data set for DBRMs created by running the DSNTPSMP stored procedure, and for DBRMs of packages that are bound by running the DSNTPSMP and DSNTBIND stored procedures.
- SQLCSRC allocates the data set for storing generated C code created by running DSNTPSMP.
- SQLLMOD allocates the data set for storing load modules created by running DSNTPSMP.
- SQLLIBC allocates C header files used by DSNTPSMP for compiling SQL procedures.
- SQLLIBL allocates include files used by DSNTPSMP for link editing external modules for SQL procedures.
- SYSMSGS allocates the message file for the IBM Language Environment pre-link editor.
- CFGTPSMP (optional) allocates a site-provided sequential dataset or member, used to define customized operation of DSNTPSMP in this application environment.
- SQLSRC allocates a work file that is used by DSNTPSMP. This work file needs to have a fixed block record format, a logical record length of 80, and a block size of 23440.
- SQLPRINT allocates a work file that is used by DSNTPSMP. This work file needs to have a variable block record format, a logical record length of 137, and a block size of 23476.
- SQLTERM allocates a work file that is used by DSNTPSMP. This work file needs to have a variable block record format, a logical record length of 137, and a block size of 23476.
- SQLOUT allocates a work file that is used by DSNTPSMP. This work file needs to have a variable block record format, a logical record length of 137, and a block size of 23476.
- SQLCPRT allocates a work file that is used by DSNTPSMP. This work file needs to have a variable block record format, a logical record length of 137, and a block size of 23476.
- SQLUT1 allocates a work file that is used by DSNTPSMP. This work file needs to have a fixed block record format, a logical record length of 80, and a block size of 23440.
- SQLUT2 allocates a work file that is used by DSNTPSMP. This work file needs to have a fixed block record format, a logical record length of 80, and a block size of 23440.
- SQLCIN allocates a work file that is used by DSNTPSMP. Do not specify a DCB for allocating this work file.
- SQLLIN allocates a work file that is used by DSNTPSMP. This work file needs to have a fixed block record format, a logical record length of 80, and a block size of 3200.
- SYSMOD allocates a work file that is used by DSNTPSMP. This work file needs to have a fixed block record format, a logical record length of 80, and a block size of 23440.
- SQLDUMMY is required, but must be allocated to DUMMY.

Other considerations:

- The TSO/E DYNAMNBR parameter must indicate the number of system resources the system should hold for possible reuse.
- The STEPLIB DD must allocate at least one library that is not APF-authorized.
- The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_UTILS

This environment is appropriate for DB2 utilities stored procedures, which require special DD allocations. These routines require a maximum NUMTCB setting of 1. Job DSNTIJMV creates an address space procedure for DSNWLM_UTILS that is called DSNWLMU. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMU.

DSNWLM_UTILS has the following characteristics:

NUMTCB
: 1

APF-authorized
: Yes

Special DDs:
: These DDs are required:

- UTPRINT allocates a data set for messages from the sort program.
- RNPRIN01 allocates a data set for messages from the sort program. This DD is required only if you plan to invoke RUNSTATS and collect distribution statistics.
- STPRIN01 allocates a data set for messages from the sort program. This DD is required when a utility is directed to collect frequency statistics on at least one data-partitioned secondary index or when TABLESPACE TABLE COLGROUP FREQVAL is specified.
- DSSPRINT allocates a data set for messages when making concurrent copies.
- SYSIN allocates a work file for temporarily storing utility input statements
- SYSPRINT allocates a work file for temporarily storing utility output messages.

Example:

```
//UTPRINT  DD SYSOUT=*
//RNPRIN01 DD SYSOUT=*
//STPRIN01 DD SYSOUT=*
//DSSPRINT DD SYSOUT=*
//SYSIN DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
//SYSPRINT DD UNIT=SYSDA,SPACE=(4000,(20,20),,,ROUND)
```

Other considerations:
: The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

DSNWLM_WEBSERVICES

This environment is appropriate for running DB2 web services and SOAP functions. Job DSNTIJMV creates an address space procedure for DSNWLM_WEBSERVICES that is called DSNWLMW. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMW.

DSNWLM_WEBSERVICES has the following characteristics:

NUMTCB	10
APF-authorized	No

Special DDs:	This DD is required:
	• WSERROR allocates an HFS file for diagnostic output from the DB2 web services functions.
	Example:
	`//WSERROR DD PATH='/tmp/wsc.err',`
	`//  PATHOPTS=(ORDWR,OCREAT,OAPPEND),`
	`//  PATHMODE=(SIRUSR,SIWUSR,SIRGRP,`
	`//  SIWGRP,SIROTH,SIWOTH)`
	Ensure that "PATH" specifies a valid path name and that the specified directory structure exists.

Other considerations:	• The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside. • The default timeout to read data from WS provider is 2 seconds. If you need to increase the timeout value, redefine the UDFs with the following runtime options: `RUN OPTIONS 'POSIX(ON),XPLINK(ON),ENVAR("DB2SOAP_TIMEOUT=X")'` where *X* is a number that specifies the timeout value in seconds.

DSNWLM_XML

This environment is appropriate for DB2-supplied XML processing routines that require large amounts of memory. Job DSNTIJMV creates an address space procedure for DSNWLM_XML that is called DSNWLMX. The name and library name of this procedure change according to the DB2 subsystem name that is specified on panel DSNTIPM. For example, if the subsystem name is VA1A, the procedure is named VA1AWLMX.

DSNWLM_XML has the following characteristics:

NUMTCB	40
APF-authorized	Yes
Special DDs:	None
Other considerations:	The STEPLIB DD must include DSN=*prefix*.SDSNEXIT or other library where the authorization exit modules (DSN3@ATH and DSN3@SGN) reside.

Related concepts:

"Configuration control statements for DB2-supplied routines" on page 849

Related tasks:

"Setting up a WLM application environment for stored procedures during installation" on page 543

"Setting up a WLM application environment for stored procedures during migration" on page 617

DSNTWLMB tool

The DSNTWLMB tool provides a batch interface for defining, installing, and activating WLM environments. This tool is invoked by job DSNTIJRW to create the core DB2 WLM environments.

Authorization

If the RACF facility class is active and a profile has been defined for the MVSADMIN.WLM.POLICY facility class, the user of this tool needs the following access:

- ACCESS(READ) for action WLMINFO

  ```
  PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(user) ACCESS(READ)
  ```

- ACCESS(UPDATE) for action ADD, ADD_ACTIVATE, ACTIVATE

  ```
  PERMIT MVSADMIN.WLM.POLICY CLASS(FACILITY) ID(user) ACCESS(UPDATE)
  ```

Parameter descriptions

The batch program accepts the following parameters:

ACTION(*action-type*)
Identifies the type of action that is to be performed. This parameter is required and it supports the following values:

ADD
The batch program installs a new WLM application environment to an existing WLM service definition without activating a service policy. The new WLM application environment becomes available on the next activation of a WLM policy.

ADD_ACTIVATE
The batch program installs a new WLM application environment to an existing WLM service definition and activates a WLM service policy to enable the new WLM application environment.

ACTIVATE
The batch program activates a WLM service policy.

WLMINFO
The batch program queries the existing WLM service definition and returns basic information.

POLICY(*policy-ID*)
Identifies the WLM service policy that is to be activated. This parameter is required if the specified action is ADD_ACTIVATE or ACTIVATE. The service policy ID can be 1 – 8 characters or “ACTIVE”. If “ACTIVE” is specified as the policy ID, the current active service policy is used.

SYSIN parameters

If the specified action is ADD or ADD_ACTIVATE, additional input is required from SYSIN to provide details about the WLM environment that is to be installed. The additional information can be entered from the SYSIN DD statement. The parameters can be entered on one line (up to 72 characters per line) or multiple lines, and they should be separated by one or more spaces. The following parameters are supported under SYSIN DD:

WLMNAME(*name*)
The defined name for an application environment. The name can be 1 – 32 characters and cannot begin with “SYS”. This is a required parameter.

DESCRIPTION(*description*)
A 32 character area that describes the application environment. This parameter can be set to an empty value. For example:

```
DESCRIPTION()
```

PROCNAME(*procedure-name*)
Defines the JCL procedure that WLM uses to start server address spaces for the application environment. This is a required parameter.

STARTPARM(*start-up-parameters*)
Contains the parameters that WLM uses to start the JCL procedure. The parameters can be up to 115 characters, or it can be set to an empty value. If the parameters include the subsystem name, you can use the symbol &IWMSSNM to cause WLM to substitute the subsystem name instead of directly typing the subsystem name. This is useful because multiple instances of the subsystem with different names can use the application environment. For example:

```
STARTPARM(DB2SSN=&IWMSSNM,APPLENV=WLMENV1,NUMTCB=1)
```

WLMOPT(WLM_MANAGED|SINGLE_SERVER)
Tells WLM whether it should limit the number of server address spaces that are created. For a single system, the WLM_MANAGED option means that any number of servers can be created for the subsystem on the system where it is connected to Workload Manager. In a sysplex, WLM_MANAGED means that servers can be created for the subsystem on any number of systems in the sysplex. If multiple DB2 subsystems are on a system, WLM will create one server for each DB2 subsystem that calls the routines.

The SINGLE_SERVER option means that only one address space can exist per system. For DB2-supplied routines, you can set a limit of one address space per system if this limit is required by the routine. For example, if a server address space requires exclusive use of a resource, only a single address space can exist. Use this option if a stored procedure cannot execute concurrently in multiple address spaces. This option should be used if a stored procedure is to be run in debug mode and writes to a trace. A limit of one address space per sysplex does not apply to DB2-supplied routines.

This parameter can be set to an empty value and will default to WLM_MANAGED.

Sample SQL statements for installing DB2-supplied routines

Sample member DSNTESR contains the full template data definition language, which is used by job DSNTIJRT. DSNTIJRT uses the full template data definition language to create DB2-supplied routines and supporting objects, and bind statements for creating related packages.

In earlier versions of DB2, the data definition language for creating and granting access to DB2-supplied routines and related objects, and the commands for binding packages, was provided as in-stream input processed by a number of installation jobs. In Version 10, those statements are removed, but DSNTESR enables users to view the full SQL statements if necessary. In DSNTESR, if a routine has a package, the command to bind that package appears immediately after the CREATE statement from that routine.

Configuration control statements for DB2-supplied routines

Configuration control statements supply the basic input for the DSNTRIN program, which installs and configures all routines that DB2 supplies.

Configuration control statements consist of a routine's schema and name (for stored procedures) or specific name (for user-defined functions), followed by customizable parameters. You can manually customize these parameters, but to

save your settings and facilitate future migrations, specify changes using the DB2 routines: Advanced configuration options installation panels, which are accessed from the DSNTIPR1 panel.

Parameters

The following parameters are used in configuration control statements:

WLMENV
The name of the WLM environment for running the routine. The default environment depends on the routine.

GRANTTO
The authorization IDs allowed to execute the routine and use any associated tables or created global temporary tables. Separate multiple IDs with commas. Do not use blanks. The default is PUBLIC, which allows everyone to execute the routine.

PKGOWNER
The authorization ID to own the routine's package. The specified owner must have the privileges required to execute the SQL statements contained in the routine. If this parameter is specified for a routine that does not have a package, the parameter is ignored. If this optional parameter is not specified, the default package owner is the primary ID used to run DSNTRIN.

Statement format

The general form of a configuration control statement is:

```
schema.routine-name
 WLMENV(wlm-environment-name)
 GRANTTO(auth-ID1,...,auth-IDn)
 PKGOWNER(package-owner)
```

Statement rules

The following rules apply to configuration control statements:

- Statements must reside in records having a fixed-block record format with a logical record length of 80 (same as JCL).
- Only columns 1 - 72 of each record are inspected. Columns 73 - 80 are ignored.
- Statement elements are delimited by one or more blank characters.
- Statement elements cannot contain embedded blanks.
- Statements are delimited by one or more blank lines.
- A statement can be continued on multiple input records. A statement element can be continued by breaking it at column 72 of one record and resuming it at column 1 of the next record.
- An asterisk in column 1 of a record indicates a comment line and the content is ignored.
- Comment lines cannot begin after column 1.
- Comments and statement elements cannot be on the same line.
- Comment lines can be embedded within a statement.
- The routine must be specified as the schema name followed by a period and the routine name.
- The WLMENV parameter is required and must directly follow the *schema.routine name*.

- The GRANTTO parameter is required and must directly follow the WLMENV parameter.
- The PKGOWNER parameter is optional. If specified, it must follow the GRANTTO parameter.

Examples

The following examples are valid configuration control statements:

```
* Configuration statements for the Utilities stored procedures follow:
SYSPROC.DSNUTILS
WLMENV(WLMENV_DSNUTIL)
*
* A comment line within the statement is legal
*
  GRANTTO(PUBLIC)
  PKGOWNER(ALFA)

* Notice the all-blank line above to delineate the two statements
SYSPROC.DSNUTILU
  WLMENV(WLMENV_DSNUTIL)
*
* Here is an example of wrapping an element to the next line
  GRANTTO(ALFA,BRAVO,CHARLIE,DELTA,ECHO,FOXTROT,GOLF,HOTEL,INDIA,JULIETT
E,KILO,LIMA)
*
* PKGOWNER is commented-out below so the OWNER parm will be excluded
*          from the BIND PACKAGE statement
* PKGOWNER(ALFA)
```

The following examples are invalid configuration control statements that violate the statement rules:

```
* Configuration statement for the WLM refresh stored procedure follows:
* - Error 1: The statement must start with schema.routine-name
WLM_REFRESH
* - Error 2: Comments can't start after column 1
 * This is not a valid comment
* - Error 3: Comments and elements can't be in the same line
  WLMENV(WLMENV_APFAUTH) * This is an invalid place to put a comment
* - Error 4: GRANTTO argument contains blanks
  GRANTTO( ALFA, BRAVO, CHARLIE, DELTA, ECHO, FOXTROT )
* - Error 5: PKGOWNER is present but has no argument
  PKGOWNER()
*
* - Error 6: The statements are not separated by a blank line
SYSPROC.DSNTPSMP
* - Error 7: The statement does not have a WLMENV parm
* - Error 8: The statement contains a blank line

  GRANTTO(HOTEL)
```

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Disabling DB2-supplied routines for validation

Job DSNTIJRT installs all DB2-supplied routines on your subsystem. After installation or migration, you can disable any routines that are unnecessary or unusable so the validation program can exclude it from processing.

Procedure

To disable a DB2-supplied routine:

Assign DSNWLM_NOT_ENABLED as the WLM environment of the routine. The DSNTRVFY validation program does not validate routines that are defined with DSNWLM_NOT_ENABLED as the WLM environment name.

Example

For example, if CICS is not installed, assign DSNWLM_NOT_ENABLED as the WLM environment of SYSPROC.DSNACICS by using the following statement:

GUPI

```
ALTER PROCEDURE SYSPROC.DSNACICS WLM ENVIRONMENT DSNWLM_NOT_ENABLED
```

GUPI

Setting up support for external SQL procedures

If you plan to create external SQL procedures by using DSNTPSMP or IBM Optim Development Studio, you need to complete some additional setup. Alternatively, you can write your own JCL to create external SQL procedures, in which case you do not need to perform this setup.

About this task

DSNTPSMP is a REXX stored procedure that you can use to create and prepare external SQL procedures. DSNTPSMP is also used by IBM Optim Development Studio to create external SQL procedures. When DSNTPSMP is invoked, DB2 uses the DSNTPSMP REXX EXEC, which is in the *prefix*.SDSNCLST data set.

Procedure

To set up support for external SQL procedures:

1. If DSNTPSMP does not already exist, run job DSNTIJRT to create this stored procedure.
2. Install DB2 REXX Language Support.
3. Set up a WLM environment in which to run DSNTPSMP and create a WLM startup procedure.

   Recommendation: If you do not have a WLM environment defined to run DSNTPSMP, use the core WLM environment DSNWLM_REXX.

Related concepts:

"Job DSNTIJRT" on page 549

Related tasks:

"Installation step 16: Create default storage group, define temporary work files, and bind DB2 REXX Language Support: DSNTIJTM" on page 532

"Setting up a WLM application environment for stored procedures during installation" on page 543

"Setting up a WLM application environment for stored procedures during migration" on page 617

Creating an external SQL procedure (DB2 Application programming and SQL)

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Additional steps for enabling the stored procedures and objects for XML schema support

DB2 for z/OS provides several stored procedures that you can call in your application programs to perform XML schema registration and removal.

Before you begin

GUPI Before you complete this step, set up WLM application environments for the stored procedures for XML support, and either install the DB2-supplied routines during migration or install the DB2-supplied routines during installation.

About this task

DB2 provides the following stored procedures for XML schema support:

- SYSPROC.XSR_REGISTER
- SYSPROC.XSR_ADDSCHEMADOC
- SYSPROC.XSR_COMPLETE
- SYSPROC.XSR_REMOVE

DB2 provides the following tables for XML schema support:

- SYSIBM.XSRANNOTATIONINFO
- SYSIBM.XSRCOMPONENT
- SYSIBM.XSROBJECTCOMPONENTS
- SYSIBM.XSROBJECTHIERARCHIES
- SYSIBM.XSROBJECTGRAMMAR
- SYSIBM.XSROBJECTPROPERTY
- SYSIBM.XSROBJECTS
- SYSIBM.XSRPROPERTY

Procedure

To enable the stored procedures and objects for XML schema support:

Install Java support.
The XSR_COMPLETE stored procedure is a Java stored procedure and therefore requires Java support.

Related concepts:

"Support for Java" on page 561

"Job DSNTIJRT" on page 549

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

"XML schema processing routines panel: DSNTIPRI" on page 471

Related information:

DB2-supplied stored procedures for XML schema registration and removal (DB2 Programming for XML)

Additional steps for enabling the administrative task scheduler and administrative enablement routines

The administrative task scheduler routines are DB2-supplied routines that enable you to schedule work. The administrative enablement routines are DB2-supplied routines that enable you to execute administrative work immediately.

Before you begin

Before you complete these steps, set up WLM application environments for the administrative task scheduler and administrative enablement routines, and either install the DB2-supplied routines during migration or install the DB2-supplied routines during installation.

About this task

The following routines are administrative task scheduler routines:

- ADMIN_TASK_ADD
- ADMIN_TASK_CANCEL
- ADMIN_TASK_LIST
- ADMIN_TASK_OUTPUT
- ADMIN_TASK_REMOVE
- ADMIN_TASK_STATUS
- ADMIN_TASK_UPDATE

The following routines are administrative enablement routines:

- ADMIN_COMMAND_DB2
- ADMIN_COMMAND_DSN
- ADMIN_COMMAND_UNIX
- ADMIN_DS_BROWSE
- ADMIN_DS_DELETE
- ADMIN_DS_LIST
- ADMIN_DS_RENAME
- ADMIN_DS_SEARCH
- ADMIN_DS_WRITE
- ADMIN_INFO_HOST
- ADMIN_INFO_SMS
- ADMIN_INFO_SQL
- ADMIN_INFO_SSID

- ADMIN_INFO_SYSLOG
- ADMIN_INFO_SYSPARM
- ADMIN_JOB_CANCEL
- ADMIN_JOB_FETCH
- ADMIN_JOB_QUERY
- ADMIN_JOB_SUBMIT
- ADMIN_UTL_EXECUTE
- ADMIN_UTL_MODIFY
- ADMIN_UTL_MONITOR
- ADMIN_UTL_SCHEDULE
- ADMIN_UTL_SORT

As part of the installation and migration process, job DSNTIJRT defines, binds, and grants access to these DB2–supplied routines.

Procedure

To enable the administrative task scheduler and administrative enablement routines:

1. In the JCL for starting the WLM-established address space for running the stored procedures whose load module must reside in an APF-authorized library, ensure that all libraries in the STEPLIB DD concatenation are APF-authorized.
2. If the BPX.DAEMON facility class is active and the BPX.DAEMON.HFSCTL facility class is not defined, perform the following actions for the stored procedures that must be registered to RACF program control. The stored procedures that require RACF program control are:
   - ADMIN_COMMAND_UNIX
   - ADMIN_JOB_CANCEL
   - ADMIN_JOB_FETCH
   - ADMIN_JOB_QUERY
   - ADMIN_JOB_SUBMIT

   a. Customize and run job step DSNADER in job DSNTIJRA to define the security environment for these stored procedures. You need to uncomment job step DSNADER before you run job DSNTIJRA. You do not need to run any other job steps in job DSNTIJRA.
   b. Ensure that you have a WLM environment for these stored procedures that meets the following requirements:
      - The WLM-established stored procedure address space loads only controlled programs.
      - In the JCL for starting the WLM-established address space for running these stored procedures, ensure that all libraries in the STEPLIB DD concatenation are APF-authorized.
      - The name of this environment must match the WLMENV parameter value in the configuration control statement for each stored procedure in job DSNTIJRT.

      Recommendation: If you do not have a WLM environment defined to run these routines, use DB2 core WLM environment DSNWLM_GENERAL, except as follows:

- ADMIN_COMMAND_DSN - Use DSNWLM_REXX
- ADMIN_COMMAND_UNIX - Use DSNWLM_PGM_CONTROL
- ADMIN_INFO_SYSLOG - Use DSNWLM_REXX
- ADMIN_INFO_SYSPARM - Use DSNWLM_NUMTCB1
- ADMIN_JOB_CANCEL - Use DSNWLM_PGM_CONTROL
- ADMIN_JOB_FETCH - Use DSNWLM_PGM_CONTROL
- ADMIN_JOB_QUERY - Use DSNWLM_PGM_CONTROL
- ADMIN_JOB_SUBMIT - Use DSNWLM_PGM_CONTROL
- ADMIN_UTL_MONITOR - Use DSNWLM_PGM_CONTROL

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

"DB2 administration routines panel: DSNTIPRL" on page 476

Related information:

Stored procedures for administration (DB2 Administration Guide)

Additional steps for enabling the CICS transaction invocation procedure

The CICS transaction invocation stored procedure, DSNACICS, invokes CICS server programs.

Before you begin

Before you complete these steps, either install the DB2-supplied routines during migration or install the DB2-supplied routines during installation.

Procedure

To enable the CICS transaction invocation procedure:

1. Optional: Set up a WLM application environment specifically for running DSNACICS.

   Although DSNACICS can run in a WLM address space with other stored procedures, DSNACICS performs better if it runs in its own address space.

   Recommendation: If you do not have a WLM environment defined to run this routine, use DB2 core WLM environment DSNWLM_CICS.
2. Create or modify the JCL startup procedure for the stored procedure address space for running DSNACICS.

   If you ran installation job DSNTIJMV, such a sample startup procedure has been created in your SYS1.PROCLIB data set. The generic name of the procedure is DSNWLMC. The CICS data set names in the procedure are based on the values that you specified on panel DSNTIPW. If needed, you can modify this startup procedure.

   If you did not run installation job DSNTIJMV or the part of it that creates this startup procedure, run the job now. You can also use the sample startup procedure in the job as a model to create your own startup procedure.

Related concepts:

"Job DSNTIJRT" on page 549

Related tasks:

"Migration step 13: Define DB2 Version 10 to z/OS: DSNTIJMV" on page 593

Assigning stored procedures and functions to WLM application environments (DB2 Performance)

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

"Data set names panel 3: DSNTIPW" on page 309

DSNACICS stored procedure (DB2 Administration Guide)

Additional steps for enabling the DSNAEXP EXPLAIN stored procedure

The DSNAEXP stored procedure can be used to issue EXPLAIN on SQL statements without needing the authority to issue the SQL statements.

Before you begin

Before you complete these steps, either install the DB2-supplied routines during migration or install the DB2-supplied routines during installation. Although DSNAEXP can run in an address space with other stored procedures, DSNAEXP performs better if it runs in its own address space. Therefore, you should set up a WLM application environment and startup procedure specifically for running DSNAEXP.

About this task

Procedure

To enable the DSNAEXP EXPLAIN stored procedure:

Define a PLAN_TABLE for each user. See member DSNTESC in *prefix*.SDSNSAMP for sample SQL statements.

Related tasks:

Assigning stored procedures and functions to WLM application environments (DB2 Performance)

Additional steps for enabling the common SQL API stored procedures

You can use the common SQL API stored procedures to retrieve information about the data server configuration, the short message text for an SQL code, and system information. These stored procedures are used primarily by tools.

Before you begin

Before you complete these steps, either install the DB2-supplied routines during migration or install the DB2-supplied routines during installation. To create the GET_CONFIG stored procedure with the correct security attributes, job DSNTIJRT must be run by a user ID with SYSOPR and MONITOR1 privileges.

Job DSNTIJRT installs and configures these stored procedures, but you must complete additional steps before you use them.

If GET_SYSTEM_INFO will be called to return the ICF catalog information about DB2 data sets, the user who defines the stored procedure must have MONITOR1 and DISPLAY privileges. If GET_SYSTEM_INFO will be called to return catalog attributes or ICF catalog information about DB2 data sets, the user who defines the stored procedure must have READ access to the OPERCMDS resource MVS.MCSOPER.DSNADMGS. If MVS.MCSOPER.DSNADMGS does not exist, the user must have READ access to the OPERCMDS resource MVS.MCSOPER.*. Otherwise, catalog attributes information will not be returned from GET_SYSTEM_INFO. To give the user that defines the procedures access to the RACF OPERCMDS class, the RACF security administrator must:

1. If the OPERCMDS class is not active, issue the following SETROPTS command to activate the OPERCMDS class:
   ```
   SETROPTS CLASSACT(OPERCMDS)
   ```
2. If the generic profile for OPERCMDS is not active, issue the following SETROPTS command to activate it:
   ```
   SETROPTS GENERIC(OPERCMDS)
   ```
3. Issue the RDEFINE command to establish a profile for MVS.MCSOPER.*console-name*. In this example, *console-name*=DSNADMGS:
   ```
   RDEFINE OPERCMDS MVS.MCSOPER.DSNADMGS UACC(NONE)
   ```
4. Give the user ID access to the class. In this example, the user ID is USER001:
   ```
   PERMIT MVS.MCSOPER.DSNADMGS CLASS(OPERCMDS) ID(USER001) ACCESS(READ)
   ```
5. Issue the SETROPTS RACLIST command to refresh the OPERCMDS reserve class:
   ```
   SETROPTS RACLIST(OPERCMDS) REFRESH
   ```

Before you complete the additional steps to enable the common SQL API stored procedures, set up WLM application environments for them to run in. These stored procedures run in WLM-established stored procedures address spaces.

The common SQL API stored procedures are:

- SYSPROC.GET_CONFIG
- SYSPROC.GET_MESSAGE
- SYSPROC.GET_SYSTEM_INFO

GET_SYSTEM_INFO must run in a separate WLM environment from DSNUTILS or DSNUTILU.

Procedure

To enable the common SQL API stored procedures:

In the JCL for starting the WLM-established address space for running the GET_SYSTEM_INFO stored procedure, ensure that all libraries in the STEPLIB DD concatenation are APF-authorized.

What to do next

Restriction: To run the GET_CONFIG and GET_SYSTEM_INFO stored procedures, DB2 must be started with subsystem parameters of LOBVALA=4096 KB or higher and LOBVALS=4096 MB or higher.

Additional steps for enabling WebSphere MQ user-defined functions

Before applications can call the WebSphere MQ user-defined functions, you must enable the user-defined functions.

Before you begin

If you want to use the MQI-based MQ functions, you must install WebSphere MQ Version 6 or later.

About this task

The MQ functions in the DB2MQ1C, DB2MQ2C, DB2MQ1N, and DB2MQ2N schemas are no longer supported because they use AMI, which is no longer supported. These functions are:

- GETCOL
- MQPUBLISH
- MQREAD
- MQREADALL
- MQREADALLCLOB
- MQREADCLOB
- MQRECEIVE
- MQRECEIVEALL
- MQRECEIVEALLCLOB
- MQRECEIVECLOB
- MQSEND
- MQSUBSCRIBE
- MQUNSUBSCRIBE

The schema for the following functions is now DB2MQ.

- MQREAD
- MQREADALL
- MQREADALLCLOB
- MQREADCLOB
- MQRECEIVE
- MQRECEIVEALL
- MQRECEIVEALLCLOB
- MQRECEIVECLOB
- MQSEND

Related reference:

Systems Programmer's Guide to Resource Recovery Services (RRS)

Systems Programmer's Guide to: Workload Manager

WebSphere MQ library

Moving from previous versions of the WebSphere MQ user-defined functions

During installation or migration, the current versions of some WebSphere MQ user-defined functions are installed under schema DB2MQ by job DSNTIJRT.

About this task

The following user-defined functions have new versions:

- MQREAD
- MQREADALL
- MQREADALLCLOB
- MQREADCLOB
- MQRECEIVE
- MQRECEIVEALL
- MQRECEIVEALLCLOB
- MQRECEIVECLOB
- MQSEND

Recommendation: Code new applications to use the functions in schema DB2MQ. The functions in schemas DB2MQ1C, DB2MQ2C, DB2MQ1N, and DB2MQ2N are no longer supported.

Converting from the AMI-based MQ functions to the MQI-based MQ functions

The MQI-based MQ functions use two tables, SYSIBM.MQSERVICE_TABLE and SYSIBM.MQPOLICY_TABLE, instead of Application Messaging Interface (AMI) configuration files. You need to bind the new functions, create the tables, and convert the contents of the AMI configuration files to rows in the MQ tables.

About this task

Job DSNTIJRT binds the new functions and creates the tables. You need to manually convert the contents of the AMI configuration files.

Procedure

To manually move the AMI configuration file data to the MQ tables:

1. Map sub-elements and attributes under the <ServicePoint> element to columns in SYSIBM.MQSERVICE_TABLE as indicated in the following table.

Table 159. How elements and attributes under the <ServicePoint> element map to columns in the SYSIBM.MQSERVICE_TABLE table

Element	Attribute of element	Column	Example
<ServicePoint>	longDescription	DESC_LONG	`<ServicePoint longDescription="" shortDescription="" ..xmi.label="DB2.DEFAULT.SERVICE">`
	shortDescription	DESC_SHORT	
	xmi.label	SERVICENAME	
<ServicePoint _queueName>	value	INPUTQUEUE	`<ServicePoint_queueName .. value="DB2MQ_DEFAULT_Q"/>`
<ServicePoint _queueMgrName>	value	QUEUEMANAGER	`<ServicePoint_queueMgrName .. value="MQND"/>`

2. Map sub-elements and attributes under the <Policy> element to columns in SYSIBM.MQPOLICY_TABLE as indicated in the following table.

Table 160. How elements and attributes under the <Policy> element map to columns in the SYSIBM.MQPOLICY_TABLE table

Element	Attribute of element	Value of attribute	Column	Value of column	Example
<Policy>	long Description	*value*	DESC_LONG	*value*	`<Policy longDescription="" shortDescription="".. xmi.label= "DB2.DEFAULT.POLICY">`
	short Description		DESC_SHORT		
	xmi.label		POLICYNAME		
<GeneralAttributes _syncpoint>	value	Yes	SYNCPOINT	Y	`<GeneralAttributes _syncpoint .. value="Yes"/>`
		No		N	
<SendAttributes _priority>	value	AsTransport	SEND_PRIORITY	-1	`<SendAttributes _priority .. value="AsTransport"/>`
<SendAttributes _persistence>	value	AsTransport	SEND _PERSISTENCE	Q	`<SendAttributes _persistence .. value="AsTransport"/>`
		Yes		Y	
		No		N	
<SendAttributes _expiry>	value	Unlimited	SEND_EXPIRY	-1	`<SendAttributes_expiry .. value="Unlimited"/>`
<SendAttributes _retryCount>	value	*n*	SEND_RETRY _COUNT	*n*	`<SendAttributes _retryCount .. value="0"/>`
<SendAttributes _retryInterval>	value	*n*	SEND_RETRY _INTERVAL	*n*	`<SendAttributes _retryInterval .. value="1000"/>`
<SendAttributes _newCorrelId>	value	Yes	SEND_NEW _CORRELID	Y	`<SendAttributes _newCorrelId .. value="No"/>`
		No		N	
<SendAttributes _responseCorrelId>	value	MessageID	SEND_RESPONSE _CORRELID	C	`<SendAttributes _responseCorrelId .. value="MessageID"/>`
		CorrelID		P	
<SendAttributes _exceptionAction>	value	DLQ	SEND_EXCEPTION _ACTION	Q	`<SendAttributes _exceptionAction .. value="DLQ"/>`
		Discard		D	
<SendAttributes _reportType Exception>, <SendAttributes _reportData>	value, value[1]	No	SEND_REPORT _EXCEPTIONDESC _LONG	N	`<SendAttributes _reportTypeException .. value="No"/>` `<SendAttributes _reportData .. value="Report"/>`
		Yes, Report		E	
		Yes, Report _With_Data		D	
		Yes, Report _With_Full_Data		F	
<SendAttributes _reportTypeCOA>, <SendAttributes _reportData>	value, value[1]	No	SEND_REPORT _COA	N	`<SendAttributes _reportTypeCOA .. value="No"/>` `<SendAttributes _reportData .. value="Report"/>`
		Yes, Report		C	
		Yes, Report _With_Data		D	
		Yes, Report _With_Full_Data		F	

Table 160. How elements and attributes under the <Policy> element map to columns in the SYSIBM.MQPOLICY_TABLE table (continued)

Element	Attribute of element	Value of attribute	Column	Value of column	Example
<SendAttributes _reportTypeCOD>, <SendAttributes _reportData>	value, value[1]	No	SEND_REPORT _COD	N	`<SendAttributes _reportTypeCOD .. value="No"/>` `<SendAttributes _reportData .. value="Report"/>`
		Yes, Report		C	
		Yes, Report _With_Data		D	
		Yes, Report _With_Full_Data		F	
<SendAttributes _reportTypeExpiry>, <SendAttributes _reportData>	value, value[1]	No	SEND_REPORT _EXPIRY	N	`<SendAttributes _reportTypeExpiry .. value="No"/>` `<SendAttributes _reportData .. value="Report"/>`
		Yes, Report		E	
		Yes, Report _With_Data		D	
		Yes, Report _With_Full_Data		F	
<ReceiveAttributes _waitInterval>	value	*n*	RCV_WAIT _INTERVAL	*n*	`<ReceiveAttributes _waitInterval .. value="60"/>`
<ReceiveAttributes _convert>	value	Yes	RCV_CONVERT	Y	`<ReceiveAttributes _convert .. value="Yes"/>`
		No		N	
<ReceiveAttributes _acceptTruncated Messages>	value	Yes	RCV_ACCEPT _TRUNC_MSG	Y	`<ReceiveAttributes _acceptTruncated Messages .. value="Yes"/>`
		No		N	
<ReceiveAttributes _openShared>	value	Yes	RCV_OPEN _SHARED	S	`<ReceiveAttributes _openShared .. value="Yes"/>`
		No		E	

Note:

1. The two values are used in conjunction to determine the column value.

Starting the queue manager

To enable WebSphere MQ user-defined functions, start the queue manager after you create and configure the broker domain.

Procedure

To start the queue manager:

1. Issue the following command from the z/OS console, where *<command-prefix-string>* is the command prefix string for the WebSphere MQ subsystem.:

   ```
   <command-prefix-string> START QMGR
   ```

   For example, if the command prefix string for your WebSphere MQ subsystem is –MQND, issue this command:

   ```
   –MQND START QMGR
   ```

2. To check whether the queue manager is available, issue the following command from the TSO Command Processor panel, which is option 6 of the ISPF/PDF primary options menu:

CSQOREXX

Customizing WLM application environments for running WebSphere MQ user-defined function support

To run WebSphere MQ user-defined functions, you need to set up two different WLM application environments and corresponding WLM startup procedures.

About this task

The WLM environments that you must set up are:

- An application environment and startup procedure for running the single-phase commit functions
- An application environment and startup procedure for running the two-phase commit functions

Related concepts:

DB2 MQ functions and DB2 MQ XML stored procedures (DB2 Application programming and SQL)

Related reference:

"Core WLM environments for DB2-supplied routines" on page 840

Verifying the DB2 and WebSphere MQ setup

After you set up the WebSphere MQ environment, you can verify that it is set up correctly by running DB2 jobs.

Procedure

To verify your WebSphere MQ setup:

1. Customize and run job DSNTEJMQ to verify that your WebSphere MQ environment is set up correctly for invoking DB2 WebSphere MQ user-defined functions. Instructions for customizing this job are in the job prolog. Job DSNTEJMQ defines a local queue and invokes each of the WebSphere MQ functions through DSNTEP2.
2. Customize and run jobs DSNTEJSQ and DSNTEJSV to verify that your WebSphere MQ environment is set up correctly for invoking DB2 WebSphere MQ publish/subscribe user-defined functions. Instructions for customizing these jobs are in the job prolog.

Enabling DB2 web services

Web services are sets of business functions that applications or other web servers invoke over the Internet by using standard HTTP or HTTPS requests.

Additional steps for enabling web service user-defined functions

Enabling the web service user-defined functions allows you to specify a complete SOAP message as input, so you receive the complete SOAP messages from the web service provider's response.

Before you begin

Before you complete these steps, set up WLM application environments for web service user-defined functions.

Recommendation: For web service functions, use the DSNWLM_WEBSERVICES core WLM environment.

About this task

During installation or migration, the following web service user-defined functions were installed by job DSNTIJRT:

- DSNWSCVV
- DSNWSCVC
- DSNWSCCV
- DSNWSCCC

Procedure

To set up the web service user-defined functions:

1. The default timeout to read data from a web service provider is two seconds. Increase the timeout value by redefining the web service user-defined functions with the key and value pair DB2SOAP_TIMEOUT=*X* in the ENVAR parameter values, where *X* is a number that specifies the timeout value in seconds.
2. If you want to use SOAP Version 1.2, include the key and value pair DB2SOAP_VERSION=SOAP_12 in the ENVAR parameter values. By default, DB2 uses SOAP Version 1.1. If you specify any other value for DB2SOAP_VERSION besides SOAP_12, SOAP Version 1.1 is assumed.

   If your subsystem uses SOAP Version 1.1 and SOAP Version 1.2, you must define two sets of SOAP user-defined functions: one without ENVAR("DB2SOAP_VERSION=SOAP_12"), and one with ENVAR("DB2SOAP_VERSION=SOAP_12").
3. If you do not use the DSNWLM_WEBSERVICES core WLM environment, add the following DD card to the WLM startup procedure where the user-defined functions are defined.

   ```
   // WSERROR DD PATH='<HFS filename>',
   // PATHOPTS=(ORDWR,OCREAT,OAPPEND),
   // PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH,SIWOTH)
   ```

   where <*HFS filename*> is the name of an HFS, such as /tmp/wsc.err.

   If an error occurs, the web service provider returns a SOAP response to the client with an HTTP return code set other than 200. In this case, the user-defined functions end with SQLCODE=-443 and the complete HTTP response that was sent by web service provider is stored in the WSERROR DD file, as specified in the WLM startup procedure. The file contains the actual bytes that are sent by the web service provider.

   If the response is in Unicode, you can browse the file on an MVS system by issuing the following command:

   ```
   iconv -f UTF-8 -t IBM-1047 < tmp/wsc.err
   ```

   You can store the contents in EBCDIC by saving it to another file by issuing the following command:

   ```
   iconv -f UTF-8 -t IBM-1047 < wsc.txt > wsc_ebcdic.txt
   ```

   You can also FTP the file in BINARY mode to a workstation and open it using a text editor.

Related concepts:

The SOAPHTTPNV and SOAPHTTPNC user-defined functions (DB2 Application programming and SQL)

Related reference:

AT-TLS policy statements (z/OS Communications Server: IP Configuration Reference)

Supported web service calls:

The web service user-defined functions support unsecured web services, HTTP Basic Authorized web services, and HTTPS Secured web services.

1. Unsecured web services

   For unsecured web services, the HTTP request that contains SOAP is sent in clear text to the web service provider, where no authentication is performed. The endpoint URL that is specified during invocation of the web service user-defined functions has the following form:

   ```
   http://[hostname[:port]/[path]
   ```

2. HTTP Basic Authorized web services

   For HTTP Basic Authorized web services, the specified user and password is encoded in Base64 and sent as part of an HTTP Header. The HTTP request that contains SOAP is sent in clear text to the web service provider, where it authenticates the supplied user and password before processing the request. The endpoint URL that is specified during invocation of the web service user-defined functions has the following form:

   ```
   http://[user[:password]@]hostname[:port]/[path]
   ```

3. HTTPS Secured web services

   For HTTPS Secured web services, as part of the HTTPS protocol, the complete HTTP request that contains SOAP is encrypted and sent to the web service provider. The endpoint URL that is specified during invocation of the web service user-defined functions has the following form:

   ```
   https://[hostname[:port]/[path]
   ```

Enabling the web service user-defined function support for HTTPS:

Support for HTTPS in the web service user-defined functions uses Application Transparent Transport Layer Security (AT-TLS). AT-TLS is based on z/OS System SSL, and it transparently implements Transport Layer Security (TLS) protocols in the TCP layer of the stack.

About this task

SSL connections make use of public/private key mechanisms (PKI) for authenticating each side of the SSL session and agreeing on bulk encryption keys to be used for the SSL session. To use PKI, public/private key pairs must be generated. In addition, X.509 certificates (which contain public keys) might need to be created, or certificates must be requested, received, and managed.

System SSL supports the following two methods for managing PKI private keys and certificates:

- A z/OS shell-based program called gskkyman. gskkyman creates, fills in, and manages a z/OS file that contains PKI private keys, certificate requests, and certificates. This z/OS file is called a key database and, by convention, has a file extension of .kdb.
- The z/OS Security Server (RACF) RACDCERT command. RACDCERT installs and maintains PKI private keys and certificates in RACF.

Procedure

To enable support for HTTPS and AT-TLS:

1. Specify the TTLS parameter on the TCPCONFIG statement in PROFILE.
2. To protect TCP/IP connections, you can configure the RACF EZB.INITSTACK.sysname.tcpname resource in the SERVAUTH class to block all stack access except for the user IDs that are permitted to use the resource. Refer to member EZARACF in sample data set hlq.SEZAINST where hlq is the high level qualifier dataset name for TCPIP datasets.
3. Configure AT-TLS policy rules. The policy agent provides AT-TLS policy rules to the TCP/IP stack. Each rule defines a set of security conditions that the policy agent compares to the conditions at the connection that it is checking. When the policy agent finds a match, it assigns the connection to the actions that are associated with the rule.
4. Create a client keyring for each client with necessary certification authority certificates. The name of the client keyring should match the name that is provided for the keyring in the policy configuration file.

   If the web service user-defined functions are defined with DB2 security, the authorization ID that is associated with the WLM-established address space where the user-defined functions are defined must have all the necessary permissions to access the keyring.
5. Stop and start TCP/IP stacks.
6. Start the policy agent. Verify that message EZZ4248E is released from the console.

Results

Setup is complete and you can run the web service user-defined functions with HTTPS.

Enabling DB2 as a web service provider

Enabling DB2 as a web service provider allows you to create web services on z/OS with your DB2 data and applications.

About this task

DB2 for z/OS as a web service provider has the following prerequisites:

- Enable JDBC (legacy or universal) in DB2.
- Install WebSphere Application Server Version 5 or later on z/OS.
- Ensure that the WebSphere Application Server library contains the following two files:
  - mail.jar
  - activation.jar

  If mail.jar is not present, download JavaMail Version 1.2 or later. If activation.jar is not present, download JavaBeans Activation Framework Version 1.0.1 or later.

To use DB2 Web Services Object Runtime Framework (WORF), you need to make the runtime services available to WebSphere Application Server (WAS). By default, WORF is installed in the HFS directory:

```
/usr/lpp/db2a10_worf/
```

The base install directory contains the `lib/` subdirectory that contains the runtime JAR file `worf.jar`. To begin using WORF, copy `worf.jar`, `mail.jar`, and `activation.jar` to a WAS shared library directory that you have already set up and restart WAS.

WORF provides a sample web application in the following directory:

```
lib/services.war
```

The application contains sample Document Access Definition Extension (DADX) files that define sample DB2 web services.

Procedure

To set up this application with sample DADX files:

1. Follow the instructions in the job prolog to customize and run job DSNTEJWS, which is located in the *prefix*.SDSNSAMP directory. DSNTEJWS creates the DB2 tables that are used by the sample application.
2. The sample application is set up to use universal JDBC driver connectivity to connect to a z/OS DB2 server. Configure WAS with JDBC providers that make use of universal JDBC driver and perform the following steps:
   a. Copy `services.war` to a temporary directory.
   b. Unjar `services.war` by issuing the following command:

      ```
      jar -xvf services.war
      ```

   c. Open the `group.properties` files, which are located in the following directories:

      ```
      WEB-INF/classes/groups/dxx_sample
      WEB-INF/classes/groups/dxx_travel
      ```

      Modify the dbDriver, dbURL userID, and password fields to have the following values:

      ```
      dbDriver=com.ibm.db2.jcc.DB2Driver
      dbURL=jdbc:db2://server:port/database
      userID=DB2 userid
      password=DB2 userid password
      ```

   d. Jar the files by issuing the following command:

      ```
      jar -cvf services.war *
      ```

3. Use a web browser to connect to your WAS Administrative Console.
4. Under "Applications", select "Install New Application".
5. Select the "Server Path" option, and type the location of the `services.war` file in the text box. If you installed WORF in the default location, the server path is:

   ```
   /usr/lpp/db2a10_worf/lib/services.war
   ```

6. Fill in a context root for the application (for example, services). Click "Next".
7. On the following screens, respond as necessary for your local setup. You can accept the default settings.
8. On the last screen, click "Finish". Click "Save to Master Configuration" to apply your changes.

What to do next

To load the application:

1. On the WAS Administrative Console's main page, under "Applications", select "Enterprise Applications". Select the application and click "Start" to load the application.
2. After the application loads, point a browser to your server with the context root that you chose (for example, http://*server*:*port*/services/). The welcome page lists the sample DADX files that are provided in `services.war`. To test the services, click on the links.

Enabling DB2 as a web service consumer

Enabling DB2 as a web service consumer allows you to receive web service data in your DB2 applications.

Before you begin

DB2 for z/OS as a web service consumer has the following prerequisites:

- Install IBM XML Toolkit for z/OS, C++ Edition 1.8
- Configure TCP/IP

About this task

Installation job DSNTIJMV contains sample startup procedure DSNWLM, which you can use as a model for your startup procedures. You must modify the startup procedure only if the web services routines are not defined to the DSNWLM_WEBSERVICES core WLM environment.

Procedure

To enable DB2 as a web service consumer:

1. Change the following items in each startup procedure:
   a. Change the procedure name from DSNWLM to the procedure name that you specified when you set up the WLM application environment.
   b. Change the value of APPLENV to the name of the WLM application environment that you set up for the web services consumer user-defined functions. The name of this environment must match the WLMENV parameter value in the configuration control statement for each stored procedure in job DSNTIJRT.
   c. Change the value of DB2SSN to your DB2 subsystem name.
   d. Add the data set name of the XML Toolkit load library (XPLINKed version) to the STEPLIB concatenation. If you used the default data set names when you installed the XML Toolkit, the load library data set name is *userid*.SIXMLOD1.
   e. Ensure that the DB2 SDSNLOD2 library is added to the STEPLIB.
2. After you set up the WLM application environment, create a JCL startup procedure for the stored procedure address space.

The IBM Data Server Driver for JDBC and SQLJ

If you plan to run JDBC or SQLJ applications, you need to install the current version of the IBM Data Server Driver for JDBC and SQLJ. You must complete this task after you install DB2 for z/OS or migrate to the current version of DB2 for z/OS.

Related tasks:

Installing the IBM Data Server Driver for JDBC and SQLJ (Application Programming for Java)

Connecting the IMS attachment facility

Connecting DB2 to IMS requires coordination with your company's group that supports IMS systems.

Procedure

To connect the IMS attachment facility:

- Make DB2 load modules available to IMS.
- Define DB2 to IMS.
- Define new programs and transactions to IMS.
- Depending on your site, you might also need to:
  - Define DB2 plans for IMS applications.
  - Generate a user language interface.

What to do next

An IMS system definition might be required to complete the steps to connect DB2 to IMS. If RACF is installed, you also need to define the IMS-to-DB2 connection to RACF.

Related tasks:

Defining DB2 resources to RACF (Managing Security)

Making DB2 load modules available to IMS

As part of connecting DB2 to IMS, you must make the DB2 load modules available to IMS.

About this task

If you have already included the *prefix*.SDSNLOAD library in your LNKLST*xx*, you can skip this step. Version 8 and Version 9.1 modules will be available through normal z/OS module search.

Procedure

To make DB2 load modules available to IMS:

- If DB2 load libraries are not included in your LNKLST*xx*, add STEPLIB statements to your startup procedures.
- Add *prefix*.SDSNLOAD to the DFSESL DD statement.
- If any IMS region connects to more than one release of DB2, then you must ensure that the DB2 load library that is used for that region is compatible with

each release. The IMS attachment facility is upward compatible, but not downward compatible. This means that you should use the oldest release of the DB2 load library for the IMS region.

– If all the data sets referred to in the JOBLIB or STEPLIB statement for an IMS region are APF-authorized, then add the DD statement for *prefix*.SDSNLOAD to the JOBLIB or STEPLIB statement. If the DYNAM option of COBOL is being used, the IMS RESLIB DD statement must precede the reference to *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statement.
– Add the ddname DFSESL DD statement for *prefix*.SDSNLOAD. All libraries specified on the DFSESL DD statement must be APF-authorized. The DFSESL DD statement is not required by DB2 DL/I batch support. IMS requires that an IMS RESLIB DD statement also be referenced by the DFSESL DD statement, as in the following:

```
//DFSESL     DD      DSN=ims_reslib,DISP=SHR
//           DD      DSN=prefix.SDSNLOAD,DISP=SHR
```

Defining DB2 to IMS

To define DB2 to IMS, the DB2 identification must be defined to the control region and the DL/I batch region. Optionally, you can also define the DB2 identification to each dependent region that accesses the DB2 subsystem.

Procedure

To define DB2 to IMS:

1. Name an SSM member in IMS.PROCLIB by concatenating the value of the IMSID field of the IMS IMSCTRL macro (one to four alphanumeric characters) with any name (one to four alphanumeric characters) defined by your site.

   Each SSM entry in IMS.PROCLIB defines at least one connection from an IMS region to at least one different z/OS subsystem. One SSM member can be shared by all of the IMS regions, or a specific member can be defined for each region. This record contains as many entries as there are connections to external subsystems. Each entry is an 80-character blocked or deblocked record. Fields are keyword or positional and are delimited by commas. The fields in this record are:

   SST=,SSN=,LIT=,ESMT=,RTT=,REO=,CRC=

   where:

   SST=DB2
   Identifies a required one-to eight-character name that defines the external subsystem type. It must be set to DB2 for IMS to connect to DB2.

   SSN=
   Identifies a required one-to four-character DB2 subsystem name. This name must be the name you specified for SUBSYSTEM NAME on installation panel DSNTIPM. The default is DSN1.

   LIT=
   Identifies a required four-character alphanumeric option, specifying the language interface token (LIT) supplied to IMS. The IMS-supplied language interface module (DFSLI000) requires a value of **SYS1** for this option.

   ESMT=
   Identifies a required one-to eight-character alphanumeric option specifying

the external subsystem module table. This module specifies which attachment modules must be loaded by IMS. **DSNMIN10** is the required value for this field.

RTT=
Identifies an optional one to eight character alphanumeric name of the user-generated resource translation table (RTT). This table maps the IMS application names into DB2 plan names. If this entry is omitted, the DB2 plan name is the IMS application load module name.

REO=
Identifies the optional one-character region error option to be used if an IMS application attempts to reference a non-operational external subsystem or if resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

R Passes a SQL return code to the application, indicating that the request for DB2 services failed (default). The most commonly returned SQL codes are -922, -923, and -924. However, there might be other SQL codes returned to the application.

When the first connection to DB2 cannot be established, a SQL return code is not returned. Instead, the application is terminated with an abend code U3047.

Q Abends the application with an abend code U3051, backs out activity to the last commit point, does a PSTOP of the transaction, and re-queues the input message. This option only applies when an IMS application attempts to reference a non-operational external subsystem or if the resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

A Abends the application with an abend code of U3047 and discards the input message. This option only applies when an IMS application attempts to reference a non-operational external subsystem or if the resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

If DB2 is not active or the connection cannot be established when the first SQL call is made from the application program (such as DB2 unavailable, DB2 quiescing, or DB2 terminating), the action you take depends on the region error option specified. SQL codes of -922, -923, or -924 might be returned to the application if option R is specified.

You can change the default for an application if a resource translation table entry is generated for that application.

CRC=
Identifies a command recognition character used by IMS to identify DB2 commands entered from an IMS terminal with the /SSR command. Any character is valid for the CRC except the period (.), slash (/), or comma (,). The default CRC is the hyphen (-).

These options apply to DL/I batch only:

CONNECTION_NAME=
Represents the name of the job step that is the coordinator for DB2 activity. The connection name is optional. The connection name defaults are:

Table 161. Default connection names for DL/I batch

Type of Application	Default Connection Name
Batch job	Job name
Started task	Started task name
TSO user	TSO authorization ID

If a batch job fails, you must use a separate job to restart the batch job. The connection name used in the restart job must be the same as the name used in the batch job that failed. Or, if the default connection name is used, the restart job must have the same job name as the batch update job that failed.

DB2 requires unique connection names for DB2 DL/I batch support. If two applications try to connect with the same connection name, then the second application is not allowed to connect to DB2. CONNECTION_NAME can be 1 to 8 characters long.

PLAN=
Identifies a DB2 plan name. If you do not specify a plan name, the application program module name is checked against the optional resource translation table. If a match is found, the translated name is used as the DB2 plan name. If no match is found, the application program module name is used as the plan name. PLAN can be 1-8 characters long.

PROG=
Identifies the name of the application program that is to be loaded and to receive control. PROG is required and can be 1-8 characters long.

2. Provide support for entering DB2 commands through IMS by defining the command recognition character (CRC) in the SSM member of the IMS control region. If the /SSR command is entered through the z/OS console, the AUTHID WTOR needs to be granted the appropriate authority. If the /SSR command is entered through an IMS terminal, the IMS LTERM name or the signon ID (if active) needs to be granted the appropriate authority. The /SSR command format is:

   ```
   /SSR crc DB2 command
   ```

   as in

   ```
   /SSR -DISPLAY THREAD (*)
   ```

3. Specify the SSM EXEC parameter in the startup procedure of the IMS control, MPP, BMP, or DL/I batch region. The SSM is concatenated with the IMSID to form a member name in IMS.PROCLIB. The IMSID comes from the IMSID option of the IMSCTRL generation macro or the IMSID option in the control region startup procedure.

   For DL/I batch regions, you can specify the DB2 connection parameters in the DDITV02 data set instead of an SSM member. The DDITV02 data set and an SSM member have the same format.

   If you specify the SSM for the IMS control region, any dependent region running under the control region can attach to the DB2 subsystem named in the IMS.PROCLIB member specified by the SSM parameter. The IMS.PROCLIB member name is the IMS ID (IMSID=*xxxx*) concatenated with the one to four characters specified in the SSM EXEC parameter. The IMS ID is the IMSID parameter of the IMSCTRL generation macro.

   IMS allows you to define as many external subsystem connections as are required. More than one connection can be defined for different DB2 subsystems. All DB2 connections must be within the same z/OS system. For a dependent region, you can specify a dependent region SSM or use the one

specified for the control region. You can specify different region error options (REOs) in the dependent region SSM member and the control region SSM member. The following table shows the different possibilities of SSM specifications.

Table 162. SSM specifications options

SSM for Control Region	SSM for Dependent Region	Action	Comments
No	No	None	No external subsystem can be connected
No	Yes	None	No external subsystem can be connected.
Yes	No	Use the control region SSM	Applications scheduled in the region can access external subsystems identified in the control region SSM. Exits and control blocks for each attachment are loaded into the control region address space.
Yes	Yes (NULL entry)	No SSM is used for the dependent region	Applications scheduled in this region can access DL/I databases only. Exits and control blocks for each attachment are loaded into the control region address space and each dependent region address space.
Yes	Yes	Check the dependent region SSM with the control region SSM.	Applications scheduled in this region can access only external subsystems identified in both SSMs. Exits and control blocks for each attachment are loaded into the control region and the dependent region address space.

No specific parameter exists to control the maximum number of SSM specification possibilities.

Related concepts:

Input and output data sets for DL/I batch jobs (DB2 Application programming and SQL)

Items to include in a batch DL/I program (DB2 Application programming and SQL)

System definition

Related tasks:

"Defining DB2 plans for IMS applications (optional)" on page 874

Defining new programs and transactions to IMS

You can define new programs and transactions that access DB2 resources to your IMS system.

About this task

Coordinate with your IMS support group to install the programs and transactions for Phase 4 of the verification process.

Programs and transactions already defined to IMS can use SQL without any additional definition to IMS.

Related tasks:

Chapter 7, "Verifying that installation or migration was successful with the sample applications," on page 743

Defining DB2 plans for IMS applications (optional)

The application plan defines the DB2 resources that an application accesses.

About this task

The application plan is identified by its plan name. Each IMS application is associated with a plan name.

The default is to have the DB2 plan name the same as the IMS application program load module name. If you assigned a different name to the plan, you need a resource translation table (RTT). If you chose an error option different from the REO default, you also need an RTT. DB2 provides the DSNMAPN macro in *prefix*.SDSNMACS to generate an RTT. After it is assembled, the table must be link-edited as REENTRANT with RMODE=24 into any authorized library that is concatenated with the library from which IMS loads the DB2 IMS attach modules.

The format of DSNMAPN macro is shown in the following table.

Table 163. DSNMAPN macro format

Macro	Option	Meaning
DSNMAPN	APN=	IMS application name
	,PLAN=	Associated DB2 plan name
	[,OPTION=]	Specific entry error option R, Q, or A. See REO in the SSM entry.
	[,END=]	Indicates last entry (YES/NO). NO is the default.

Related concepts:

"IMS attachment facility macro (DSNMAPN)" on page 875

Items to include in a batch DL/I program (DB2 Application programming and SQL)

IMS language interface module

You must generate a user language interface only if you intend to access two DB2 subsystems from the same dependent region.

To provide this access, the subsystem member (SSM) must contain one entry for each subsystem. Each entry contains a different subsystem ID and its associated language interface token (LIT). IMS provides the DFSLI macro to generate additional language interface modules with unique LITs. The general format of the macro is shown in the table below.

Table 164. DFSLI macro format and meaning

Macro	Option	Meaning
DFSLI	TYPE	Specifies the type of subsystem that can be accessed through this language interface module. DB2 is the only value supported by this option
	LIT	Defines a name (called LIT) to relate a language interface module with an entry in the SSM for the dependent region

When an IMS application issues a DB2 request, IMS knows the target subsystem by the LIT used in the request. For example, consider the case of a dependent region accessing two DB2 subsystems (DSN1 and DSN2):

- You generate a language interface with LIT=SYS2 (DFSLI001).
- You define two entries in the SSM member. The first entry points to DSN1 with LIT=SYS1; the second points to DSN2 with LIT=SYS2.
- You link-edit applications accessing the DSN1 subsystem with the IMS-provided language interface (DFSLI000).
- You link-edit applications accessing the DSN2 subsystem with the user-generated language interface (DFSLI001).

Even though a region can communicate with two or more DB2 subsystems, an IMS application can access only one—the DB2 subsystem referred to in the language interface that is link-edited. You can alter the SSM to route application requests to a different DB2 subsystem.

IMS attachment facility macro (DSNMAPN)

The DSNMAPN macro is required when the IMS application load module name is different than the name of its related IBM DB2 application plan. The macro is also required if the error option is different than the ERR value that is specified on the IMS SSM entry.

Macro statements are assembled in *prefix*.SDSNMACS and must be link-edited as REENTRANT with RMODE=24 into the DB2 library *prefix*.SDSNLOAD. The module name must be specified on the IMS SSM entry for the DB2 subsystem. The name must be specified as in the RTT entry for the SSM member defining the connection of this region. IMS loads the RTT module into the dependent region address space.

Note:

1. The macro name must be followed by one or more blanks before options are coded.
2. Multiple options must be separated by commas (with no blanks).

label **DSNMAPN**

DSNMAPN is the name of the macro. It must be coded exactly as it appears here, and it must be separated from any optional options by one or more blanks.

For *label*, substitute the CSECT name of your module. This name must match the name of the module specified to the linkage editor. *Label* is optional except for the first invocation of the DSNMAPN macro. The last invocation requires END=YES.

APN=*program-name*
Specifies the name of an application load module scheduled by IMS. For *program-name*, substitute an application name of up to eight characters.

PLAN=*plan-name*
Specifies an application plan name that is used (instead of the default application name) when a thread is created. For *plan-name*, substitute an application plan name of up to eight characters.

OPTION=R|Q|A
Specifies the action taken when an application program call cannot be performed because there is some problem in communication between the application program and the DB2 subsystem or if resources are unavailable. The value can be R, Q, or A.

If OPTION is not specified, the region error option (REO) is used.

R Specifies that a return code is returned to the application to indicate that the request for DB2 services failed.

Q Specifies that the transaction is abnormally terminated with an abend code U3051, activity is backed out to the last commit point, and the input message is re-queued.

A Specifies that the transaction is abended with an abend code of U3047, and the input message is deleted.

Default: R

END=NO|YES
Specifies whether this is the last DSNMAPN macro invocation. The value can be NO or YES.

NO Specifies that this is not the last DSNMAPN macro invocation.

YES
Specifies that this is the last DSNMAPN macro invocation.

Default: NO

The last DSNMAPN macro invocation must be followed by the specification END=YES.

Usage notes

- To enter more than one application name (with its corresponding plan name and OPTION specification), you must use multiple invocations of the DSNMAPN macro. The first invocation requires the label; the last invocation requires END=YES.
- Invocations must be in ascending order by application name. If they are not, an MNOTE macro error is generated.

Related tasks:

"Defining DB2 plans for IMS applications (optional)" on page 874

Connecting distributed database systems

You can use the distributed data facility (DDF) of DB2 to access data that is held by other data management systems. You can also use DDF to make your DB2 data accessible to other systems.

About this task

DB2 does not place any upper limit on the number of systems it can connect to; available storage is the limiting factor.

Related concepts:

Communicating with data sharing groups (DB2 Data Sharing Planning and Administration)

The communications protocols

DDF uses TCP/IP or SNA to communicate with other systems.

The figure below shows the connectivity options that you have with DDF in DB2.

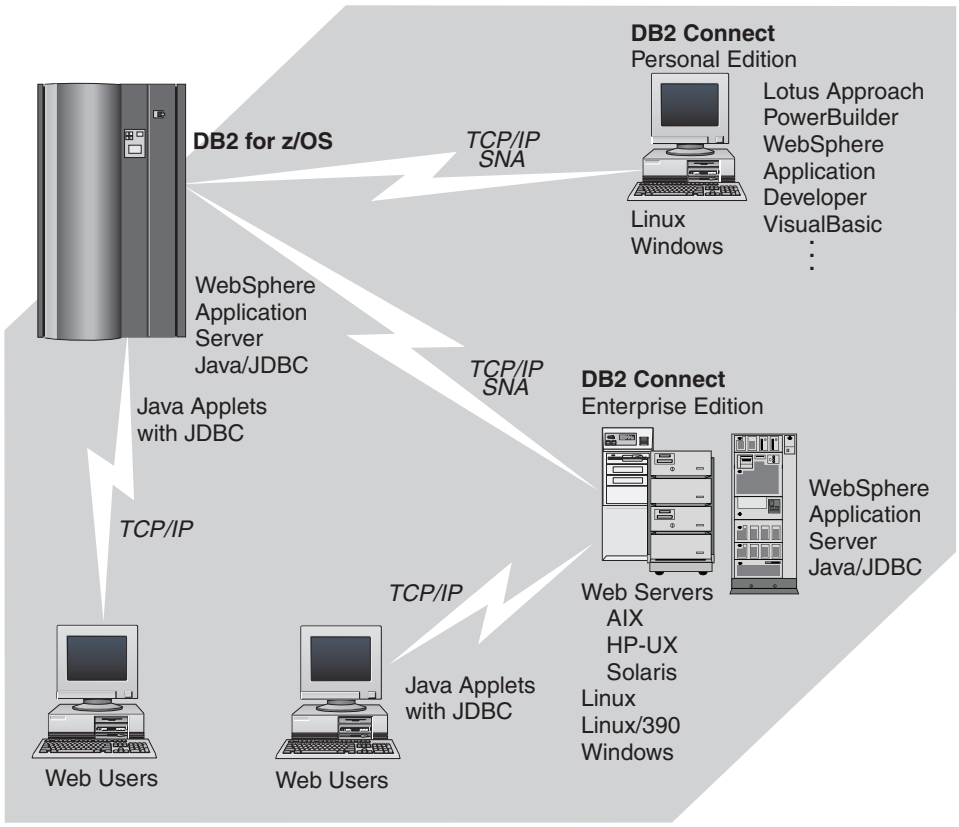


Figure 118. Connectivity options

Setting up a network for use by database management systems requires knowledge of both database management and communications. Thus, you must put together a team of people with those skills to plan and implement the network.

The role of the communications database (CDB)

When sending a request, DB2 uses the LINKNAME column of the SYSIBM.LOCATIONS catalog table of the communications database (CDB) to determine which protocol to use.

To receive VTAM requests, you must specify an LUNAME on installation panel DSNTIPR. To receive TCP/IP requests, you must select a DRDA port and a resynchronization port on installation panel DSNTIP5. TCP/IP uses the server's port number to pass network requests to the correct DB2 subsystem.

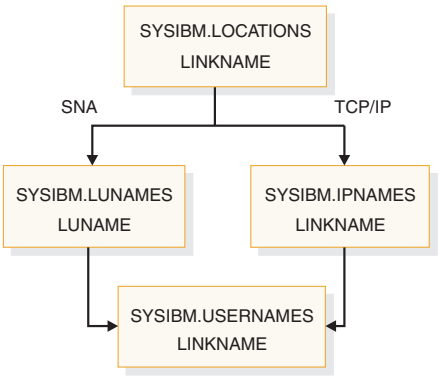


Figure 119. The LINKNAME column of SYSIBM.LOCATIONS determines protocol

If the value in the LINKNAME column is found in the SYSIBM.IPNAMES table, TCP/IP is used for DRDA connections. If the value is found in SYSIBM.LUNAMES table, SNA is used. If the same name is in both SYSIBM.LUNAMES and SYSIBM.IPNAMES, TCP/IP is used.

Attention: A requester cannot connect to a given location using both SNA and TCP/IP protocols. For example, if your SYSIBM.LOCATIONS table specifies a LINKNAME of LU1, and LU1 is defined in both the SYSIBM.IPNAMES and SYSIBM.LUNAMES table, TCP/IP is the only protocol used to connect to LU1 from this requester for DRDA connections.

The DRDA database protocol

Applications can use the Distributed Relational Database Architecture™ (DRDA) protocol to control remote access.

With DRDA, the application connects to a server at another location and executes packages that have been previously bound at that server. The application uses a CONNECT statement, a three-part name, or an alias to access the server.

Queries can originate from any system or application that issues SQL statements as a *requester* in the formats required by DRDA.

For two-phase commit using Systems Network Architecture (SNA) connections, DB2 supports both presumed abort and presumed nothing protocols that are defined by DRDA. If you are using TCP/IP, DB2 uses the sync point manager that is defined in the documentation for DRDA Level 3.

Related concepts:

Types of Two-Phase Commit (z/OS MVS Programming: Resource Recovery)

Related information:

The Open Group

Preparing your system for DRDA access

In DB2 Version 10, private protocol is no longer supported. Plans and packages that were bound with DBPROTOCOL(PRIVATE) and access remote locations can exist in Version 10, but attempts to load or run those plans or packages fail. DB2 provides a tool for preparing a subsystem to use DRDA access only.

Procedure

To prepare a DB2 subsystem to use DRDA access only:

1. Run migration job DSNTIJPM. This job identifies any plans and packages that must be rebound with DBPROTOCOL(DRDA). The DSNTIJPM queries identify two categories of plans or packages that were previously bound with DBPROTOCOL(PRIVATE):
   - Objects that only need to be rebound with DBPROTOCOL(DRDA).
   - Objects that have a private protocol remote location dependency, which at a minimum will require that the object be rebound with DBPROTOCOL(DRDA).
2. Run migration job DSNTIJPD to call the private protocol to DRDA protocol REXX tool (DSNTP2DP). The tool creates recommended commands to convert packages and plans to DBPROTOCOL(DRDA).
3. Review and edit the output of the DSNTP2DP tool as needed, and submit the commands to rebind the packages and plans to use DRDA protocol.
4. After all plans and packages have been successfully rebound, monitor a test of the applications.
   - If a package not found error (SQLCODE -805) occurs for a package at a remote location, the error is likely caused by the application using embedded dynamic SQL. The location that is specified in the error could not be determined from the catalog or the SQL statement text in the package. To fix this problem, submit a BIND PACKAGE COPY statement for the indicated package with the indicated location name specified as the target:

     GUPI

     ```
     BIND PACKAGE(location-name.collection-id) -
       COPY(collection-id.package-id) -
       COPYVER(version-id) OPTIONS(COMPOSITE) -
       OWNER(owner) QUALIFIER(qualifier) -
       DBPROTOCOL(DRDA) SQLERROR(CONTINUE)
     ```

     GUPI
   - A package error might also occur for a plan that could not be previously determined to access remote locations, and therefore its package list did not contain packages at remote locations. If you receive such an error, rebind the plan and use the PKLIST keyword to create a new package list. The package list should either contain the additional remote package or have a wildcard (*.) prefixed to each of the local package names in the list.

The private to DRDA protocol REXX tool (DSNTP2DP):

To help you convert your plans and packages from using private protocol to using DRDA protocol, DB2 provides the private to DRDA protocol REXX tool, DSNTP2DP. This tool scans your catalog and generates the necessary commands to convert all objects that have a private protocol and remote location dependency to DRDA.

Use job DSNTIJPD to invoke DSNTP2DP. You can tailor the generated output from the tool and run it at your discretion.

A package or plan has a private protocol and remote location dependency when the plan or package is bound with the DBPROTOCOL(PRIVATE) option and accesses remote locations. Packages and plans that were bound with the DBPROTOCOL(PRIVATE) option and access remote locations cannot execute in Version 10.

Authorization required

The DSNTP2DP statement itself requires no privileges to execute.

Syntax diagram

The private to DRDA protocol REXX tool (DSNTP2DP) statement defines the function that the tool performs.

```
                          ┌─PACKAGES──=──Y─┐   ┌─PLANS──=──Y─┐
►►──DSNTP2DP──SSID=ssid──┤                ├──┤             ├──────────────────────────────►◄
                          └─PACKAGES──=──N─┘   └─PLANS──=──N─┘
```

Option descriptions

DSNTP2DP has the following required parameter:

SSID=*ssid*
Specifies the subsystem ID of the DB2 subsystem for DSNTP2DP to examine. The subsystem ID is used to connect the tool to the DB2 subsystem.

The default is **ssid**.

DSNTP2DP has the following optional parameters. If none of the optional parameters are specified, DSNTP2DP examines all catalog objects (plans and packages).

PACKAGES
Specifies whether DSNTP2DP processes packages.

Y DSNTP2DP examines packages in the catalog. The default is **Y**.

N DSNTP2DP does not examine packages in the catalog.

PLANS
Specifies whether DSNTP2DP processes plans.

Y DSNTP2DP examines plans in the catalog. The default is **Y**.

N DSNTP2DP does not examine plans in the catalog.

DRDA enhancements for migration from DB2 Version 8

Enhancements to DRDA in DB2 Version 10 improve performance and security.

Some of the DRDA enhancements in Version 10 include:

- DB2 can use IPv6 only with DRDA.
- DDF supports running the address space in AMODE 64, which relieves the storage constraints that exist in Version 8. The performance of TCP/IP has improved because the communications buffer is in 64-bit shared memory. This change means that you do not have to copy the buffer into your own address space or into the DBM1 address space. Reply communications are built and defined in 64-bit shared memory and can be pointed to instead of copied into your own address space or into the DBM1 address space.
- DB2 can use trusted application servers with DRDA.
- The performance of LOB retrieval is improved.
- VTAM independence allows you to set up a DB2 DDF to support TCP/IP and DRDA access.
- DB2 has a new DRDA secure port, which allows use of SSL authentication methods. SSL authentication methods support secure communications between a client and DB2.
- DB2 uses new DRDA capabilities to encode DRDA metadata in Unicode instead of EBCDIC, thereby eliminating the need for character conversion from EBCDIC to Unicode.
- New monitoring capabilities in DRDA enable monitoring tools to track the execution of statements and to track the sequence of transactions in the distributed environment.
- DB2 supports the following new DRDA user data type capabilities:
  - TIMESTAMP data (SQLTYPE 392/393) with extended timestamp precisions
  - TIMESTAMP WITH TIME ZONE data type (SQLTYPE 2448/2449)
- DB2 implements new DRDA capabilities that support various new Version 10 SQL functions. If new DRDA capabilities are required to support the new Version 10 SQL functionality, then the requester system must be at the correct DRDA level to use the new SQL function. These new capabilities include support for the PREPARE ATTRIBUTES clause CONCENTRATE STATEMENTS WITH LITERALS.
- DRDA always resolves aliases during a PREPARE of a dynamic SQL statement or binding of a static statement into a package at a remote location.

DRDA enhancements for migration from DB2 Version 9

Some enhancements were made to DRDA in DB2 Version 10.

The DRDA enhancements in Version 10 include:

- DB2 uses new DRDA capabilities to encode DRDA metadata in Unicode instead of EBCDIC, thereby eliminating the need for character conversion from EBCDIC to Unicode.
- New monitoring capabilities in DRDA enable monitoring tools to track the execution of statements and to track the sequence of transactions in the distributed environment.
- DB2 supports the following new DRDA user data type capabilities:
  - TIMESTAMP data (SQLTYPE 392/393) with extended timestamp precisions
  - TIMESTAMP WITH TIME ZONE data type (SQLTYPE 2448/2449)

- DB2 implements new DRDA capabilities that support various new Version 10 SQL functions. If new DRDA capabilities are required to support the new Version 10 SQL functionality, then the requester system must be at the correct DRDA level to use the new SQL function. These new capabilities includes support for the PREPARE ATTRIBUTES clause CONCENTRATE STATEMENTS WITH LITERALS.
- DRDA always resolves aliases during a PREPARE of a dynamic SQL statement or binding of a static statement into a package at a remote location.

DDF installation and implications for DB2

The installation options for DDF are described on installation panels DSNTIPR and DSNTIP5.

Use these options to define, among other things:

- Whether you want DDF to start automatically when DB2 starts
- Important names for this DB2 subsystem, including a LU name and NETID

  If you plan to use both VTAM and TCP/IP, you must define an LU name and NETID. DB2 as a requester generates the unit of work using NETID and LUNAME.
- Thread management options
- Security options
- TCP/IP port numbers
- Control of the number of DRDA query blocks that can flow on a network request that was specified with OPTIMIZED FOR n ROWS where *n* exceeds the number of rows that fit in a single query block.

If you do not plan on using SNA/APPC communications, you can set up DDF to not activate its SNA/APPC support. To set up DDF to not activate its SNA/APPC support, give the DB2 subsystem an IPNAME value by updating the BSDS DDF record. Setting up DDF to not activate its SNA/APPC support is not supported by the DB2 installation option panels. DB2 TCP/IP communications uses the IPNAME value and a character representation of the TCP/IP resync port (RESPORT) hexadecimal value to identify units of work.

Support for extended dynamic SQL: If this DB2 subsystem services requesters that support extended dynamic SQL, such as DB2 Server for VSE & VM, enter YES in field DESCRIBE FOR STATIC on installation panel DSNTIPF. This option lets applications from the requesting system execute SQL DESCRIBE statements that appear as extended dynamic SQL statements in the requesting system, but appear as static SQL in the DB2 package. For the option to take effect, you must bind the package with DESCRIBE FOR STATIC enabled.

Test your connections You should test systems with each other to ensure that their communications setups are correct. If you are testing with another DB2 for z/OS, enter the location name of that other site in field REMOTE LOCATION of installation panel DSNTIPY. The remote location must also have DDF installed and active and must have run the first sample job, DSNTEJ1.

Related reference:

"Distributed data facility panel 1: DSNTIPR" on page 442

"Distributed data facility panel 2: DSNTIP5" on page 448

DSNJU003 (change log inventory) (DB2 Utilities)

Connecting systems with VTAM

Virtual Telecommunications Access Method (VTAM) is a program that you can use to control remote DB2 communication.

About this task

Remote communications terminology:

Logical unit (LU)
A source of requests entering the network and a receptor of replies from the network. For example, a particular DB2 is an LU.

Session
A logical connection between two LUs. Multiple sessions can run on a single physical connection.

Conversation
A dialog that uses a session to transfer information between transaction programs, such as DB2 to DB2. A single session can support multiple conversations, but only one at a time.

To prepare DB2 for communication using VTAM and the distributed data facility (DDF), complete the following steps. You can complete steps 1, 2, and 3 after installing DB2. Steps 6 through 8 are optional.

- **Step 1: Customize VTAM for DB2**

  To make monitoring of the network easier, consider installing NetView.

- **Step 2: Choose names and a password**

  You need to choose two names for the local DB2 subsystem: a location name and a logical unit name (LU name).

  A *location name* distinguishes a specific database management system in a network, so applications use this name to direct requests to your local DB2 subsystem. Other systems use different terms for a location name. For example, DB2 Connect calls this the *target database name*. The DRDA term, *RDBNAM*, is used to refer to non-DB2 systems' relational database names.

  An *LU name* is the name by which VTAM recognizes this subsystem in the network. You might need to know the LU names of other systems that can request data from the local DB2 subsystem, or you can use a default LU name of eight blanks.

  If you plan to request data from other systems, you need the LU names *and* location names for those serving systems. Most of the time, system administrators and operators need to know both names, because they can use both names in various commands, and DB2 uses both names in messages.

  In addition to the names mentioned above, you can choose an optional password to validate your local DB2 subsystem to VTAM. If the z/OS system on which DB2 is running is part of an z/OS Parallel Sysplex, you can choose a generic LU name to define a DB2 group to remote locations.

- **Step 3: Define the DB2 subsystem to VTAM**

Use the VTAM APPL statement to make the DB2 subsystem known to VTAM. You must include the APPL definitions in the VTAM SYS1.VTAMLST library at VTAM startup.

Also use the VTAM MODEENT statement to define default session modes. You must include mode tables in the VTAM SYS1.VTAMLIB library at VTAM startup.

Sample VTAM definitions are provided in the data set DSN8VTAM in SDSNSAMP and in examples throughout this topic.

- **Step 4: Populate the communications database**

  The DB2 catalog includes the communications database (CDB), which contains several tables that hold information about your connections with remote systems. You must populate some of these tables before you can request data from those remote systems. If this DB2 system only services data requests, you do not have to populate the CDB; you can use the default values.

- **Step 5: Start VTAM to use DB2**

  When you start VTAM to use DB2, you must be sure that the proper definitions are in the VTAM libraries VTAMLST and VTAMLIB.

- **Step 6: Tune the system**

  This is an optional step, which you can do after you have established communications between two or more systems. The procedure outlined up to this point gives you default values for your DB2 modes and your class of service. Although the defaults are probably adequate for your preliminary testing, you can change them to improve performance in the network, or to assign different modes to different application plans.

  When VTAM links two nodes, it establishes a session. The number of available sessions can have a significant impact on performance; therefore, you might need to modify your session limit values. Also, large amounts of DB2 data travelling through the network can severely affect VTAM storage, and you might need to tune buffer storage.

  You can also tune the system by changing mode options. A *mode* describes various characteristics of a session, such as the maximum number of bytes sent at one time. Modes can point to a *class of service* table, which ranks the available virtual routes for this mode with respect to preference of use and paths through the network. Essentially, the class of service table allows you to assign different network priorities to your modes.

- **"Step 7: Create Aliases"**

  This is an optional step. Each DB2 location can create aliases for the tables it wants to access, using DRDA, at the other DB2 locations.

- **"Step 8: Provide Authorization for an Appropriate Level of Security"**

Related concepts:

"Sample VTAM definitions to connect two DB2 subsystems" on page 911

Related tasks:

"Calculating session limits" on page 907

"Calculating VTAM I/O buffer pool (IOBUF) storage" on page 907

Related reference:

- IBM Tivoli NetView for z/OS (Installation: Getting Started)
- Generic resources (VTAM Network Implementation Guide)
- Implementing a VTAM network (VTAM Network Implementation Guide)
- Tuning VTAM for your environment (VTAM Network Implementation Guide)

Related information:

- Managing distributed access (Managing Security)

Customize VTAM for DB2

For DB2 to provide the best performance for distributed database systems, you probably need to customize VTAM.

Before you customize VTAM, consider the communication needs of your DB2 connections. Because you could allow your DB2 subsystem to send large amounts of data through the network, reexamine the capacity of your existing network. In some cases, portions of your existing network might need additional communication hardware to provide the required capacity.

Related reference:

- VTAM Network Implementation Guide

Choosing names and a password

When you connect a DB2 subsystem to VTAM, you must choose names for your local DB2 subsystem and, possibly, a VTAM password for it. You might also need to know the names of remote systems in the network.

Choosing names for the local subsystem

You define the names for the local subsystem and its VTAM password to DB2 by using the installation panels, or by using the change log inventory utility.

Procedure

Choose the following names for the local DB2 subsystem:

- A unique name by which the other systems in the network can recognize your subsystem. The name can have from 1 to 16 characters and is called the *location name*. (DB2 Connect refers to this as the *target database name*.) Make sure that the local location name is different from the name of every other system in the network, no matter where it is physically located.

  You must share the location name with the other systems that need to send SQL requests to this one.

  The location name should not change even if the network changes. Therefore, tightly control the allocation of location names. To ensure uniqueness, you use an IBM-registered SNA NETID as the first six bytes of your location name. If

location names are not unique, you have to change many programs and tables if your network is later joined with another network using the same location name.

The IBM recommendation for the NETID is the following format:

– The first two bytes are the country code as defined in ISO standard ISO 3166. These codes include the uppercase letters A through Z.
– The next four bytes are the enterprise code of the registering enterprise. This might already be registered with IBM as your SNA NETID. The enterprise code can include the uppercase letters A through Z, the numbers 0 through 9, and the underscore character (_).

To register your SNA NETID, see your IBM representative.

- A name by which VTAM can recognize the local subsystem. It must be either a unique name or, in some cases, a generic name.
  – The *unique* name must be unique within the network of connected systems, can have from 1 to 8 characters, and is called the *LU name*. The LU name and the location name of a subsystem can be identical, but this is not recommended; LU names are unique only within a network, and networks can change. You must share the LU name with any system that requests data from your local subsystem. Later, you enter this name in the VTAM APPL statement.
  – If the z/OS system on which DB2 is running is part of an z/OS sysplex, you can use a *generic* 8-character name to represent a group of VTAM LU names. The generic name might be useful if your network is in a transitional period, and you want to use generic names to reference network nodes.

    Specify the generic LU name in the field DB2 GENERIC LUNAME on installation panel DSNTIPR. Use column GENERIC of SYSIBM.LUNAMES to indicate that you want to use the generic LU name for CNOS processing and SQL requests to a particular server.
- Server location aliases. You can use the Change Log Inventory utility to define up to eight aliases for a location. Applications can use these alias names to refer to the local DB2 subsystem or data sharing group.

  Aliases are most useful in a data sharing environment in which two or more DB2 subsystems are migrated to a single data sharing group. In this case, you can define the old location for each subsystem as an alias for the location name of the group. Remote applications that refer to the old location names do not need to change.

  Use the Print Log Map utility to print all location alias names that are defined for a DB2 subsystem.

  You can set up multiple server location aliases as locations on the z/OS requester to restrict which members an application will use when accessing a data sharing group. The z/OS requester must be using the IPLIST or LULIST table to route connections to the data sharing group.

Related concepts:

SYSIBM.LUNAMES (DB2 Data Sharing Planning and Administration)

Related tasks:

"Defining the DB2 subsystem to VTAM" on page 888

"Using the change log inventory utility to update the BSDS" on page 916

Choosing a password for the local subsystem

Choosing a VTAM password is optional. DB2 does not require you to use a password as long as you have not included one in the VTAM APPL statement.

Procedure

To set a local subsystem password:

Enter the password by using the PRTCT option of the VTAM APPL statement. The password can have from one to eight EBCDIC characters.
This password is not transmitted through the network, so there is no need to share the password with the other systems.

Related tasks:

Controlling LU attachments to the network (Managing Security)

Names that you need from the remote systems

If you plan to have your DB2 subsystem request data from other systems, you need the LU names and location names for those serving systems.

Location names and LU names: When you populate the communications database (CDB) in the local DB2, you must know the location names (or DRDA RDBNAMs) and LU names of remote servers. The remote servers are the systems from which this DB2 system is to request data. The local DB2 system does not need location names of requesters. However, you need to know the LU names of the requesters if you intend to change default communication options.

DB2 does not receive DRDA RDBNAM from requesters other than DB2 for z/OS. If DB2 does not have an RDBNAM, it displays LU names in messages, display output, and trace output. To help you distinguish between location names and LU names in those cases, the LU name is enclosed in less-than (<) and greater-than (>) brackets.

When your systems begin communicating, you and others involved in working with distributed systems need to be aware of the LU name to DRDA RDBNAM mappings. When you have obtained the necessary names, enter them in the CDB.

Transaction program names (TPNs): If a server is not a DB2 for z/OS, it might have an additional name that uniquely identifies it. In LU 6.2, this is known as a *transaction program name* (TPN), and can be from 1 to 64 characters long. When a DB2 for z/OS subsystem communicates with other DB2 for z/OS subsystems, you do not need to supply TPN values. The DB2 subsystems automatically choose the correct TPN values.

When a TPN is necessary: You might need to supply TPN values when a DB2 subsystem requests data from a server that is not a DB2 for z/OS subsystem. For cases where the server does not accept the default TPN for DRDA access, enter into your CDB the TPN chosen by that server. For DB2 for VM, for example, the TPN is the SQL database machine ID.

TPN values accepted by DB2 for z/OS: A requester that is not DB2 for z/OS must use either the TPN name X'07F6C4C2' or DB2DRDA, which are the only values DB2 recognizes when it accepts a request from another system. Some requesters enter the TPN as two separate fields: a 1-byte prefix (X'07') and a 3-byte suffix ('6DB').

Related tasks:

"Populating the communications database for use with TCP/IP" on page 925

Names that Spiffy Computer Company chose

Spiffy has chosen location names and LU names, some of which are used in the provided examples.

Table 165. Spiffy's location names, LU names, and transaction program names (TPNS)

Location Name	LU Name	TPN	Comments
USIBMSTODB21	LUDB21		DB2 *
USIBMSTODB22	LUDB22		DB2
USIBMSTOSQL1	LUSQLDS	TPNSQLDS1	DB2 for VM production system
USIBMSTOSQL2	LUSQLDS	TPNSQLDS2	DB2 for VM test system

Note: *USIBMSTODB21 plans to accept requests from many Windows requesters.

Defining the DB2 subsystem to VTAM

You need to use an APPL definition statement and a MODEENT macro to define the DB2 subsystem to VTAM. Samples of both the APPL and MODEENT macros are in the DSN8VTAM sample data set.

The APPL statement

A VTAM APPL definition statement defines the VTAM options for the DB2 subsystem and includes it in a major node.

With VTAM, you can use a model application program definition for DB2. With a model definition, you use wild card characters for the application name (LU name).

Spiffy uses the statement in the table below for the USIBMSTODB21 DB2 subsystem:

```
LUDB21 APPL    APPC=YES,                                        X
               ATNLOSS=ALL,                                     X
               AUTH=(ACQ),                                      X
               AUTOSES=1,                                       X
               DMINWNL=25,                                      X
               DMINWNR=25,                                      X
               DSESLIM=50,                                      X
               MODETAB=DB2MODES,                                X
               PRTCT=D02DN,                                     X
               SECACPT=ALREADYV,                                X
               SRBEXIT=YES,                                     X
               SYNCLVL=SYNCPT,                                  X
               VERIFY=NONE,                                     X
               VPACING=2
```

Figure 120. Example of a VTAM APPL definition statement

For your convenience, the APPL statement example is provided in data set DSN8VTAM, in the sample library, SDSNSAMP.

The topics that follow describe the APPL options that Spiffy uses and a few more in which you might be interested. Not all APPL options are described.

Related reference:

VTAM Application Program Major Node (VTAM Resource Definition Reference)

Options for which you must choose values:

For some options in your APPL statement, you must supply a specific value; for others, DB2 suggests values that are not the VTAM default values.

In your APPL statement, you must code values for the following options:

name
: The 1 to 8 character LU name you chose. For their USIBMSTODB21 DB2 system, Spiffy uses LUDB21.

AUTOSES
: The number of contention winner sessions that VTAM is to activate automatically between this DB2 and another system on a given mode before DB2 requests a conversation to be created.

  Contention occurs when two LUs want to allocate a conversation at the same time in the same session. In order to resolve contention situations, VTAM denotes one LU as the contention winner and one as the contention loser. The winner automatically prevails and is allowed to allocate its conversation. The loser must wait to allocate its conversation.

  The default is 0. The suggested value is 1 or greater to ensure that VTAM informs DB2 if a session is inactivated.

  Too large a number can take up storage and create resources that are not used. A small number can result in a one-time delay to bring up additional sessions when they are needed by an application.

DMINWNL
: The minimum number of parallel sessions in which, if there is contention for a conversation, this local DB2 subsystem is the winner.

  The suggested value is one-half the value of DSESLIM, described below.

DMINWNR
: For the same situation as described for DMINWNL, the number of sessions in which the remote system is the winner. The suggested value is one-half the value of DSESLIM, described below.

DSESLIM
: The default maximum number of sessions allowed for this DB2 subsystem as it communicates with any other system on a given mode. For performance reasons, the DB2 suggested value for DSESLIM is the maximum number of sessions that can possibly be in use on any mode. For example, assume you have 5 modes for which the following maximum numbers of sessions could be active: 10, 12, 20, 30, 40. In this case, DSESLIM should be 40.

  Because calculating a precise value for this number can be rather difficult if you do not know exactly how many applications run on a specific mode, Spiffy chooses 50. They can modify this option later if they have problems obtaining enough sessions, or if they find they are requesting sessions that they never need.

  You can use DSESLIM to control the number of sessions that this subsystem can issue or receive. For example, to avoid overloading this subsystem with

requests from remote application processes, you can assign a low number to DSESLIM to limit the number of simultaneous remote requests issued by a given partner and mode.

Use the CONVLIMIT column of the LUMODES table in the CDB to override this value for specific cases.

MODETAB
The name of the VTAM logon mode table you use to define DB2 session modes. Only modes defined in this table are eligible for conversations created by the local DB2. If you leave this blank, DB2 uses the default mode table shipped with VTAM (ISTINCLM). Spiffy decides to set up a separate mode table and chooses the name DB2MODES. DB2 cannot use either the default mode table or the one you set up yourself until you make entries into the table.

PRTCT
If you decided to use a password, this is that password. Later, you must store the same password in the bootstrap data set (BSDS), entering it through installation panels or the change log inventory utility.

If you prefer not to use a password, omit this option. The installation panels and the change log inventory utility do not require you to enter a password.

SECACPT
The level of conversation-level security allowed.

Recommendation: Use ALREADYV, which gives you the most flexibility in determining your security. You can use the CDB to determine levels of security on a more granular basis.

In general, SECACPT=CONV should not be used because in many cases, it does not allow already verified conversations for DRDA access.

VERIFY
Whether you want SNA partner LU verification. The default, VERIFY=NONE, means that any system can connect with yours. Because Spiffy is setting up a small, restricted network, it chooses the default for now.

Use VERIFY=REQUIRED to activate partner LU verification. This means that you let RACF and VTAM check the identity of an LU that is attempting to connect with yours.

DB2 has no dependency on the value you choose.

VPACING
The maximum number of messages that another system can send to this local DB2 subsystem during a conversation before waiting to receive a pacing response. The suggested value is 2.

These message sizes are determined by the RUSIZES option of the MODEENT macro. VPACING and RUSIZES, together with some overhead, determine the amount of storage required for the pacing window.

Related concepts:

"CNOS processing in DB2" on page 909
"Session-level pacing" on page 901
"The MODEENT macro" on page 893

Related tasks:

"Calculating session limits" on page 907
"Update SYSIBM.LUMODES with conversation limits" on page 904
Controlling LU attachments to the network (Managing Security)

Related reference:

LU 6.2 Security (VTAM Network Implementation Guide)

Options that you must code exactly as given:

In some cases, DB2 requires particular values of APPL options.

For the following options, you must code the values exactly as shown; they are *not* the VTAM defaults:

APPC=YES
Tells VTAM that DB2 uses APPC conversation verbs.

AUTH=(ACQ)
Determines the DB2 system authority to use certain VTAM functions.

SRBEXIT=YES
Tells VTAM that DB2 uses service request block (SRB) processing in its exit routines.

SYNCLVL=SYNCPT
Tells VTAM that DB2 supports two-phase commit. Other systems communicating with this DB2 use this indication to determine if DB2 supports the updating of many locations in one unit of work.

Coding SYNCLVL=SYNCPT does not preclude the support of partner LUs that do not support two-phase commit. DB2 still supports the non-two-phase process.

Options that must use VTAM defaults:

Some options in your APPL statement require you to use the VTAM default values. You do not need to code these options.

DB2 must use the VTAM default values for the following options:

HAVAIL=NO
Indicates whether an XRF session can be supported. DB2 requires the default, NO.

PARSESS=YES
Specifies that parallel sessions are allowed. This defaults to YES when APPC=YES.

ENCR=NONE
Specifies information about specific cryptographic requirements. There is no support for encryption in this release of VTAM for LU 6.2 applications; therefore, this must be NONE.

SONSCIP=NO
Specifies information about SCIP exit routines. DB2 does not have SCIP exit routines; this must be NO.

VTAMFRR=NO
Specifies whether a VTAM functional recovery routine is in effect when control is returned to DB2. DB2 uses its own recovery routines; this must be NO.

Other options of interest:

In most cases, you can reasonably use the VTAM default values for the APPL statement at first.

You can change these values later. The options are listed below in case you have some reason to not use the default values.

ACBNAME
The LU name for the DB2 subsystem. If the ACBNAME is different from the APPL name and both the originating and destination LUs are in the same VTAM domain, do not refer to the ACBNAME in a CDB definition. If the ACBNAME is not the same as the APPL name, VTAM may encounter name conflicts.

DDRAINL
Whether the local DB2 subsystem wants to accept permission to drain its allocation requests if a change-number-of-sessions (CNOS) request is received that specifies that draining is allowed. The suggested value is the default, NALLOW (do not allow draining).

DRESPL
Whether the local DB2 is responsible for deactivating sessions when it receives a CNOS request specifying the local DB2 as the responsible system. The suggested value is the default, NALLOW (do not be responsible).

EAS
The approximate number of concurrent sessions for this DB2 subsystem. For performance reasons, it is better to estimate slightly high. The VTAM default is 509.

LMDENT
The number of entries to be used for a hash table of other systems. The suggested value is the approximate number of other systems in the network. In the scenario, Spiffy decides to use the default value of 19.

MAXPVT
The maximum additional amount of private area storage that can be used by VTAM within the DDF address space for the session-related control blocks and messages for DB2. Specifying 0 indicates an unbounded amount; this is the VTAM default.

OPERCNOS
The ability to have a VTAM operator display and set VTAM session limits for a given LUNAME and MODENAME.

- Use ALLOW to enable a VTAM operator to change session limits dynamically without stopping DDF or changing the CONVLIMIT column of the SYSIBM.LUMODES table.
- Use NALLOW, the default, to make sure VTAM operators are not able to dynamically change session limits for DB2.

Options that are ignored by DB2:

Some options for VTAM APPL statements are not applicable to DB2. Do not code these options in your APPL statement.

The following options are not applicable to DB2 :

ASLENT	ATNLOSS	MDLTAB	SSCPFM
ASLTAB	MDLENT	POAQNAM	USSTAB

The MODEENT macro

A VTAM link between two systems is a *session*. For every session, a defined set of characteristics called a *mode* must exist in a VTAM table called a *log mode table*.

This is the table that you named in the MODETAB option of the APPL statement.

You can create your own log mode table or add mode names to the default mode table, ISTINCLM, which is shipped with VTAM. If you decide to add your modes to ISTINCLM, you can find that table in SYS1.SAMPLIB.

Spiffy uses the DB2 default modes at first, but also sets up a separate mode table for modes that are used by DB2 for distributed data processing. This table can then be populated with additional modes as they are needed.

Related concepts:

"The APPL statement" on page 888

Default VTAM modes:

Use the MODEENT macro to enter each default mode into your mode table. When this table is complete, you must assemble and link-edit it into SYS1.VTAMLIB.

The following modes are the default modes:

- SNASVCMG is an optional mode. It is reserved for use by VTAM for CNOS processing and exists in the VTAM default log mode table. Because SNASVCMG is reserved for use by VTAM, do not enter it as a mode name in the CDB. If you have decided to set up a separate mode table for DB2, you can, if you choose, copy the SNASVCMG mode entry into your DB2 mode table, or just use it as it exists in the ISTINCLM mode table.
- IBMRDB is a recommended mode entry because it is used as a default for DRDA access whenever you do not explicitly assign a mode to a session. It does *not* exist in the default table; to use it as a default you must add it to your mode table.

Related concepts:

"CNOS processing in DB2" on page 909

Related information:

User-defined tables and data filter (VTAM Resource Definition Reference)

Sample mode entries:

The sample mode entries for IBMRDB contain some options that are necessary for dependent LUs to request VTAM sessions.

These options are listed in the following table.

COMPROT	PRIPROT	TSPROF	SECPROT

FMFPROF	PSERVIC	TYPE

The samples in the figure below work for both dependent and independent LUs. However, if you have no dependent LUs, it is not necessary to re-assemble your existing mode table with the above options. For your convenience, a sample MODEENT is included in data set DSN8VTAM, in SDSNSAMP.

The ENCR option is ignored by LU 6.2 and is thus not included in the DB2 samples.

```
DB2MODES MODETAB

IBMRDB   MODEENT LOGMODE=IBMRDB,     DB2 DEFAULT MODE FOR APP-DIR ACC  X
               TYPE=0,             NEGOTIABLE BIND                    X
               SSNDPAC=X'02',      SECONDARY SEND PACING COUNT        X
               SRCVPAC=X'00',      SECONDARY RECEIVE PACING COUNT     X
               RUSIZES=X'8989',    RUSIZES IN-4096 OUT-4096           X
               FMPROF=X'13',       LU6.2 FM PROFILE                   X
               TSPROF=X'07',       LU6.2 TS PROFILE                   X
               PRIPROT=X'B0',      LU6.2 PRIMARY PROTOCOLS            X
               SECPROT=X'B0',      LU6.2 SECONDARY PROTOCOLS          X
               COMPROT=X'50A5',    LU6.2 COMMON PROTOCOLS             X
               PSERVIC=X'06020000000000000122F00'    LU6.2 LU TYPE
         MODEEND
         END
```

Figure 121. Sample mode entries

MODEENT options:

When you define your DB2 subsystem to VTAM, you can specify mode values. As you choose these values, be aware that partner systems can have different values. VTAM negotiates different values to limits that are acceptable to both systems when the session is established for the mode.

The options that are used in the MODEENT macro have the following meanings.

name
: The *name* option (IBMDB2LM and IBMRDB in the examples) is optional and has no function in the specification of a logon mode table.

LOGMODE
: Specifies the logon mode name that is to be used as a key for the session options in this table entry. This logon mode name corresponds to mode name columns in the CDB.

TYPE
: TYPE=0 indicates that DB2 is using a negotiable BIND, which is required for communicating with dependent LUs.

SRCVPAC
: Specifies the secondary receive pacing count. The DB2 suggested value is X'00'.

SSNDPAC
: Specifies the secondary send pacing count. The DB2 suggested value is any nonzero number. Do not use 0; this turns off pacing, which can result in problems with IOBUF storage.

RUSIZES
: Specifies the maximum length of data in bytes that can be sent and received in one *request/response unit* (RU). It is read as two numbers, each having two

hexadecimal digits. The first number is for the send amount. The second number is for the receive amount. The suggested value of X'8989' means that VTAM sends a maximum of 4096 bytes (8×2^9) across at one time, but there is no limit to how much total information can be sent.

FMPROF
This constant specifies the function management profile that is required for LU 6.2.

TSPROF
This constant specifies the transmission services profile that is required for LU 6.2.

PRIPROT
This constant specifies the primary LU protocols that are used in LU 6.2.

SECPROT
This constant specifies the secondary LU protocols that are used in LU 6.2.

COMPROT
This constant specifies the common LU protocols that are used in LU 6.2.

PSERVIC
This constant specifies this LU as an LU type 6.2.

Some of the above options can have a profound effect on performance because of their impact on pacing.

The ENCR option is not included in the sample above because it is ignored by LU 6.2.

Related concepts:

"Session-level pacing" on page 901

Populating the communications database

If you plan to use DB2 only as a server, you do not need to populate the CDB; default values are used. However, if you intend for DB2 to act as a requestor, you need to insert one row for each remote system into SYSIBM.LOCATIONS and SYSIBM.LUNAMES.

About this task

For example, Spiffy's USIBMSTODB21 subsystem works as a server for many Windows requesters. It is not necessary for Spiffy to register all those requesters in DB2's CDB.

You do not need to populate table SYSIBM.LULIST unlessDB2 is acting as a requester of data that resides in a data sharing group.

After you populate these tables, you can write queries that access data at a remote system.

Related concepts:

Support for executing long SQL statements in a distributed environment (DB2 Application programming and SQL)

SYSIBM.LULIST (DB2 Data Sharing Planning and Administration)

Related tasks:

Granting privileges to remote users (Managing Security)

VTAM-relevant columns of the SYSIBM.LOCATIONS table

When you issue an SQL CONNECT statement, the LOCATION column of the SYSIBM.LOCATIONS table correlates the location name (or DRDA RDBNAM) with the VTAM LU name. If necessary, the location name is also correlated with the transaction program names (TPNs).

LOCATIONS has the following columns relating to VTAM:

DBALIAS VARCHAR(128) NOT NULL
The name that is associated with the server. This name is used to access a remote database server. If DBALIAS is blank, the location name is used to access the remote database server. This column does not change database object names that are sent to the remote site using a location qualifier. Use the DBALIAS column to access data at two or more different remote locations when those remote locations have the same name. The LOCATION specifies where the database is in the network, and the DBALIAS is used to access the database server. This column does not change database object names that are executed in the application using the LOCATION. All fully qualified table names must reference the server's LOCATION name or one of its server LOCATION alias names, otherwise the SQL statement will fail because the table does not exist.

LOCATION CHAR(16)
The unique network location name, or DRDA RDBNAM, assigned to a system, remote or local. You must provide location names for any systems that you request data from. This column is the primary key for this table. If the remote LU exists in the same VTAM domain, specify the APPL name, not the ACBNAME. DBALIAS can override this name.

LINKNAME CHAR(8)
Identifies the VTAM attributes associated with this location. For each LINKNAME specified, you must have a row in SYSIBM.LUNAMES whose LUNAME matches the value specified in this column. Because this table is used for outbound requests, you must provide an LUNAME or your requests fail. Do not enter blanks in this column.

TPN VARCHAR(64)
This column is used to enter a transaction program name (TPN) for SNA conversations with non-DB2 systems. You only need to use this column if you are sending or receiving SQL requests from systems using non-default TPNs.

Spiffy's USIBMSTODB21 location wants a LOCATIONS table that looks like the following table.

Table 166. Spiffy's LOCATIONS table

LOCATION	LINKNAME	TPN
USIBMSTODB21	LUDB21	

Table 166. Spiffy's LOCATIONS table (continued)

LOCATION	LINKNAME	TPN
USIBMSTODB22	LUDB22	
USIBMSTOSQL1	LUSQLDS	TPNSQLDS2
USIBMSTOSQL2	LUSQLDS	TPNSQLDS1

For example, add the second row with this statement:

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
  VALUES ('USIBMSTODB22','LUDB22');
```

A row for the local location: You do not need a row for the local DB2 in the LUNAMES and LOCATIONS tables. For example, Spiffy's USIBMSTODB21 subsystem does not require a row that shows its own LU name and location name. However, for convenience, Spiffy decides to populate one LUNAMES table and one LOCATIONS table and to duplicate them entirely at each location. As a result, each table contains a row for its own LU name or location name.

VTAM-relevant columns of the SYSIBM.LUNAMES table

The LUNAMES table defines the security and mode requirements for conversations with other systems. Decisions about how to populate this table depend on how you intend to use DB2.

- If you use this system only as a server, DB2 can use a blank in the LUNAME column as a default. DB2 uses the values in the default row as defaults for LUs that are not explicitly defined in LUNAMES. If you do not have a row with a blank in the LUNAME column, DB2 rejects client connections that do not explicitly state a valid LUNAME. The DSNTIJSG installation job creates the default row in table SYSIBM.LUNAMES.
- If this DB2 requests data from other systems, you need to provide LU names for those systems.If the remote LU exists in the same VTAM domain, specify the APPL name, not the ACBNAME.

The LUNAMES table has the following columns:

LUNAME CHAR(8)

The LU name of the remote system. The default of 8 blanks indicates that this row is used for serving the requests of any system that is not specifically listed in the LUNAMES table. For example, because USIBMSTODB21 acts strictly as a server for many Windows requesters, Spiffy leaves the LUNAME column blank for those requesters and uses default values for the entire row.

However, you must provide LU names for any remote system that uses different values from the defaults.

SYSMODENAME CHAR(8)

This column is ignored for DRDA access conversations.

SECURITY_IN CHAR(1)

Defines the security options that are accepted by this DB2 subsystem when an SNA client connects to DB2. The default, A, means that an incoming connection request is accepted if it includes any of these:

- A user ID
- A user ID and password
- A user ID and RACF PassTicket

- A Kerberos security ticket.

SECURITY_OUT CHAR(1)
Defines the security option that is used when local DB2 SQL applications connect to any remote server associated with this LUNAME. The default, A, means that outgoing connection requests contain an authorization ID without a password.

ENCRYPTPSWDS CHAR(1)
For now, Spiffy uses a blank to indicate no encryption of passwords.

MODESELECT CHAR(1)
Determines whether to use the default mode or to choose a mode from the MODESELECT table. Spiffy uses a blank to use the default IBMRDB mode for conversations using DRDA access.

USERNAMES CHAR(1)
This column is used for inbound and outbound requests to control authorization ID translation.

Spiffy uses a blank to indicate that no authorization IDs are translated, and also that no passwords are sent to the server.

GENERIC CHAR(1)
A Y in this column indicates that a generic LU name is to be used for CNOS processing and SQL requests sent to the partner LU. A value of N or a blank indicates that the name specified in the LUNAME column is to be used.

Spiffy's USIBMSTODB21 location wants a LUNAMES table that looks like the following table.

Table 167. Spiffy's SYSIBM.LUNAMES table. The row of blanks is a default row that Spiffy intends to use for Windows requesters in its initial testing.

LUNAME	SYSMODENAME	USERSECURITY[1]	ENCRYPTPSWDS	MODESELECT	USERNAMES
LUDB21					
LUDB22					
LUSQLDS					
(blanks)					

Note:

1. USERSECURITY refers to SECURITY_IN AND SECURITY_OUT

Spiffy can use an SQL INSERT statement to add the appropriate rows. For example, they add the LU name for USIBMSTODB22 with this statement:

```
INSERT INTO SYSIBM.LUNAMES (LUNAME)
  VALUES ('LUDB22');
```

Related concepts:

SYSIBM.LUNAMES (DB2 Data Sharing Planning and Administration)

Related tasks:

"Associating applications with modes" on page 903

Related reference:

SYSIBM.LUNAMES columns (Managing Security)

SYSIBM.USERNAMES columns (Managing Security)

VTAM-relevant columns of the SYSIBM.USERNAMES table

SYSIBM.USERNAMES contains information that is needed for outbound and inbound ID translation and also for "come from" checking.

Related reference:

SYSIBM.USERNAMES columns (Managing Security)

Starting VTAM to use DB2

You do not need to code any special VTAM start options to use DB2, but you can tailor start option values for DB2 communications.

About this task

You must start VTAM before starting DDF.

Two VTAM libraries must contain definitions for DB2:

- SYS1.VTAMLST contains the definitions that define DB2 as a VTAM application.
- SYS1.VTAMLIB contains mode table definitions used by DDF. This must be an APF-authorized library, or in a concatenation of APF-authorized libraries.

Procedure

To enable VTAM to use DB2:

You can use the following VTAM command, assuming that the member DB2APPLS contains definitions for DB2:

```
V NET,ACT,ID=DB2APPLS
```

Related concepts:

"The MODEENT macro" on page 893

Related tasks:

Starting the Domain (VTAM Network Implementation Guide)

"Defining the DB2 subsystem to VTAM" on page 888

Related reference:

VTAM Start Options (VTAM Resource Definition Reference)

Tuning the VTAM system

As you begin testing with the DB2 distributed data facility, you probably need to modify VTAM options and CDB values to handle certain potential problems.

About this task

You should consult a VTAM communications expert to tune your network. Consider the following tasks when tuning VTAM for DDF.

- Controlling buffer storage

  By sending large amounts of data through the network, DB2 can cause problems with your VTAM I/O buffer pool.
- Controlling pacing

  You probably need to tune your pacing options if your VTAM buffers become overloaded with data that is sent to this local DB2.
- Modifying default session limits

  Consider modifying session limits if you have problems obtaining enough sessions to handle your distributed workload efficiently.
- Modifying class of service

  Specifying a class of service can help you assign priorities to your network applications.
- Associating applications with modes

  Tuning the system can require that you add new modes to your log mode table so that there is a greater variety of classes of service available for your sessions. This variety allows you to have more flexibility in tuning the system for specific uses. This topic tells you how to associate specific sessions with modes.

Before you begin tuning the network, you must understand the relationship between VTAM options and associated values in DB2's CDB. The table below summarizes the relationship.

Table 168. Relationship between DB2's CDB and VTAM macros

Macro Name	Option	CDB table.column	Relationship
APPL	*name*	LOCATIONS.LINKNAME LUNAMES.LUNAME LUMODES.LUNAME MODESELECT.LUNAME USERNAMES.LINKNAME LULIST.LINKNAME	The LU name used in VTAM communication. This name maps 1:1 to the system's location name in LOCATIONS.
APPL	DSESLIM	LUMODES.CONVLIMIT	CONVLIMIT overrides session limits specified with DSESLIM. Session limit values are used in CNOS processing.
MODEENT	LOGMODE	LUMODES.MODENAME	LUMODES creates session limits for *specific* LU name and mode name combinations.
MODEENT	LOGMODE	MODESELECT.MODENAME	MODESELECT maps authorization IDs and plans to specific modes.

Related tasks:

Improving performance for applications that access distributed data (DB2 Performance)

Controlling buffer storage

VTAM uses buffer pools for control blocks, network traffic data, and channel programs. A shortage of buffer pools can have an adverse effect on VTAM CPU time, storage consumption, and the ability to serve DB2 requests.

Procedure

To control buffer storage:

- Monitor VTAM buffer pools by using one of the following methods:
  - The VTAM command DISPLAY NET,BFRUSE
  - A VTAM trace, obtained by entering the following z/OS MODIFY command:

    `F procname,TRACE,TYPE=SMS,ID=VTAMBUF`

    Procname in the command is the VTAM start procedure name. The data is collected by the generalized trace facility (GTF).
- Use any of the following methods to prevent shortages of these VTAM buffers (IOBUFs):

  Increase the number of IOBUF buffers

  The IOBUF pool definition is one of the VTAM start options. You can enter the IOBUF option from the z/OS console, or you can include it at VTAM startup in SYS1.VTAMLST in member ATCSTRxx.

  Tuning the IOBUF pool encompasses both base allocation and dynamic expansion values. At installation, you can specify a base allocation for the IOBUF pool (in number of buffers) and a dynamic expansion (in number of buffers). When storage runs short in the buffer pool, VTAM temporarily expands the IOBUF pool by the dynamic expansion value, based on a trigger which you can also specify in VTAM definitions. Recommendation: Set a maximum size for the IOBUF pool size using the *xpanlim* start option for the buffer pool. If you turn off pacing accidentally, *xpanlim* prevents DB2 from causing VTAM to grab unlimited amounts of storage.

  Decrease the session level pacing count

  Pacing is vital for controlling the potentially large amounts of data that are transferred around the network.

  Decrease the number of concurrent conversations

  You can reduce the number of concurrent conversations by reducing the number of sessions.

  Decrease the request unit (RU) size

  The RUSIZES option is part of the mode entry statement.

  Because reducing the number of sessions and the RUSIZES value can adversely affect performance, you should first consider increasing IOBUF buffers and decreasing the session pacing count.

Related concepts:

"Session-level pacing"

"The MODEENT macro" on page 893

Related tasks:

"Calculating VTAM I/O buffer pool (IOBUF) storage" on page 907

Related reference:

Buffer pools (VTAM Network Implementation Guide)

Session-level pacing

Session-level pacing is the mechanism by which the receiver of data (DB2, in this case) can control the pace at which the sender sends data (in the form of RUs).

The pacing size is the number of RUs that VTAM sends across the line at one time, and you can set that value by using the VPACING option of the VTAM APPL definition statement. You set the RU size in the MODEENT macro. The receiving VTAM stores these RUs in its IOBUF pool; it uses pacing so that its buffers do not become flooded with data.

The pacing process works as shown in the following figure. The system at the sending side (assume it is USIBMSTODB22) passes data to its VTAM system. VTAM formats the data into RUs and sends those RUs across the network. If, for example, the pacing size is 2, then it sends two RUs. A 29-byte network header is sent with each RU.

After the USIBMSTODB22 VTAM system sends the specified number of RUs, it does not send any more data on this session until it receives a pacing response from the VTAM system at USIBMSTODB21. The USIBMSTODB21 VTAM system does not send a response until VTAM transfers the data into the DB2 buffers.

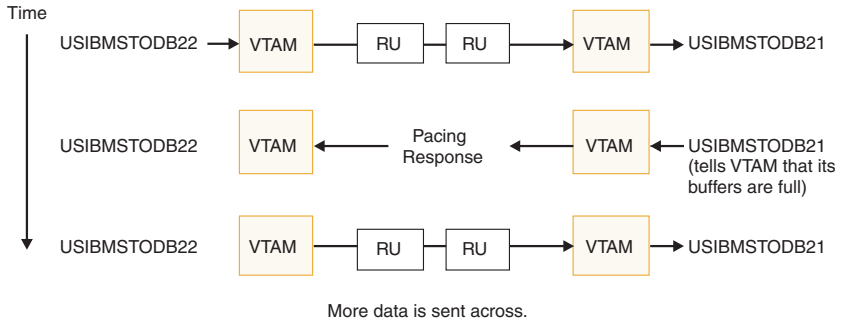


Figure 122. How pacing works

Although it is generally true that the receiving system can control inbound pacing, both communicating systems negotiate final pacing values.

Related concepts:

"The APPL statement" on page 888

"The MODEENT macro" on page 893

Related reference:

Session-level pacing (VTAM Network Implementation Guide)

Recommendation for APPL pacing option:

The VPACING option of the APPL statement determines the maximum number of RUs that another LU can send, on a session, to this LU. When the maximum number of RUs is sent, the other LU waits to receive a pacing response.

Specify a nonzero value for this option unless you want to turn off all pacing for all sessions that are affected by this option.

Recommendation: Start with a value of 2 for both communicating systems. This pacing size is the same in both directions for all modes.

The VPACING value is used with the RUSIZES option of the MODEENT macro to control the pacing window size. Thus, if VPACING is 2 and RUSIZES is 4–KB (X'8989'), then about 2 × 4KB = 8KB are sent before waiting to receive a pacing response. You should verify that your VTAM buffer pools are large enough to accommodate the chosen pacing and RU sizes.

Related tasks:

"Calculating VTAM I/O buffer pool (IOBUF) storage" on page 907

Recommendation for MODEENT pacing options:

The MODEENT macro contains several pacing options.

PSNDPAC
This does not apply to DB2; therefore, you can ignore this option.

SSNDPAC
This option is really a flag that you set to either 0 (off) or nonzero (on). When 0, outbound pacing for sessions is disabled, which can lead to severe problems with IOBUF storage.

Recommendation: Specify a nonzero value for this option.

SRCVPAC
If 0, which is the recommended value, the VPACING option of the VTAM APPL statement controls both the send and receive pacing for all sessions in all modes. A value of 0 makes it easier for you to predict pacing results and makes it easier to maintain your pacing definitions.

If nonzero, VPACING controls pacing in one direction, and SRCVPAC controls it in another. LU 6.2 protocols make it difficult to predict which option is in control at any given time.

Related concepts:

"The MODEENT macro" on page 893

Modifying class of service

You can define the transmission priority and paths between systems with entries in a class of service (COS) table.

About this task

Each entry in the table is associated with a list of routes to be used with a particular class. For example, you might want to place interactive sessions on a faster route than a batch job.

Procedure

To specify a class of service for a specific mode:

Use the COS option of the MODEENT macro. When you specify a name of a COS entry in the mode description, you select the list of routes that you want to be used for the session. When VTAM establishes the session, it chooses the first available route in the list of routes that you tell it to use for that class.
If you do not specify a COS name, the mode gets the default list of routes from VTAM.

Associating applications with modes

As you tune your system, you can assign certain applications, such as a high-priority job, to the mode that is best suited for that job.

About this task

GUPI

You can also use a specific mode assignment for an application that uses many conversations. You can assign such an application to a mode that allows more conversations than the VTAM DSESLIM value you entered in the APPL statement.

To associate a specific mode with a particular session, you need to update or insert rows into three tables in the CDB: LUNAMES, LUMODES, and MODESELECT.

GUPI

Related concepts:

"When updates to CDB values take effect for VTAM connections" on page 907

Update LUNAMES to associate modes with LU names:

The LUNAMES table associates a mode with each remote system that the local subsystem can send a query to.

About this task

GUPI

You can specify the following type of connections in this table:

SQL processing conversations
For DRDA access, an SQL processing conversation is established. The mode name for SQL processing conversations is determined by the MODESELECT table of the CDB. If the MODESELECT column of LUNAMES table is blank or contains N, then the default mode (IBMRDB) is used. If it contains a Y, then MODESELECT is searched.

GUPI

Related tasks:

"Populating the communications database for use with TCP/IP" on page 925

Update SYSIBM.LUMODES with conversation limits:

Use SYSIBM.LUMODES to provide VTAM with conversation limits for specific LU name and mode name combinations.

About this task

GUPI

Populating SYSIBM.LUMODES is optional; if you do not specify mode names in this table, the VTAM defaults are used. The table is unlike the DSESLIM option of the VTAM APPL definition statement, which provides the default session limits for *all* LU name and mode name combinations. The primary key for this table is formed by the LU name and mode name combination. Only one entry with the same LU name and mode name is allowed.

LUMODES is accessed for negotiation of session limits with a remote DB2 for a specific mode. This negotiation is called *change number of sessions* (CNOS).

Example

For example, suppose Spiffy wants to allocate 75 sessions instead of 50 (the value in DSESLIM) for conversations to USIBMSTODB22, using the mode named LOC2MODE. They can use the INSERT statement below to update the value in the CONVLIMIT column to 75. The new session limit takes effect the next time DDF is started, or in the initial connection to this LU for this mode.

```
INSERT INTO SYSIBM.LUMODES VALUES ('LUDB22','LOC2MODE',75,'N');
```

CNOS processing negotiates a value that is the lesser of the number of sessions available at either system for that mode. Therefore, USIBMSTODB22 must also increase its CONVLIMIT value to derive any benefit from the added sessions.

Columns of the LUMODES table:

LUNAME CHAR(8)
Again, this is the LU name of the other system. This column is a foreign key of the LUNAMES table; thus, all LU names defined in this table must be defined in LUNAMES. When you delete an LU name from the LUNAMES table, all associated rows in LUMODES are deleted.

MODENAME CHAR(8)
The name of the logon mode description in the VTAM logon mode table that VTAM uses when creating a conversation to support the local DB2's request for data from another system. The mode named here must exist in the mode table used by DB2 before a conversation can be created between USIBMSTODB21 and USIBMSTODB22.

CONVLIMIT SMALLINT
The maximum number of conversations to be concurrently active between this DB2 subsystem and the other system for this mode. This number overrides the number in the DSESLIM option of the VTAM APPL definition statement during CNOS processing..

GUPI

Related concepts:

"CNOS processing in DB2" on page 909

Update SYSIBM.MODESELECT to associate plans with modes:

SYSIBM.MODESELECT correlates authorization IDs and plan names with mode names. The primary key for this table is the combination of AUTHID, LUNAME, and PLANNAME.

About this task

GUPI

Only one entry with the same AUTHID, LUNAME, and PLANNAME is allowed.

Use this table to make sure that certain authorization IDs using certain plans always have a predefined class of service suited for that operation. For example, the USIBMSTODB21 location might want to work with USIBMSTODB22 to set up a high performance mode for DBADM to run queries to USIBMSTODB22. After the following statement is committed, all subsequent threads to USIBMSTODB22 use mode DB2MODE1 to process SQL processing conversations:

```
INSERT INTO SYSIBM.MODESELECT VALUES ('DBADM',' ','LUDB22','DB2MODE1');
```

Populating this table is optional. If the remaining columns are blank for any given LU name, then the mode name applies to all authorization IDs for all PLANNAMEs accessing the given LU name.

Columns of the MODESELECT table:

AUTHID CHAR(8)
The authorization ID of the request for data from another system. A blank AUTHID indicates that the specified mode name applies to all authorization IDs. Blank is the default.

PLANNAME CHAR(8)
The plan name associated with the request for data from another system. A blank plan name indicates that the specified mode name applies to all plan names. Blank is the default.

LUNAME CHAR(8)
The LU name to which the specific mode name applies. This column is a foreign key of the LUNAMES table; therefore, all LU names defined in this table must be defined in LUNAMES.

MODENAME CHAR(8)
The name of the logon mode description in the VTAM logon mode table that is used when creating a conversation to support the request for data from another system. If this column is blank, the default mode (IBMDB2LM or IBMRDB) is used.

How an SQL processing conversation mode is chosen:

The MODESELECT table of the CDB is used to choose a mode for an SQL processing conversation (if the MODESELECT column of the LUNAMES table contains Y for this LU name). The table below shows the search order of the MODESELECT table.

Table 169. Precedence search order for MODESELECT table of CDB

AUTHID	PLANNAME	Result
Name	Name	The MODENAME applies to the named AUTHID for the named PLANNAME accessing the named LU.
Name	Blank	The MODENAME applies to the named AUTHID for all PLANNAMEs accessing the named LU.
Blank	Name	The MODENAME applies to all AUTHIDs for the named PLANNAME accessing the named LU.
Blank	Blank	The MODENAME applies to all AUTHIDs for all PLANNAMEs accessing the named LU.

If the MODESELECT column of the LUNAMES table contains Y for a particular LU name and no row is found for that LU name in the MODESELECT table, then you receive a negative SQL return code when trying to access the system at that LU.

Plan name for remote bind operations: If you want to specify a particular mode for remote bind operations, use the plan name DSNBIND in MODESELECT.

GUPI

When updates to CDB values take effect for VTAM connections

Any table in the CDB can be updated while DDF is active.

The changes take effect as follows:

- Changes to LUMODES take effect the next time DDF is started, or on the initial session to a given LUMODE combination.
- Changes to LUNAMES, LOCATIONS, and LULIST take effect as follows:
  - If DDF has not yet tried to communicate with a particular remote location, rows added to LUNAMES and LOCATIONS take effect when DDF attempts to communicate with that location.
  - If DDF has already attempted communication with a particular location, rows added to LUNAMES and LOCATIONS take effect the next time DDF is started.
- Changes to USERNAMES and MODESELECT take effect at the next thread access.

In all cases, existing conversations continue to operate as the table specified before the update.

The process of modifying the CDB, particularly MODESELECT and USERNAMES, can interfere with DDF's access to the tables. This could potentially cause deadlocks and timeouts, which cause the attempted access to the remote system to fail.

Calculating session limits

You can calculate session limit values for DRDA access.

About this task

You might have to derive a precise figure for your session limits (DSESLIM). For example, if you specify a very large number for your session limits and you are running short of space, it might help to calculate a number closer to what you actually need.

Procedure

To calculate the session limit for a specific mode:

For DRDA access, determine the maximum number of concurrently active applications that use the mode to and from the remote system.

Example

For example, suppose location A has a maximum of 3 DRDA access applications that can run concurrently on Mode1 to other locations. Also suppose that a maximum of 10 DRDA access applications can be incoming to location A on Mode1. In this case, Mode1 should support 13 sessions for DRDA access applications.

Calculating VTAM I/O buffer pool (IOBUF) storage

You can estimate VTAM buffer pool storage when you use the distributed data facility of DB2. Every *path information unit* (PIU) that enters or leaves VTAM resides in one or more IOBUF buffers.

About this task

A PIU is composed of a 26-byte transmission header, a 3-byte request/response header, and the request/response unit (RU) that contains VTAM application data. You define the length of the RU in a mode entry, using the RUSIZES option.

Procedure

To calculate the maximum number of buffers that are required for the local DB2 subsystem:

1. Calculate the number of buffers that each PIU occupies, and call it PIUBUF.

   ```
   PIUBUF = CEILING(( 29 + RUSIZE ) / BUFSIZE)
   ```

   RUSIZE is the length of the RU in bytes. It is assumed to be the same for both session directions. BUFSIZE is the value you specified in the IOBUF pool definition.

   Assume you have a buffer size of 441 bytes, and an RUSIZE of 4096. With these values, PIUBUF would be 10 ((29+4096) / 441, rounded up).

   For channel-to-channel (CTC) and NCP connections, you need to be concerned with the VTAM MAXBFRU value. For CTC connections, MAXBFRU is the number of 4KB buffers allocated to hold the PIUs sent over the channel. If your RU size is 4096 and you allow 29 bytes for the header, then you need to allocate at least 2 4KB buffers. Thus, you need a MAXBFRU value of at least 2.

   When you route data through NCP, MAXBFRU is the number of VTAM IOBUF buffers allocated to hold the PIUs sent to the NCP, which means MAXBFRU must be at least as large as PIUBUF.

2. Calculate the maximum number of IOBUF buffers used by a session, and call it SESSBUF.

   ```
   SESSBUF = PACECNT × PIUBUF
   ```

   PACECNT stands for pacing count. For this example, assume that pacing is the same in both directions, and it is the same for all modes.

   If pacing is set to 2, then SESSBUF is 20.

3. Calculate the maximum number of sessions that can be active for all modes to all systems and call it SESCNT. Calculate the maximum for each mode, then add those results to get SESCNT.

4. Calculate the maximum number of VTAM buffers used by DB2, and call this DB2BUF. The formula for DB2BUF is based on a worst case scenario, because it assumes that all sessions are used by concurrent conversations.

   ```
   DB2BUF = SESCNT × SESSBUF
   ```

   If you assume that the maximum number of sessions that can be active is 50 (SESCNT), then 1000 is the number of IOBUF entries required by DB2 in a worst case scenario.

5. Calculate actual VTAM buffer storage consumption used by DB2, and call it STORAGE.

   ```
   STORAGE = DB2BUF × (BUFSIZE + 71)
   ```

   Each buffer includes 71 bytes for VTAM internal headers.

   So, to continue the above example, an upper value of real storage is estimated as follows:

   ```
   1000 × (441 + 71) = 500KB
   ```

Related concepts:

"Session-level pacing" on page 901

"Definitions for channel-connected DB2 subsystems" on page 912

"NCP-connected DB2 subsystems" on page 914

Related tasks:

"Calculating session limits" on page 907

CNOS processing in DB2

The distributed data facility of DB2 can request to alter the number of sessions with another system for a specific VTAM logon mode. This automatic process is called *change number of sessions* (CNOS).

This topic contains a brief overview of the process as it relates to DB2; it should help you understand the messages that CNOS processing generates.

When sessions are started: The AUTOSES option of the VTAM APPL determines whether, and how many, sessions are started at the time CNOS is negotiated. If AUTOSES is 0, then the sessions are not started at CNOS negotiation time; they are started as they are needed. AUTOSES should generally not be 0, because then DB2 is not informed if CNOS fails, and you receive a "resource unavailable" SQL code with the first SQL request to the remote system.

If AUTOSES is not 0, then sessions are started as follows:

- If AUTOSES is equal to or less than the number of contention winner sessions for a specific DB2 subsystem, then the number of sessions that are automatically started at CNOS negotiation is equal to AUTOSES.
- If AUTOSES is greater than the number of contention winner sessions for a specific DB2 subsystem, only the contention winner sessions are automatically started at CNOS negotiation.

Each LU has its own value for the number of contention winner sessions to start. The total number of sessions started on behalf of a CNOS negotiation request is the sum of the sessions started at each site.

Example: Suppose the DB2 subsystems at USIBMSTODB21 and USIBMSTODB22 have the following values in their VTAM APPL statements and LUMODES tables:

USIBMSTODB21
APPL Values

DSESLIM	50
DMINWNL	25
DMINWNR	25
AUTOSES	1

USIBMSTODB22
APPL Values

DSESLIM	40
DMINWNL	20
DMINWNR	20
AUTOSES	1

USIBMSTODB21 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB22	SYSTOSYS	2
LUDB22	IBMDB2LM	80

USIBMSTODB22 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB21	SYSTOSYS	2
LUDB21	IBMDB2LM	50

Figure 123. CNOS negotiation example: VTAM and DB2 definitions

Assume that USIBMSTODB21's DDF is started first. CNOS processing fails because USIBMSTODB22's DDF has not yet started, and you get a message at the console. When USIBMSTODB22's DDF is started, CNOS processing can begin.

USIBMSTODB21 sends to USIBMSTODB22 a CNOS value of 80, which is its CONVLIMIT value. However, USIBMSTODB22 replies with a value of 40 (its DSESLIM value), and, as shown in the following figure, that becomes the negotiated value for the CNOS that is started by USIBMSTODB21. Both systems begin starting the number of sessions that are specified in their respective AUTOSES options.

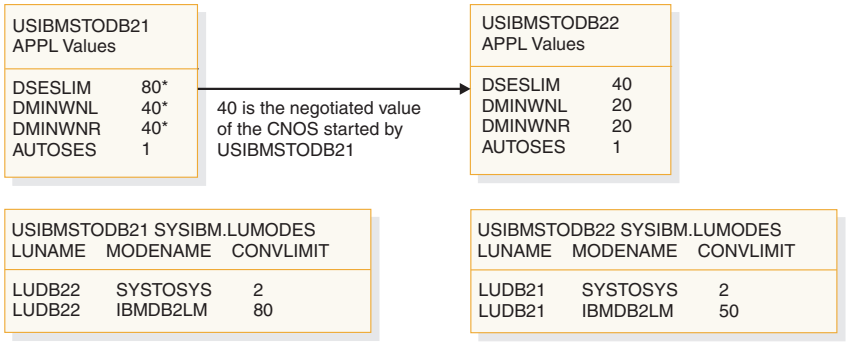


Figure 124. Result of CNOS negotiation started by USIBMSTODB21. Overridden values are noted with asterisks (*).

VTAM does not start all 40 sessions unless the two AUTOSES values total up to 40 or greater. Instead, VTAM delays starting the other sessions until they are needed.

Everything up to this point occurred because USIBMSTODB21 issued CNOS. Now, as shown in the following figure, USIBMSTODB22 starts CNOS processing back to USIBMSTODB21 because its CDB has CNOS limits specified (50 in CONVLIMIT). USIBMSTODB21's VTAM sees that DB2 allows up to 80, so VTAM sends the CNOS reply message back to USIBMSTODB22 unchanged (50). USIBMSTODB22's CONVLIMIT value of 50 is compared with USIBMSTODB21's overridden value of 80 from the previous CNOS, and 50 is chosen as the value.

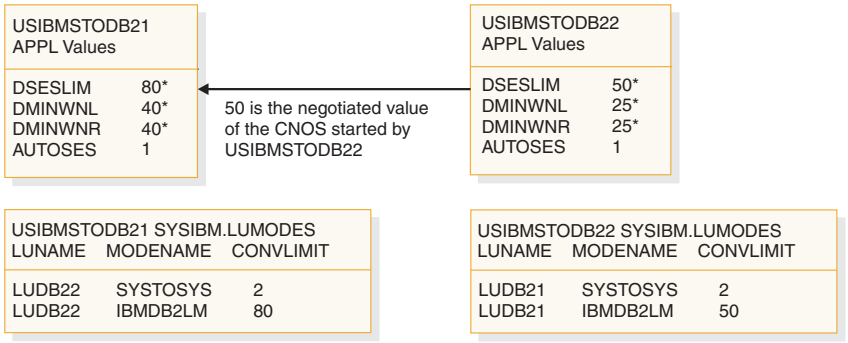


Figure 125. CNOS negotiation from USIBMSTODB22 to USIBMSTODB21. Overridden values are noted with asterisks (*).

If the new negotiated value is smaller than the number already started by USIBMSTODB21, then VTAM terminates the number of sessions that makes up the difference. If the CONVLIMIT value at USIBMSTODB22 is 20, for example, VTAM terminates 20 sessions on behalf of the request from USIBMSTODB22 because the lowest negotiated value always wins. If a session is currently being used by a conversation, the session is terminated as soon as the conversation is deallocated.

Related reference:

Managing Sessions (z/OS Communications Server: SNA Programmer's LU 6.2 Guide)

Sample VTAM definitions to connect two DB2 subsystems

Sample VTAM definitions are provided to give you some guidance on setting up your network to connect two DB2 subsystems. The samples are not intended to give you information about all the options.

This set of definitions includes the basic definitions you need to connect two DB2 subsystems. Additional options for channel-to-channel and Network Control Program (NCP) connections are covered as well.

Basic VTAM definitions

Some basic definitions are required for all VTAM connections.

These definitions are listed below.

```
*******************************************************************
**      APPL STATEMENT FOR SYSTEM 1                              *
*******************************************************************
DB1APPL  APPL APPC=YES,                                          X
         ATNLOSS=ALL,                                            X
         AUTH=(ACQ),                                             X
         AUTOSES=1,                                              X
         DSESLIM=20,                                             X
         DMINWNL=10,                                             X
         DMINWNR=10,                                             X
         PRTCT=DB1PWD,                                           X
         SECACPT=ALREADYV,                                       X
         EAS=509,                                                X
         MODETAB=DB2MODES,                                       X
         PARSESS=YES,                                            X
         SRBEXIT=YES,                                            X
         SYNCLVL=SYNCPT,                                         X
         VPACING=2
*******************************************************************
**      APPL STATEMENT FOR SYSTEM 2                              *
*******************************************************************
DB2APPL  APPL APPC=YES,                                          X
         ATNLOSS=ALL,                                            X
         AUTH=(ACQ),                                             X
         AUTOSES=1,                                              X
         DSESLIM=20,                                             X
         DMINWNL=10,                                             X
         DMINWNR=10,                                             X
         PRTCT=DB2PWD,                                           X
         SECACPT=ALREADYV,                                       X
         EAS=509,                                                X
         MODETAB=DB2MODES,                                       X
         PARSESS=YES,                                            X
         SRBEXIT=YES,                                            X
         SYNCLVL=SYNCPT,                                         X
         VPACING=2
*********************************************************************
**    LOGMODE TABLE FOR SYSTEM 1 TO SYSTEM 2 CONNECTIONS
*********************************************************************
DB2MODES MODETAB
*******************************************************************
*     LU6.2 SERVICES MANAGER BASE MODE FOR APPC/VTAM
*******************************************************************
SNASVCMG MODEENT LOGMODE=SNASVCMG,                               X
```

```
                FMPROF=X'13',                                   X
                TSPROF=X'07',                                   X
                PRIPROT=X'B0',                                  X
                SECPROT=X'B0',                                  X
                PSERVIC=X'060200000000000000000300',            X
                COMPROT=X'D0B1',                                X
                RUSIZES=X'8585',                                X
                ENCR=B'0000'
*********************************************************************

*********************************************************************
** DB2 DEFAULT FOR SYSTEM 1 AND SYSTEM 2 DRDA ACCESS               **
*********************************************************************
IBMRDB MODEENT LOGMODE=IBMRDB,                                     X
               TYPE=0,                                             X
               PSNDPAC=X'00',                                      X
               SSNDPAC=X'02',                                      X
               SRCVPAC=X'00',                                      X
               RUSIZES=X'8989',                                    X
               FMPROF=X'13', LU6.2 FM PROFILE                      X
               TSPROF=X'07', LU6.2 TS PROFILE                      X
               PRIPROT=X'B0', LU6.2 PRIMARY PROTOCOLS              X
               SECPROT=X'B0', LU6.2 SECONDARY PROTOCOLS            X
               COMPROT=X'50A5', LU6.2 COMMON PROTOCOLS             X
               PSERVIC=X'06020000000000000122F00' LU6.2 LU TYPE
*********************************************************************
** ATCSTRTA VTAM START OPTIONS FOR SYSTEM 1, INCLUDES IOBUF
*********************************************************************
  CONFIG=TA,                                                        X
  SSCPID=53,MAXSUBA=150,HOSTSA=53,                                  X
  SSCPNAME=SSCP004,NETID=USIBMSY,                                   X
  IOBUF=(328,441,20,,64,48,768)
*********************************************************************
** ATCSTRTB VTAM START OPTIONS FOR SYSTEM 2, INCLUDES IOBUF
*********************************************************************
  CONFIG=TB,                                                        X
  SSCPID=54,MAXSUBA=150,HOSTSA=54,                                  X
  SSCPNAME=SSCP00E,NETID=USIBMSY,                                   X
  IOBUF=(328,441,20,,64,48,768)
```

Figure 126. Basic VTAM definitions

Related reference:

VTAM Application Program Major Node (VTAM Resource Definition Reference)

VTAM Start Options (VTAM Resource Definition Reference)

VTAM Logon Mode Table (VTAM Resource Definition Reference)

Definitions for channel-connected DB2 subsystems

When determining your channel-to-channel definitions, remember that MAXBFRU must be large enough to handle the largest PIU.

Because DB2 is sending 4096 bytes, you need enough 4KB buffers to accept 4096 + 29 bytes (the 29 bytes is for the network header). Thus MAXBFRU must be at least 2 in our example.

In many cases, the DB2 RU size is larger than any other PIUs used on existing CTCs, which can mean you must examine your MAXBFRU values on existing CTC definitions. If the values are too small, you get an SNA X'800A' sense code,

indicating that the PIU was truncated during transmission.

```
**********************************************************************
**       CTC DEFINITIONS FOR SYSTEM 1                               *
**********************************************************************
DB1CTC   VBUILD TYPE=CA          CTC MAJOR NODE DEFINITION
DB1GRPB  GROUP  LNCTL=CTCA,      CTCA LINE TYPE                      X
                MIH=YES,REPLYTO=10.0
DB1CTCL  LINE   ADDRESS=(500),   CTC ADDRESS FOR THIS LINE           X
                DELAY=0,         CTC DELAY                           X
                MAXBFRU=8,       MAX BUFFER USED                     X
                ISTATUS=ACTIVE    INITIAL STATUS IS ACTIVE
DB1CTCP  PU     ISTATUS=ACTIVE
**********************************************************************
**       CTC DEFINITIONS FOR SYSTEM 2                               *
**********************************************************************
DB2CTC   VBUILD TYPE=CA          CTC MAJOR NODE DEFINITION
DB2GRPB  GROUP  LNCTL=CTCA,      CTCA LINE TYPE                      X
                MIH=YES,REPLYTO=10.0
DB2CTCL  LINE   ADDRESS=(500),   CTC ADDRESS FOR THIS LINE           X
                DELAY=0,         CTC DELAY                           X
                MAXBFRU=8,       MAX BUFFER USED                     X
                ISTATUS=ACTIVE    INITIAL STATUS IS ACTIVE
DB2CTCP  PU     ISTATUS=ACTIVE
**********************************************************************
**       PATH - NETWORK ROUTES FOR SYSTEM 1                         *
**********************************************************************
MVSDB2   PATH DESTSA=2,ER1=(2,1),VR1=1,                              X
             VRPWS10=(2,30),VRPWS11=(2,30),VRPWS12=(2,30)
**********************************************************************
**       PATH - NETWORK ROUTES FOR SYSTEM 2                         *
**********************************************************************
MVSDB1   PATH DESTSA=1,ER1=(1,1),VR1=1,                              X
             VRPWS10=(2,30),VRPWS11=(2,30),VRPWS12=(2,30)
**********************************************************************
**      CDRSC DEFINITIONS FOR SYSTEM 1                              *
**********************************************************************
         VBUILD TYPE=CDRSC
DB2APPL  CDRSC CDRM=DB2CDRM,ISTATUS=ACTIVE
**********************************************************************
**      CDRSC DEFINITIONS FOR SYSTEM 2                              *
**********************************************************************
         VBUILD TYPE=CDRSC
DB1APPL CDRSC CDRM=DB1CDRM,ISTATUS=ACTIVE
**********************************************************************
**      CDRM DEFINITIONS FOR SYSTEM 1 AND 2 (SAME DEFINITION USED)    *
**********************************************************************
         VBUILD TYPE=CDRM
DB1CDRM CDRM SUBAREA=1,ISTATUS=ACTIVE,CDRSC=OPT
DB2CDRM CDRM SUBAREA=2,ISTATUS=ACTIVE,CDRSC=OPT
**********************************************************************
**    ATCCONTA - NETWORK CONFIGURATION LIST FOR SYSTEM 1            *
**********************************************************************
DB1PATH,DB1CTC,DB1RSC,DB1APPLS,DBCDRMS
**********************************************************************
**    ATCCONTB - NETWORK CONFIGURATION LIST FOR SYSTEM 2            *
**********************************************************************
DB2PATH,DB2CTC,DB2RSC,DB2APPLS,DBCDRMS
```

Figure 127. Channel-to-channel (CTC) definitions

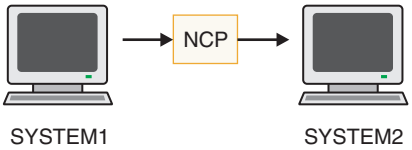
NCP-connected DB2 subsystems

The Advanced Communications Facility/Network Control Program (ACF/NCP) is a product that you can use to generate a network control program load module. The network control program load module is loaded from the host into a communications controller.

The network control program controls the lines and devices attached to it. It transfers data to and from the devices and handles any errors that occur, including retries after line errors.

A communications controller can be locally attached to a host via a channel, or it can be link-attached to another communications controller that is channel-attached.

Our sample definitions are used for the following setup:



When you are defining your NCP connections, remember the following:

- MAXBFRU must be large enough to handle the biggest PIU that is sent to the NCP. In our example, DB2 is sending 4125 bytes per PIU (4096 + a 29-byte network header). Given an IOBUF buffer size of 441 bytes, MAXBFRU must therefore be at least 10 (10 × 441 = 4410, which is greater than 4125).
- The MAXDATA option must also be large enough to handle biggest PIU (RUSIZE + 29 bytes).

If DB2 is using existing NCP definitions, you should make sure your MAXBFRU and MAXDATA options are large enough. If these values are too small, you get an SNA X'800A' sense code, indicating that the PIU was truncated during transmission.

```
************************************************************************
*        PCCU SPECIFICATION - FOR SYSTEM 1                             *
************************************************************************
PCCU1    PCCU  CUADDR=C02,         3745 BLOCK CHANNEL                  X
               AUTOSYN=YES,                                            X
               AUTODMP=NO,                                             X
               AUTOIPL=NO,                                             X
               BACKUP=YES,                                             X
               DELAY=0,                                                X
               DUMPDS=DUMPDS,      DUMP DATA SET                       X
               CDUMPDS=CDUMPDS,    CSP DUMP DATA SET                   X
               MDUMPDS=MDUMPDS,    MOSS DUMP DATA SET                  X
               INITEST=NO,         NO 3745 INITIAL TESTS AT LOAD TIME X
               MAXDATA=4302,       = BFRS*TRANSFR - 18                 X
               OWNER=HOST1,                                            X
               SUBAREA=3,          HOST SUBAREA                        X
               VFYLM=YES
************************************************************************
*        PCCU SPECIFICATION - FOR SYSTEM 2                             *
************************************************************************
PCCU2    PCCU  CUADDR=C02,         3745 BLOCK CHANNEL                  X
               AUTOSYN=YES,                                            X
               AUTODMP=NO,                                             X
               AUTOIPL=NO,                                             X
               BACKUP=YES,                                             X
               DELAY=0,                                                X
```

```
              DUMPDS=DUMPDS,      DUMP DATA SET                      X
              CDUMPDS=CDUMPDS,    CSP DUMP DATA SET                  X
              MDUMPDS=MDUMPDS,    MOSS DUMP DATA SET                 X
              INITEST=NO,         NO 3745 INITIAL TESTS AT LOAD TIME X
              MAXDATA=4302,       = BFRS*TRANSFR - 18                X
              OWNER=HOST2,                                           X
              SUBAREA=4,          HOST SUBAREA                       X
              VFYLM=YES
*********************************************************************
*        BUILD MACRO SPECIFICATIONS                                 *
*********************************************************************
ACFNCPBD BUILD BFRS=(240),         NCP BUFFER SIZE,# EP BUFFERS       X
              BRANCH=1000,                                           X
              CATRACE=(YES,10),                                      X
              CSMHDR=27F5C711C3F0405C40C8C4D9405C,                   X
              CSMHDRC=40E3C5E7E3405C5C,                              X
              CSMSG=5C5C40E5E3C1D440E2C8E4E3C4D6E6D540,              X
              CSMSGC=6040C8C1E240C2C5C7E4D5405C5C,                   X
              DIALTO=60,          WAIT 1 MIN FOR AUTOCALL ANSWER     X
              DR3270=NO,          NO DYNAMIC RECONFIG                X
              DSABLTO=3.0,        TIME TO DETECT DSR DROP            X
              ENABLTO=2.2,        TIME TO DETECT DSR AFTER ENABLE    X
              LOADLIB=NCPLOAD,    LIBRARY FOR ACF/NCP LOAD MODULE    X
              LTRACE=8,           UP TO 8 LINES CONCURRENTLY TRACED  X
              MAXSSCP=8,          NUMBER OF SSCPS IN SESSION         X
              MAXSUBA=63,         MUST BE SAME AS IN ATCSTRXX        X
              MEMSIZE=4M,         AMOUNT OF MEMORY                   X
              MODEL=3745,         3745 MODEL 410                     X
              NETID=BCR1,         3745 MODEL 410                     X
              NEWNAME=DDBLC0,     LOAD MODULE NAME                   X
              PUNAME=DDB,                                            X
              NPA=YES,            NPA WILL NOT COLLECT DATA          X
              OLT=NO,             INCLUDE ONLINE TEST FACILITY-OLTEP X
              PRTGEN=NOGEN,       DON'T PRINT ASSEMBLED STATEMENTS   X
              PWROFF=NO,                                             X
              SLODOWN=15,         SLOWDOWN AFTER 15% BUFFERS AVAIL   X
              SUBAREA=26,         NCP SUBAREA                        X
              TRACE=(YES,10),     10-16 BYTE ADDRESS TRACE ENTRIES   X
              TRANSFR=18,         =(4096+51)/BFRS--ROUNDED UP        X
              TRCPIU=2000,        SIZE OF LINE AND SIT TRACE         X
              TYPGEN=NCP,                                            X
              TYPSYS=MVS,         MVS OPERATING SYSTEM               X
              USGTIER=5,          NCP USAGE TIER - REQUIRED          X
              VERSION=V5R3,                                          X
              XBREAK=NONE
*********************************************************************
**                                                                 **
*        SYSCNTRL OPTIONS - REQUIRED BY VTAM                        *
**                                                                 **
*********************************************************************
         SYSCNTRL OPTIONS=(ENDCALL,MODE,RCNTRL,RCOND,RECMD,RIMM,     X
              SESSION,NAKLIM,LNSTAT,SSPAUSE,XMTLMT,BHSASSC,STORDSP)
*********************************************************************
HOST2    HOST  BFRPAD=0,           VTAM REQUIREMENT FOR OS           X
              INBFRS=18,          INITIAL BUFFERS FOR EACH RECEIVE  X
              MAXBFRU=10,         < BASENO IN IOBUF FOR VTAM        X
              SUBAREA=4,                                            X
              UNITSZ=441          = BUFSIZE IN IOBUF FOR VTAM

HOST1    HOST  BFRPAD=0,           VTAM REQUIREMENT FOR OS           X
              INBFRS=18,          INITIAL BUFFERS FOR EACH RECEIVE  X
              MAXBFRU=10,         < BASENO IN IOBUF FOR VTAM        X
              SUBAREA=3,                                            X
              UNITSZ=441          = BUFSIZE IN IOBUF FOR VTAM
```

```
*******************************************************************
*             PATH STATEMENTS                                     *
*******************************************************************
          PATH DESTSA=3,           SYS1                           X
               ER4=(3,1),          SYS1

          PATH DESTSA=4,           SYS2                           X
               ER4=(4,1),          SYS2
*******************************************************************
*                                                                 *
*   HOST 1 CHANNEL ADAPTER                                        *
*                                                                 *
*   LINE ADDR = 0; PHYSICAL POSITION = 5.                       *
*******************************************************************
DDBCA5   GROUP LNCTL=CA,                                           X
               ISTATUS=INACTIVE    STOP VTAM FROM ACT CHAN LINK

DDBL05   LINE  ADDRESS=0,         1ST CA PHYSICAL POSITION 1       X
               CA=TYPE6,          3745 CHANNEL ADAPTER TYPE        X
               CASDL=120,         TIME ALLOWED TO BLOCK INBOUND DATA X
               DELAY=0,           CHAN ATTN DELAY                  X
               DYNADMP=NONE,      NO EP SUBCHANNELS TO DUMP        X
               INBFRS=18,         # BUFS FOR EACH TRANSFER TO HOST X
               NPACOLL=YES,       NPA WILL COLLECT DATA ON CHANNEL X
               TIMEOUT=120        INTERVAL BEFORE CHANNEL DISCONTACT

DDBP05   PU    PUTYPE=5,          INTERMEDIATE SUBAREA FUNCTION    X
               TGN=1              MUST BE 1 FOR PUTYPE5
*******************************************************************
*                                                                 *
*   HOST 2 CHANNEL ADAPTER                                        *
*   LINE ADDR = 2; PHYSICAL POSITION = 7.                       *
*******************************************************************
DDBCA7   GROUP LNCTL=CA,                                           X
               STATUS=INACTIVE    ACT CHAN LINK

DDBL07   LINE  ADDRESS=2,         3RD CA PHYSICAL POSITION 3       X
               CA=TYPE6,          3745 CHANNEL ADAPTER TYPE        X
               CASDL=120,         TIME ALLOWED TO BLOCK INBOUND DATA X
               DELAY=0,           CHAN ATTN DELAY                  X
               DYNADMP=NONE,      NO EP SUBCHANNELS TO DUMP        X
               INBFRS=18,         #BUFS FOR EACH TRANSFER TO HOST  X
               NPACOLL=YES,       NPA WILL COLLECT DATA ON CHANNEL X
               TIMEOUT=120        INTERVAL BEFORE CHANNEL DISCONTACT

DDBP07 PU      PUTYPE=5,          INTERMEDIATE SUBAREA FUNCTION    X
               TGN=1              MUST BE 1 FOR PUTYPE5
```

Figure 128. Network control program (NCP) definitions

Using the change log inventory utility to update the BSDS

The DDF statement of the change log inventory utility can be used to insert or update information that is stored in the bootstrap data set.

Procedure

To add or update information in the bootstrap data set:

Use the options of the DDF statement of the change log inventory utility as follows.

- To update any value, you need to use only the option for that value. To insert new values, you need values for LOCATION, LUNAME, and any other values that you want to specify.

  Value Option for inserting or updating

  Location alias
  ALIAS *names*

  This value is optional. You can add or delete aliases by respecifying the ALIAS names. The new list of names replaces the existing list.

  Location name
  LOCATION=*name*

  LU name
  LUNAME=*name*

  Generic LU name
  GENERIC=*name*

  Password
  PASSWORD=*password*

  This value is optional, depending on whether you entered a password in the VTAM APPL statement

  TCP/IP connection port
  PORT=*number*

  This value is optional. If you specify a value, it must be nonzero and different from the value of RESPORT. You can delete the port by entering 0.

  TCP/IP resync port
  RESPORT=*number*

  This value is optional. If you specify a value, it must be nonzero and different from the value of PORT. You can delete the port by entering 0.

- To delete either a generic LU name, a password, or an alias, use one of these keywords:

  Value Statement for deleting

  Generic LU name
  NGENERIC

  Password
  NOPASSWD

  Alias NOALIAS

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

Connecting systems with TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is a standard communication protocol for network communications. Previous versions of DB2 supported TCP/IP requesters, although additional software and configuration was required.

About this task

The domain name (IP address) and port number (service name) uniquely identify a DB2 subsystem in the TCP/IP network. The domain name and the server port number (service name) of the database server **must** be defined in the communications database (CDB) at a requesting DB2 so that the DB2 subsystem can connect to a remote location. If you use a port number in the CDB to access a remote DB2 location, the port number must be defined to TCP/IP. The domain

name **must** be defined to the TCP/IP host so that a DB2 subsystem can accept connections from remote locations. Port numbers are never used by a DB2 subsystem that accepts connections. Optionally, you can protect the port number that DB2 uses when acting as a server within TCP/IP from being used by any other task or job in the subsystem. When DDF is started, the DB2 subsystem binds itself to its designed port.

TCP/IP terminology

DB2 uses some terms that are specific to communications.

These terms are defined as follows.

IP address
Uniquely identifies a host within the TCP/IP network. This is sometimes called an internet address. A DB2 subsystem resides on a TCP/IP host. The IP address can take two forms. The first form is a 32 bit address that displays in dotted decimal format where X'05041020' displays as 5.4.16.32. The second form is a 128 bit address that displays in colon-hexadecimal format, such as FEDC:BA98:7654:3210:FEDC:BA98:7654:3210.

Domain name
The fully qualified name that identifies an IP address. This can be used instead of the IP address. An example of a domain name is *stlmvs1.stl.ibm.com*. Some software refers to *stlmvs1* as the host name and *stl.ibm.com* as the domain name. DB2 allows the network administrator to identify a host using a domain name.

Domain name server (DNS)
Manages a distributed directory of domain names and related IP addresses. Domain names can be translated into IP addresses and you can find a domain name associated with a given IP address. DB2 uses the *gethostbyname* service to get a list of IP addresses for a given domain name.

Port
Identifies an application executing in a host. For example, a port number identifies a DB2 subsystem to TCP/IP. A port number is a two byte integer value that is displayed in decimal format. This number identifies the application within a TCP/IP instance. A port number of X'01D2' displays as *466*. There are three basic kinds of TCP/IP ports:

Well-known port
This is a port number between 1 and 1023 that is reserved in the TCP/IP architecture for a specific TCP/IP application. Some typical well-known port numbers are:
- FTP is port number 21
- Telnet is port number 23
- DRDA relational database is port number 446

Ephemeral port
Port numbers that are dynamically assigned to a client process by the client's TCP/IP instance. DB2 uses an ephemeral port when it is acting as the DRDA requester. This ephemeral port is associated with the requester for the life of the thread or connection.

Server port
Port numbers that are used when a TCP/IP program does not have a well-known port number, or another instance of the server program is already installed using the well-known port number. As a requester, DB2 defaults to using the DRDA relational database well-known port number to connect to a server location. We suggest that your DB2 subsystem be defined with the DRDA

well-known port number of 446. However, the network administrator can assign a server port to the DB2 subsystem. If two different DB2 subsystems reside on the same host, acting as two different locations (a non-data-sharing group), each DB2 subsystem must have a unique port. In this case, only one DB2 subsystem can use the DRDA well-known port number.

Service name
Another way to refer to a port number. A network administrator can assign a service name for a remote location instead of using the port number.

Enabling TCP/IP communication

To enable TCP/IP communication between DRDA partners and DB2, you must complete some tasks such as preparing the Language Environment runtime library.

About this task

You do not have to complete the following steps in exact order, but steps 1, 2, and 3 should be completed before the other steps.

1. Prepare the Language Environment runtime library
2. Enable DDF for UNIX System Services
3. Define the DB2 subsystem to TCP/IP
4. Populate the communications database
5. Start TCP/IP support
6. Tune TCP/IP

DB2 obtains more information from the TCP/IP stack, therefore requiring more configuration than many other daemons.

If you do not plan to communicate with remote sites with SNA/APPC, you do not need to define VTAM to DB2 if you update the BSDS DDF record with an IPNAME value. When you update the BSDS DDF record with an IPNAME value, DDF does not activate its SNA/APPC communications support. DB2 TCP/IP communications uses the IPNAME value and a character representation of the TCP/IP resync port (RESPORT) hexadecimal value to identify units of work. If the BSDS DDF record does not contain an IPNAME value, you must define VTAM to DB2 because DB2 TCP/IP communications uses the VTAM network ID and LUNAME to identify units of work.

DDF enhancements enable TCP/IP communication with DRDA partners with DRDA Level 3 support and earlier. To utilize the new functions, clients must have the updated versions of DB2 Connect or any DRDA requester or server that supports the latest DRDA database protocols. TCP/IP connectivity lets you connect DDF to clients on multiple platforms directly.

Attention: TCP/IP is the recommended communication protocol when accessing remote systems.

Related concepts:

TCP/IP access methods (DB2 Data Sharing Planning and Administration)

Related tasks:

"Connecting systems with VTAM" on page 883

Related reference:

DSNJU003 (change log inventory) (DB2 Utilities)

Preparing the Language Environment runtime library

Because DDF uses some functions in the Language Environment library, DDF needs access to the runtime library.

Procedure

To provide DDF with access to the runtime library:

Complete one of the following options:

- Concatenate the Language Environment library in the z/OS link list, which does not require the library to be APF-authorized. This is the standard method. If you choose this method, remove the Language Environment library concatenation from the DDF JCL procedure.
- Include the Language Environment library in a STEPLIB concatenation for the DDF JCL procedure. The Language Environment library must be APF-authorized to be added to the DDF JCL procedure. The DB2 installation automatically adds the library to the DDF STEPLIB concatenation.

Enabling DDF for UNIX System Services

DDF uses the asynchronous I/O assembler callable interface of UNIX System Services to perform TCP/IP services. Any address space that needs to use UNIX System Services must have a z/OS user ID that is defined with an OMVS segment. The address space can also have a z/OS group name.

About this task

During initialization, DDF calls UNIX System Services to raise the maximum number of open files per process to 131072. UNIX System Services considers an open TCP/IP socket an open file. This interface call requires that the process be a UNIX System Services superuser (that is, UID(0)) to raise the current limit. DDF executes as an authorized program and is protected against any unauthorized use of this privilege.

DDF obtains the current limit from the value of the MAXFILEPROC parameter. The MAXFILEPROC parameter is in the active z/OS UNIX System Services member (BPXPRMnnn) of the z/OS PARMLIB. Assuming the current value of MAXFILEPROC is 131072 or greater, the UNIX System Services call that DDF makes to set the limit to 131072 succeeds regardless of the value that you give to *nnn*.

The standard way to assign a z/OS userid and a z/OS group name to a started address space is to use the z/OS Security Server (RACF) STARTED resource class. This method enables you to dynamically assign a z/OS user ID by using commands instead of requiring an IPL to have the assignment take effect. The alternative method to assign a z/OS user ID and a z/OS group name to a started address space is to change the RACF started procedures table, ICHRIN03.

Procedure

To enable DDF to use UNIX System Services:

1. Use either of the following z/OS Security Server (RACF) commands to assign an OMVS segment to a z/OS user ID:
   - ADDUSER *ddfuid* OMVS(UID(*nnn*))...
   - ALTUSER *ddfuid* OMVS(UID(*nnn*))...

   where *ddfuid* is the z/OS user ID and *nnn* is any valid, unique identifier. If you set *nnn* to 0, the process has UNIX System Services superuser authorization.
2. Because setting the MAXFILEPROC parameter within the active BPXPRMnnn z/OS PARMLIB member to 131072 affects the entire system, any UNIX System Services process can open 131072 files or sockets concurrently. If you do not want to set the system-wide MAXFILEPROC parameter to 131072 and give the z/OS user ID superuser authority at the same time, you can explicitly authorize the z/OS user ID to raise the limit to 131072 itself by using one of the following z/OS Security Server (RACF) commands:
   - ADDUSER *ddfuid* OMVS(UID(*nnn*) FILEPROCMAX(131072))...
   - ALTUSER *ddfuid* OMVS(UID(*nnn*) FILEPROCMAX(131072))...

   where *ddfuid* is the z/OS user ID and *nnn* is any valid, unique identifier.
3. If you also want to assign a z/OS group name to the address space, assign an OMVS segment to the z/OS group name by using one of the following RACF commands:
   - ADDGROUP *ddfgnm* OMVS(GID(*nnn*))...
   - ALTGROUP *ddfgnm* OMVS(GID(*nnn*))...

   where *ddfgnm* is the z/OS group name and *nnn* is any valid, unique identifier.
4. If you actively administer the STARTED resource class, issue the following RACF commands to assign a z/OS user ID and a z/OS group name to the DDF started procedure. DDF requires only a z/OS user ID. A z/OS group name is optional.
   - RDEFINE STARTED (VA1ADIST.*) STDATA(USER(*ddfuid*) GROUP(*ddfgnm*)) ...
   - RALTER STARTED (VA1ADIST.*) STDATA(USER(*ddfuid*) GROUP(*ddfgnm*)) ...

   where VA1ADIST.* is the DDF started procedure, *ddfuid* is the z/OS user ID, and *ddfgnm* is the z/OS group name.

   Requirement: The profile name that you specify in the RACF command must be in the generic format; that is, the profile name must end with a period followed by an asterisk (.*).
5. Issue the following RACF command so that the changed profile takes effect:
   - SETROPTS RACLIST(STARTED) REFRESH

Related information:

Methods for associating started procedures with RACF identities (z/OS Security Server RACF System Programmer's Guide)

Using Started Procedures(z/OS Security Server RACF Security Administrator's Guide)

The started procedures table (ICHRIN03) (z/OS Security Server RACF System Programmer's Guide)

DB2 configuration with TCP/IP

The DB2 subsystem uses different TCP/IP ports to do different tasks.

- As a requester, DB2 uses an ephemeral port. You do not need to specify this port.
- As a server processing TCP/IP connection requests for DRDA SQL applications, DB2 uses a server port or the well-known port, 446, which is used for relational database communications.
- A server resynchronization port is used for processing 2-phase commit resynchronization requests.

This requires some planning because the port number is used to pass the network requests to the appropriate DB2 subsystem. Each location must have a unique port number.

The figure below shows some typical z/OS system configurations.

- SYSTEM1 contains only one DB2 subsystem, so the DRDA well-known port (446) can be assigned to DB2. In the example, port number 5020 is assigned for 2-phase commit resynchronization.
- SYSTEM2 contains two DB2 subsystems, so it is impossible to assign the port numbers 446 and 5020 to both DB2 subsystems because TCP/IP can only support one server at each port number. The problem is resolved by assigning the 446 and 5020 port numbers to DB2C, and port numbers 5021 and 5022 to DB2D.

Be sure to consider the impact of future system consolidations. If SYSTEM1 and SYSTEM2 are consolidated so that DB2A, DB2C, and DB2D run on a single z/OS system, you must take special precautions because DB2A and DB2C have the same TCP/IP port numbers. You can resolve this conflict by changing the port numbers of either DB2A or DB2C to eliminate the duplicate port numbers.

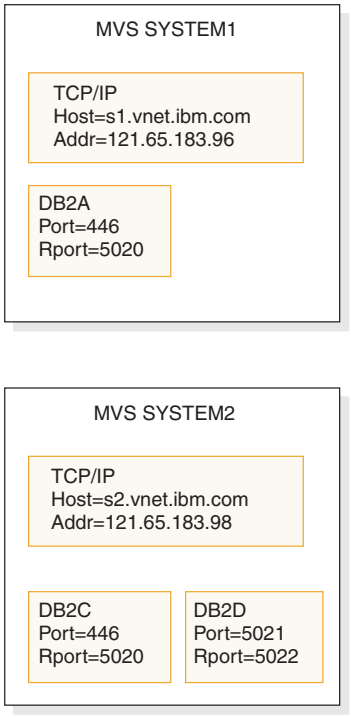


Figure 129. Typical z/OS configurations

Related concepts:

Reserving the DRDA port (DB2 Data Sharing Planning and Administration)

Customizing the TCP/IP data sets or files:

To use TCP/IP, you must customize your TCP/IP data sets or files.

Before you begin

UNIX System Services should already be installed. See the DB2 for z/OS Program Directories for required maintenance levels.

Procedure

To customize your TCP/IP data sets or files:

1. Find the high level qualifier (*hlq*) value in the `resolve/tcpip.data` file that z/OS UNIX applications use in their TCP/IP stacks. This data set defines the high level qualifier, which is added to the beginning of other data set names used by TCP/IP.
2. Find the *hlq*.TCPPARMS(PROFILE) data set. This data set contains the PORT statement used to make DRDA and resync port reservations. The following example shows a sample *hlq*.TCPPARMS(PROFILE) entry for SYSTEM2:

```
PORT  446  TCP  DB2CDIST          ; DRDA SQL port for DB2C
PORT 5020  TCP  DB2CDIST          ; Resync port for DB2C

PORT 5021  TCP  DB2DDIST          ; DRDA SQL port for DB2D
PORT 5022  TCP  DB2DDIST          ; Resync port for DB2D
```

   This example assumes that *DB2CDIST* and *DB2DDIST* are the DB2 started procedure names.

3. Define the TCP/IP host names that DB2 needs to know. The local host name **must** be defined before DDF is started. All domain names referenced in the table SYSIBM.IPNAMES **must** be defined. You define the host names by configuring the *hlq*.HOSTS.LOCAL data set, the */etc/hosts* file in the hierarchical file system (HFS), or the domain name server (DNS).

   After the *hlq*.HOSTS.LOCAL data set is configured, you have to execute the utility MAKESITE. This utility generates the *hlq*.HOSTS ADDRINFO and the *hlq*.HOSTS.SITEINFO data sets that are used to translate between domain names and IP addresses. MAKESITE generates under the userid that issued the MAKESITE. Therefore, those files have to be moved to the high level qualifier that represents TCP/IP. The host name for the local DB2 subsystem must be defined in at least one of these places.

   You can also run the MAKESITE command with the HLQ parameter. You can set the HLQ parameter to the high level qualifier of the stack that you obtained in step 1. If you use the HLQ parameter, the *hlq*.HOSTS ADDRINFO and the *hlq*.HOSTS.SITEINFO data sets are created and updated under the correct high level qualifier, so you do not have to move the data sets from under the userid high level qualifier to the TCP/IP high level qualifier.

   If domain names are present in the CDB (in field IPADDR of table SYSIBM.IPNAMES), they must be defined in the z/OS data sets, the HFS or the DNS.

   Recommendation: To support a Parallel Sysplex environment, use Dynamic Virtual IP Addresses (VIPA). To configure a DB2 subsystem to perform Dynamic VIPA routing, specify the DB2's group Dynamic VIPA on the TCP/IP PORT statement for the DRDA PORT number. This Dynamic VIPA must be the same for all DB2 members in the data sharing group. All clients must use the Dynamic VIPA to route requests to the DB2 group. To DB2 member-specific Dynamic VIPA is specified on the TCP/IP RESYNC PORT number in the TCP/IP profile data set for each DB2 member of the sysplex.

   The following example shows a sample *hlq*.TCPPARMS(PROFILE) entry for SYSTEM2:

   ```
   PORT 446  TCP  DB2DIST  BIND  db2_sysplex_VIPA       ;DRDA SQL port for DB2
   PORT 5001 TCP  DB2DIST  BIND  member_specific_VIPA    ;RESYNC port for DB2
   ```

4. Define the TCP/IP service names that DB2 needs to know. Configure the *hlq*.ETC.SERVICES data set or the */etc/services* file in the HFS. If service names are present in the CDB (in field PORT of table SYSIBM.LOCATIONS), they must be defined in the z/OS data set or the HFS.

   The following example shows a sample *hlq*.ETC.SERVICES entry:

   ```
   DRDA        446/tcp            ; DRDA databases
   ```

Related concepts:

"DB2 configuration with TCP/IP" on page 922

DVIPA network addressing (DB2 Data Sharing Planning and Administration)

Related tasks:

Configuring the base TCP/IP system

Related information:

Configuration files for the TCP/IP stack (z/OS Communications Server: IP Configuration Guide)

Modifying the change log inventory job for using TCP/IP:

If you want to use TCP/IP, the DDF statement of the change log inventory job (DSNJU003) must specify values for the parameters PORT and RESPORT.

Procedure

To modify the change log inventory job for using TCP/IP:

Specify the PORT and RESPORT parameters in the DDF statement of DSNJU003. The parameter PORT is the TCP/IP port number that DDF uses to accept incoming DRDA connection requests. The parameter RESPORT is the TCP/IP port number that DDF uses to accept incoming DRDA 2-phase commit resynchronization requests. The values for these parameters must be decimal numbers between 0 and 65534, where 0 indicates that TCP/IP support is being deactivated. The non-zero value for PORT must not be the same as the non-zero value for RESPORT.
For data sharing, all members of the DB2 data sharing group must have the same value for PORT. RESPORT must be uniquely assigned to each DB2 member so that no two DB2 members use the same TCP/IP port for 2-phase commit resynchronization. The parameters PORT and RESPORT can be changed on any DB2 member by running the utility change log inventory. After running the utility, you must stop and then restart DB2. Because PORT is the same for all members of the DB2 group, this process has to be repeated on every member of the group when PORT is changed.
If you use Dynamic VIPA to support a Parallel Sysplex environment, specify the DB2's group Dynamic VIPA on the TCP/IP PORT statement for the DRDA PORT number.
You can define an alias location for all or selected members of a data sharing group by using the ALIAS parameter.
Remember, a zero value for either PORT or RESPORT is the same as deactivating DB2's TCP/IP support.
You can also specify an IPv4 address and an IPv6 address for the subsystem in the DDF statement of the change log inventory utility (DSNJU003). Specifying an IPv4 and an IPv6 address keeps the subsystem IP addresses constant, regardless of the z/OS system where the subsystem is started. The IP address moves with the DB2 subsystem. In data sharing environments, a constant IPv4 address and a constant IPv6 address can be assigned to represent the entire group.

Related concepts:

DVIPA network addressing (DB2 Data Sharing Planning and Administration)

Update the BSDS with the DSNJU003 utility (DB2 Data Sharing Planning and Administration)

Related tasks:

Configuring subsets for member-specific access (DB2 Data Sharing Planning and Administration)

Related reference:

Syntax and options of the DSNJU003 control statement (DB2 Utilities)

Populating the communications database for use with TCP/IP

If you plan to use DB2 only as a server, you do not need to populate the CDB.

About this task

GUPI

For example, Spiffy's USIBMSTODB21 subsystem works as a server for many requesters. It is not necessary for Spiffy to register those requesters in DB2's CDB.

However, if you intend to request data, you need to enter port numbers or service names in field PORT of table SYSIBM.LOCATIONS, and IP addresses or domain names in field IPADDR of table SYSIBM.IPNAMES. The LINKNAME in table SYSIBM.LOCATIONS is used to search tables SYSIBM.IPNAMES and SYSIBM.LUNAMES.

Updates to the following tables take place whenever a new remote connection is requested, without any need to stop and restart DDF:

- SYSIBM.LOCATIONS
- SYSIBM.IPNAMES
- SYSIBM.IPLIST

If you use RACF PassTickets, the LINKNAME must match the following values at the remote site:

- LUNAME

  If the remote site is a DB2 subsystem that is defined with only an LUNAME value and no GENERIC LU name value or IPNAME value.
- GENERIC

  If the remote site is a DB2 subsystem that is defined with a GENERIC LU name value in addition to an LUNAME value, but no IPNAME value.
- IPNAME

  If the remote site is a DB2 subsystem that is defined with an IPNAME value, which triggers the DDF on the remote DB2 subsystem to activate TCP/IP communications support only.

After you populate these tables, you can write queries that access data at a remote system.

GUPI

Related concepts:

Support for executing long SQL statements in a distributed environment (DB2 Application programming and SQL)

Related tasks:

Granting privileges to remote users (Managing Security)

Related reference:

SYSIBM.USERNAMES columns (Managing Security)

TCP/IP-relevant columns of the SYSIBM.LOCATIONS table:

The LOCATIONS table contains the port number or service name that is used to connect to remote locations. The LINKNAME column of the LOCATIONS table corresponds to a row in the IPNAMES table.

GUPI

The LOCATIONS table has the following columns that relate to TCP/IP:

DBALIAS VARCHAR(128) NOT NULL
The name that is associated with the server. This name is used to access a remote database server. If DBALIAS is blank, the location name is used to access the remote database server. This column does not change database object names that are sent to the remote by using a location qualifier. Use the DBALIAS column to access data at two or more different remote locations when those remote locations have the same name.

LOCATION CHAR(16)
The unique network location name, or DRDA RDBNAM, that is assigned to a remote or local system. You must provide location names for any systems from which you request data. This column is the primary key for this table.

LINKNAME CHAR(8)
Identifies the TCP/IP attributes that are associated with this location. For each specified LINKNAME, you must have a row in SYSIBM.IPNAMES whose LINKNAME matches the value that is specified in this column. Because this table is used for outbound requests, you must provide a LINKNAME or your requests fail. Do not enter blanks in this column.

PORT CHAR(32)
If blank, the default port, 446, is used for TCP/IP communications. Otherwise, the value can be either of the following values:

- The port number of the remote database server. The number must be one to five characters and left-justified.
- A TCP/IP service name. The service name is converted to a TCP/IP port number with the *getservbyname* socket call.

Spiffy's USIBMSTODB21 location expects a LOCATIONS table that looks like the table below. The location USIBMSTODB21 uses the default DRDA PORT, 446.

Table 170. Spiffy's LOCATIONS table

LOCATION	LINKNAME	PORT
USIBMSTODB21	LUDB21	
USIBMSTODB22	LUDB22	
USIBMSTOSQL1	LUSQLDS	1234
USIBMSTOSQL2	LUSQLDS	DRDA

For example, add the second row with this statement:

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
  VALUES ('USIBMSTODB22','LUDB22');
```

Because no port number is specified, location USIBMSTODB22 uses the default DRDA port number 446.

A row for the local location: You do not need a row for the local DB2 in the IPNAMES and LOCATIONS tables. For example, Spiffy's USIBMSTODB21 subsystem does not require a row that shows its own LINKNAME and location name.

GUPI

TCP/IP-relevant columns of the SYSIBM.IPLIST table:

The IPLIST table contains a list of multiple IP addresses that are specified for a given location.

GUPI

IPLIST has the following columns:

LINKNAME CHAR(8) NOT NULL
This column is associated with the value of the LINKNAME column in SYSIBM.LOCATIONS and SYSIBM.IPNAMES. The values of the other columns in the SYSIBM.IPNAMES row apply to the server that is identified by the LINKNAME column in this row.

IPADDR VARCHAR(256) NOT NULL
This column contains an IPv4 or IPv6 address or domain name of a remote TCP/IP host of the server. If using WLM domain name server workload balancing, this column must contain the member-specific domain name. If you use dynamic VIPA workload balancing, this column must contain the member-specific dynamic VIPA address.

- An IPv4 address must be left justified and is represented as a dotted-decimal address. An example of an IPv4 address is *9.112.46.111*.
- An IPv6 address must be left justified and is represented as a colon-hexadecimal address. An example of an IPv6 address is *2001:0DB8:0000:0000:0008:0800:200C:417A*, which can also be expressed in compressed form as *2001:DB8::8:800:200C:417A*.
- A domain name is converted to an IP address by the domain name server where a resulting IPv4 or IPv6 address is determined. An example of a domain name is *stlmvs1.svl.ibm.com*. The *gethostbyname* socket call is used to resolve the domain name.

IBMREQD CHAR(1) NOT NULL WITH DEFAULT 'N'
This columns indicates whether the row came from the basic machine-readable material (MRM) tape: N=no, Y=yes

GUPI

TCP/IP-relevant columns of the SYSIBM.IPNAMES table:

The IPNAMES table defines the outbound security and host names that are used to connect to other systems through TCP/IP.

GUPI

IPNAMES has the following columns:

LINKNAME CHAR(8)
This value matches that specified in the LINKNAME column of the associated row in SYSIBM.LOCATIONS.

SECURITY_OUT CHAR(1)
Defines the security option that is used when local DB2 SQL applications connect to any remote server associated with this TCP/IP host. The default, A, means that outgoing connection requests contain an authorization ID without a password.

USERNAMES CHAR(1)
This column is used for outbound requests to control translations of authorization IDs. The values 'O' or 'B' are valid for TCP/IP connections.

IPADDR VARCHAR(254)
This column contains an IPv4 or IPv6 address or domain name of a remote TCP/IP host.

- An IPv4 address must be left justified and is represented as a dotted-decimal address. An example of an IPv4 address is *9.112.46.111*.
- An IPv6 address must be left justified and is represented as a colon-hexadecimal address. An example of an IPv6 address is *2001:0DB8:0000:0000:0008:0800:200C:417A*, which can also be expressed in compressed form as *2001:DB8::8:800:200C:417A*.
- A domain name is converted to an IP address by the domain name server where a resulting IPv4 or IPv6 address is determined. An example of a domain name is *stlmvs1.svl.ibm.com*. The *gethostbyname* socket call is used to resolve the domain name.

GUPI

TCP/IP-relevant columns of the SYSIBM.USERNAMES table:

The USERNAMES table contains information that is needed for outbound translation only.

GUPI

Remember: Inbound ID translation and *come from* checking are not done for TCP/IP requesters.

The USERNAMES table has the following columns:

TYPE CHAR(1)
Whether the row is for outbound translation. The value 'O' is valid for TCP/IP connections.

AUTHID CHAR(8)
Authorization ID to translate. If blank, it applies to all authorization IDs.

LINKNAME CHAR(8)
Identifies the TCP/IP network location associated with the row. A blank indicates it applies to all TCP/IP partners. For nonblank values, this value must match the LINKNAME value in SYSIBM.IPNAMES.

NEWAUTHID CHAR(8)
The translated value of AUTHID.

PASSWORD CHAR(8)
The password to accompany an outbound request. This column is ignored if RACF PassTickets, or already verified USERIDs are used.

GUPI

Starting TCP/IP support

Before you start DDF, ensure that z/OS UNIX System Services and TCP/IP are started. Also ensure that the local host information is set up correctly in the TCP/IP configuration files.

About this task

Important: When using dynamic VIPA, the IP addresses that have been assigned to DB2 by the TCP/IP profile must be registered with the DNS.

Procedure

To start TCP/IP support:

Start DDF.
DDF executes the following steps when it is started:

1. Establishes the TCP/IP listener by opening sockets for the DRDA SQL, Secure SQL, and resync ports to accept connections from remote locations.
2. Obtains the host name that is specified in TCPIP.DATA. The host name is used to register with WLM and, in some cases, to obtain the IP address. You can specify the IP address that DB2 should use for communication with its partners as follows:
   - Note that DB2 Version 10 is an IPv6 system and does not use gethostid to obtain the IP addresses because gethostid does not support IPv6. You can instead specify the address in the domain name server (DNS), if one is present, or in a local host file. You can specify an IPv4 address, an IPv6 address, or both, which should be mapped to the host name that is specified in TCPIP.DATA. DB2 obtains the user-specified IP addresses by resolving the host name.
   - If you want DDF to accept connections only on a specific IP address (IPv4 or IPv6), specify the IP address on the SQL port in the TCP/IP profile by using the BIND keyword. For example, use the following configuration, where 446 is the SQL port and *Vx* is either an IPv4 or IPv6 address:

     ```
     PORT
     446 TCP DB2ADIST BIND Vx
     ```

     With this method, DB2 only accepts connections on the specified IP address. Moreover, this method is not supported with secure SQL port. If your subsystem is running in data sharing mode, you must also specify an IP address on the DRDA resync port. For more details, see TCP/IP access methods (DB2 Data Sharing Planning and Administration).
   - The most recommended place to specify the IP address is in the DDF communication record of the bootstrap data set (BSDS) by using the DSNJU003 utility. For example:

     ```
     //SAMPLE    EXEC PGM=DSNJU003
     //SYSIN    DD *
       DDF IPV4=1.2.3.4,
           IPV6=1::2
     ```

     The advantage of this approach is that DB2 accepts connections not only on the specified IP address, but on any IP address that is active on the TCP/IP stack. Additionally, connections are accepted on both secure and non-secure SQL ports. DB2 also resolves the host name to an IP address through a name server or local host file to fill up any IP address that is not specified in the BSDS. For example, if the BSDS specified IPv4 but not IPv6, DB2 would try to obtain the IPv6 address by resolving the host name.

   Important: IP addresses should not be specified simultaneously in both the DB2 BSDS and TCP/IP PORT statement. Binding to a specific IP address and

binding to any IP address (INADDR_ANY) are mutually exclusive, and DDF TCP/IP services will not be available until the IP addresses are specified in only one place.

3. Obtains the domain name by resolving the IPv6 address, if one is specified, or the IPv4 address through a name server or the local host file. If you use data sharing, member domain names are obtained by resolving member IP addresses, and group domain names are obtained by resolving group IP addresses.

   Important: Whether you use data sharing or not, DB2 Version 10 uses the domain name that you specify in the name server, if one is present, or in a local host file, without further modifying or qualifying the name.However, DDF tolerates the absence of a domain name, when the IP address is specified in the DB2 BSDS.

   To determine the host IP address and domain name, issue a DISPLAY DDF command. The IPADDR field of the DSNL085I message displays the IPv4 and IPv6 addresses, if any, that are associated with the host.

```
DSNL081I STATUS=STARTD
DSNL082I LOCATION            LUNAME            GENERICLU
DSNL083I STLEC1              USIBMSY.SYEC1DB2  -NONE
DSNL084I TCPPORT=446   SECPORT=0     RESPORT=5001
DSNL085I IPADDR=::9.30.113.198
DSNL085I IPADDR=2001:DB8::8:800:200C:417A
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

   Until the local host information is available, DDF TCP/IP services are not available to local and remote applications. If a failure occurs obtaining the local host information, DDF periodically attempts to get the local host information until successful or until DDF is stopped. DDF tolerates the absence of a domain name as long as an IP address is specified in the DB2 BSDS.

4. Listens and accepts connection requests on SQL, secure SQL, and resync ports.

Related concepts:

Communicating with data sharing groups (DB2 Data Sharing Planning and Administration)

DVIPA network addressing (DB2 Data Sharing Planning and Administration)

Tuning TCP/IP

Tuning TCP/IP is optional, but it can protect DB2 from TCP/IP outages.

Procedure

To tune TCP/IP:

Specify a small value, five minutes or less, for the TCP/IP KeepAlive timer. You can specify the value for the TCP/IP KeepAlive timer either in the TCP/IP KEEPALIVE field of the Distributed data facility panel 2: DSNTIP5 or in the TCPKPALV parameter. If the network fails between the server's reply and the next client request, TCP/IP waits until the KeepAlive timer expires, then notifies the DB2 subsystem of the failure.
The server thread hangs while the timer is running. Do not use the timer default of ENABLE because it results in a value of 2 hours, which means that threads can hang for up to 2 hours. A hanging thread can cause unpredictable results, depending on what resources it has locked.
If you are connecting to a location whose LINKNAME is associated with a row in the table SYSIBM.IPNAMES and it has a domain name in the IPADDR field,

gethostbyname can return a list of IP addresses associated with the LINKNAME. When trying to connect to this location, DB2 will try each of these IP addresses in a round-robin fashion (starting with the first address) until the connection is successful, or the attempt to connect to each IP address has timed out.

TCP/IP limitations

TCP/IP does not have the built-in security features that SNA has, such as SNA partner LU verification.

Because IP addresses are not as reliable as LU names, DDF support for TCP/IP differs from support for SNA in these ways:

- There is no support for inbound name translation.
- A DB2 subsystem parameter defines the minimum security requirements for all TCP/IP clients because inbound security requirements cannot be established on individual clients. See the description of the TCP/IP ALREADY VERIFIED field on installation panel DSNTIP5.
- You cannot use the CDB for *come from* checking of TCP/IP clients.

Related reference:

"Distributed data facility panel 2: DSNTIP5" on page 448

Initializing a TCP stack for use with a VIPA

During initialization of a TCP stack for use with a virtual IP address (VIPA), the PROFILE.TCPIP configuration should contain the HOME list with the VIPA. The HOME list should be followed by the PRIMARYINTERFACE statement pointing at the appropriate VIPA that is to be used by DB2.

About this task

Recommendation: Do not change the stack's HOME list or the PRIMARYINTERFACE while DDF is started.

Procedure

To initialize a TCP stack for use with a VIPA:

Use the VARY OBEY command to change the PRIMARYINTERFACE and the stack's HOME list. This command can be used while the stack is running without recycling the stack. If a new HOME list is specified with the VARY OBEY file, the PRIMARYINTERFACE should also be specified.

Initializing a TCP stack for use with both IPv4 and IPv6 addresses

To use both IPv4 and IPv6 addresses, DB2 requires TCP/IP dual-mode stack support. Dual-mode stack support allows IPv4 address communication with IPv4 partners, and IPv6 address communication with IPv6 partners.

About this task

If you need only IPv4 address communication, an IPv4-only stack environment is sufficient because DB2 detects if the stack supports IPv6, and if not, DB2 allows communication with IPv4 addresses only.

Procedure

To enable TCP/IP for IPv4/IPv6 dual-mode stack:

1. Modify the BPXPRMxx member to define two NETWORK statements, one for AF_INET, and another for AF_INET6.

   The following example shows two NETWORK statements in the BPXPRMxx member:

   ```
   FILESYSTYPE TYPE(INET) ENTRYPOINT(EZBPFINI)
   NETWORK DOMAINNAME(AF_INET)
           DOMAINNUMBER(2)
           MAXSOCKETS(12000)
           TYPE(INET)
   NETWORK DOMAINNAME(AF_INET6)
           DOMAINNUMBER(19)
           MAXSOCKETS(13000)
           TYPE(INET)
   SUBFILESYSTYPE NAME(TCPIP) ENTRYPOINT(EZBPFINI)
           TYPE(INET)
   ```

   Separate MAXSOCKETS values are supported. The IPv6 default will be the IPv4 specified value.

2. To verify that TCP/IP is configured to support a dual-mode environment, enter a `d tcpip,,netstat,home` command on the z/OS console. Look for the INTFNAME field that has a value of LOOPBACK6 and verify that the associated ADDRESS field has a compressed colon-hexadecimal IPv6 address of ::1.

   The following example shows that TCP/IP is configured to support both IPv4 and IPv6:

   ```
   d tcpip,,netstat,home
     EZD0101I NETSTAT CS V1R5 TCPIP 034
     HOME ADDRESS LIST:
     LINKNAME: CTC1LINK
       ADDRESS: 9.30.115.135
         FLAGS: PRIMARY
     LINKNAME: LOOPBACK
       ADDRESS: 127.0.0.1
         FLAGS:
     INTFNAME: LOOPBACK6
       ADDRESS: ::1
         TYPE: LOOPBACK
         FLAGS:
     3 OF 3 RECORDS DISPLAYED
   ```

Related reference:

z/OS Communications Server: IPv6 Network and Application Design Guide

Two-phase commit for TCP/IP clients

DB2 supports two types of two-phase commit for TCP/IP clients.

- The DRDA client coordinates the two-phase commit. If a failure occurs during the commit process, DB2 might need to resynchronize with the DRDA client.
- The DRDA client gives responsibility for the resynchronization to DB2. The client sends DB2 a list of server LOCATION names, domain names, and resync IP addresses that are part of the client's unit of work. DB2 Connect uses this support by allowing DB2 to act as its Transaction Manager Database (TM_DATABASE), thereby eliminating the need to have a local database to manage the two-phase commit process. If a failure occurs during the two-phase commit process, DB2 might need to resynchronize with one or more of the server locations sent by the client.

DB2 uses the port specified on the RESYNC PORT field of installation panel DSNTIP5 for two-phase commit resynchronization. DB2 begins resynchronization using the partner's IP address and LOCATION name, and the RESYNC PORT obtained at the time of initial connection. If the partner's IP address changed, the resynchronization fails. For example, if the partner was DB2 for z/OS, the IP address can change when the automatic restart manager (ARM) restarts a data sharing member on a different CPC. The IP address can also change when an z/OS adapter fails and virtual IP addresses were not used.

If the IP address fails, DB2 uses the partner's domain name to determine the IP address for resynchronization. DRDA requesters receive the port number and domain name to be used for two-phase commit resynchronization from the server during DRDA connect processing.

No CDB definition is required to do resynchronization.

Multiple TCP/IP stacks and DB2

Configuring DB2 to run multiple TCP/IP stacks is no longer recommended.

TCP scalability and reliability are such that running two stacks may cause problems due to increased storage and CPU consumption than running one stack, with few or no expected benefits. These configuration improvements are especially helpful when there are multiple DB2 subsystems installed within a data sharing group on the single z/OS image, but these improvements can be useful in a non-data sharing environment.

Although multiple TCP/IP stacks are not recommended for DB2, there may be reasons why multiple stacks may be appropriate for your installation. A possible reason for using multiple TCP/IP stacks could be to separate internet and intranet traffic. The firewall on the stack handling external traffic can be configured only to forward very selected traffic internally, whereas the internal stack does whatever is indicated with internal traffic, including forwarding to the "external" stack. Second, a denial-of-service attack on the stack handling external traffic will not affect traffic on the stack handling internal traffic, so there is an element of resistance to attack with two stacks.

Multiple DB2 subsystems with multiple TCP/IP stacks

z/OS UNIX System Services provides a way to configure DB2 to use a single transport stack.

This allows each DB2 subsystem to be restricted to a single stack. Each would be reachable by any IP address that is owned by the selected stack, but not by other IP addresses. Adding BPXTCAFF as a new job step before the existing EXEC card for DSNYASCP for the *ssnm*DIST procedure allows DB2 to bind to a single stack when there are at least two stacks running on a single z/OS image. When you add the BPXTCAFF step as the first step, you must specify either TIME=1440 or TIME=NOLIMIT to the EXEC DSNYASCP card because the SYST parameter is disabled when a second job step is added to the *ssnm*DIST procedure.

Multiple DB2 subsystems with one TCP/IP stack

The TCP Server Bind Control allows multiple DB2 subsystems to use the same port number within the same z/OS image.

This allows multiple DB2 subsystems that bind to any IP address (also known as INADDR_ANY in sockets API terms) and the same port number on the same stack to be bound to separate IP addresses. For example, consider two DB2 subsystems

in a data sharing group where both have the same LOCATION and same DRDA PORT. A new 'BIND ipaddr' parameter is added to the PORT statement. When DB2 (identified by job name *ssnm*DIST) issues a bind to the port number in the PORT statement and to INADDR_ANY, the bind is restricted to the IP address specified on the PORT statement for that DB2 subsystem. This allows two DB2 subsystems to share the same stack by limiting them to different single IP addresses (virtual IP addresses are recommended). This allows them to be reached by any physical connectivity that can get to the stack.

Related reference:

PORT statement (z/OS Communications Server: IP Configuration Reference)

Specifying security requirements

When you set up systems to communicate through TCP/IP, you must specify security requirements.

Related tasks:

Establishing RACF protection for DB2 (Managing Security)

Configuring the DB2 command line processor

You can use the DB2 command line processor to issue SQL statements, bind DBRMs that are stored in HFS files, and call stored procedures. To use the command line processor, ensure that your system meets the command line processor prerequisites. Then, set up your system to use the command line processor.

Related concepts:

Command line processor (DB2 Commands)

Command line processor prerequisites

Before you can set up your system to use the DB2 command line processor, the system must meet several prerequisites.

Ensure that your system has the following:

- Java 1.4.2 service release 2.

  If you use or reference the DB2 Version 10 DECIMAL FLOAT SQL type, Java 1.5 is required.
- IBM Data Server Driver for JDBC and SQLJ.

  For information on installing this feature, see Installing the IBM Data Server Driver for JDBC and SQLJ (Application Programming for Java).
- DB2 metadata stored procedures installed and authorized.

  To install and authorize the DB2 metadata stored procedures, run the installation job DSNTIJRT. To check if the DB2 metadata stored procedures were successfully installed and activated, run the validation job DSNTIJRV.

Setting up your system to use the DB2 command line processor

The command line processor on DB2 for z/OS is a Java application that runs under UNIX System Services.

Before you begin

Before you begin setting up the DB2 command line processor, ensure that your system meets the command line processor prerequisites.

Procedure

To set up your system to use the DB2 command line processor:

1. Modify the CLASSPATH variable in your .profile file to include the clp.jar file. If you install the command line processor into the /usr/lpp/db2a10_base directory, you can issue the following command to modify your CLASSPATH variable:

   ```
   export CLASSPATH=$CLASSPATH:/usr/lpp/db2a10_base/lib/clp.jar
   ```

2. Optional: Define the CLPPROPERTIESFILE environment variable in your .profile file to include the full path name where the properties file is located. You can create your own properties file with any name. A sample properties file is in the directory where the command line processor is installed. If you install the command line processor into the /usr/lpp/db2a10_base directory, you can issue the following command to define the CLPPROPERTIESFILE environment variable in your .profile file:

   ```
   export CLPPROPERTIESFILE=/usr/lpp/db2a10_base/samples/clp.properties
   ```

   If the CLPPROPERTIESFILE environment variable is undefined, the command line processor uses default values for all options when it is invoked.

3. Define the following alias in your .profile file:

   ```
   alias db2="java com.ibm.db2.clp.db2"
   ```

4. Invoke the command line processor:

   ```
   db2
   ```

Results

Additional instructions for setting up the DB2 command line processor are available in the DB2 command line processor readme file. By default, the DB2 command line processor readme file (README_CLP) is installed into the /usr/lpp/db2a10_base directory.

IBM Text Search for DB2 for z/OS

IBM Text Search for DB2 for z/OS provides full-text search capability for text columns that are stored in DB2. You can enable this functionality after your DB2 for z/OS subsystem or data sharing member is in new-function mode.

Related tasks:

Enabling text search support for DB2 Version 10 (IBM OmniFind Text Search Server Installation, Administration, and Reference)

Creating tables for use by an accelerator product

If you use an accelerator product, you need to create a database, a table space, tables, and indexes for the accelerator product to use. An *accelerator product* is a separate product that is designed to reduce the response times of database queries.

About this task

After the tables are created, they must be populated to define a server and the associated IP address.

Procedure

To create tables for use by an accelerator product:

1. Run job DSNTIJAS to create the objects that are required for an accelerator product.

   The tables are populated by the accelerator product.
2. Restrict access to the SYSACCEL.SYSACCELERATORS table that is created by job DSNTIJAS. This table contains the authentication token that is used to authenticate DB2 with the accelerators in column ACCELERATORAUTHTOKEN.

Related reference:

Information about one example of an IBM version 2 accelerator product

Chapter 9. Working with international data

DB2 handles different character sets in distributed data through character conversions.

Character conversion concepts

Character data that is transmitted from one DBMS to another might need to be converted to a different coded character set.

In different database management systems (DBMSs), character data can be represented by different encoding schemes. Within an encoding scheme, there are multiple coded character set identifiers (CCSIDs). EBCDIC, ASCII, and Unicode are ways of encoding character data.

The Unicode character encoding standard is a character encoding scheme that includes characters from almost all living languages of the world. DB2 supports two implementations of the Unicode encoding scheme: UTF-8 (a mixed-byte form) and UTF-16 (a double-byte form).

All character data has a CCSID. Character conversion is described in terms of CCSIDs of the source and of the target. When you install DB2, you must specify a CCSID for DB2 character data in either of the following situations:

- You specify AUTO or COMMAND for the DDF STARTUP OPTION field on panel DSNTIPR.
- Your system will have any ASCII data, Unicode data, EBCDIC mixed character data, or EBCDIC graphic data. In this case, you must specify YES in the MIXED DATA field of panel DSNTIPF, and the CCSID that you specify is the mixed data CCSID for the encoding scheme.

The CCSID that you specify depends on the national language that you use.

DB2 performs most character conversion automatically, based on system CCSIDs, when data is sent to DB2 or when data is stored in DB2. If character conversion must occur, DB2 uses the following methods:

1. DB2 searches the catalog table SYSIBM.SYSSTRINGS.
2. DB2 uses z/OS Unicode Conversion Services.

If DB2 or z/OS Unicode Conversion Services does not provide a conversion for a certain combination of source and target CCSIDs, you receive an error message. If the conversion is incorrect, you might get an error message or unexpected output. To correct the problem, you need to understand the rules for assigning source and target CCSIDs in SQL operations.

© Copyright IBM Corp. 1982, 2012

Related concepts:

"System-coded character set identifiers"

"Unicode support in DB2"

Coded character sets and CCSIDS (DB2 SQL)

Related reference:

Character conversion (z/OS Support for Unicode: Using Unicode Services)

System-coded character set identifiers

To support character conversion, the IBM Distributed Relational Database Architecture (DRDA) uses CCSIDs to label the various character representation schemes.

The CCSID is a two-byte binary number that uniquely identifies one or more pairs of character sets and code pages. The coded character set defines how bit configurations are mapped for character data. .

The CCSID of character strings at your site is determined by the CCSID that you specify on installation panel DSNTIPF. **If this CCSID is not correct, character conversion produces incorrect results.** The correct CCSID is the number that identifies the coded character set that is supported by your site's I/O devices, local applications such as IMS and QMF, and remote applications such as CICS Transaction Server.

Related information:

Character Data Representation Architecture Reference

Unicode support in DB2

Unicode is a universal encoding scheme for written characters and text that enables the exchange of data internationally. Unicode provides a character set standard that can be used all over the world.

Unicode uses an encoding scheme that provides code points for approximately 100 000 characters. An extension called UTF-16 allows for encoding as many as a million more characters. Unicode provides the ability to encode all characters used for the written languages of the world. Unicode treats alphabetic characters, ideographic characters, and symbols equivalently because it specifies a numeric value and a name for each of its characters. Unicode includes punctuation marks, mathematical symbols, technical symbols, geometric shapes, and dingbats.

DB2 provides the following Unicode encoding forms:

- UTF-8: Unicode Transformation Format, 8-bit encoding form that is designed for ease of use with existing ASCII-based systems.
- UTF-16: Unicode Transformation Format, 16-bit encoding form that is designed to provide code values for over a million characters and a superset of UCS-2. UCS-2 is Universal Character Set and is coded in 2 octets, which means that characters are represented in 16 bits per character.

Unicode CCSIDs: The Unicode CCSID field of panel DSNTIPF is pre-filled with 1208. DB2 chooses the CCSIDs for double-byte and single-byte values (1200 for DBCS and 367 for SBCS). CCSID 1200 corresponds to UTF-16 and CCSID 367 is for 7-bit ASCII.

Customizing support for Unicode

You must customize z/OS Unicode support to use Unicode data in DB2.

Before you begin

Ensure that the conversion environment is active. The steps for this process can be found in z/OS Support for Unicode: Using Unicode Services. DB2 can use the conversion services of z/OS support for Unicode only when the conversion environment is active. The infrastructure provides tools to create a conversion image. When the image is loaded into a common data space, the conversion environment is activated, and the conversion services are ready to be used by DB2.

Procedure

To customize support for Unicode:

1. Customize the job card. The jobs in *hlq*.SCUNJCL, as shipped by IBM, have placeholders for values on the JOB statement, such as in the following example:

   ```
   //$JOBPREF$$JOBNAME$ JOB ($ACCOUNT$), '$USER$',
   //     NOTIFY=$NOTIFY$,MSGCLASS=$MC$,MSGLEVEL=$ML$,
   //     TIME=$TI$,CLASS=$CL$,REGION=$REGION0M$
   ```

   Use the REXX EXEC CUNRUCST in *hlq*.SCUNREXX to customize these values. When you run the REXX EXEC CUNRUALL, the values that you specify are supplied on all of the JCL images. You can choose to customize your system by displaying Japanese messages or displaying English messages with modified date and time formats. You can set up the z/OS Message Service to specify how you want messages to be displayed.

2. Set up the conversion image. The following example is the sample JCL member in *hlq*.SCUNJCL (CUNJIUTL):

   ```
   //CUNMIUTL EXEC PGM=CUNMIUTL
   //SYSPRINT DD   SYSOUT=*
   //TABIN    DD   DISP=SHR.DSN=hlq.SCUNTBL
   //SYSIMG   DD   DSN=hlq.IMAGES(CUNIMG00),DISP=SHR
   //SYSIN    DD   *
      /*******************************************
       * INPUT STATEMENTS FOR THE IMAGE GENERATOR *
       *******************************************/
         CASE NORMAL;          /* ENABLE TOUPPER AND TOLOWER */
         CONVERSION 1047,850;  /* EBCDIC -> ASCII */
         CONVERSION 850,1047;  /* ASCII -> EBCDIC */
   /*
   ```

   In the preceding example, the two CONVERSION statements provide conversion between the EBCDIC code page 1047 and the ASCII code page 850 in both directions.The DD names that are passed to CUNMIUTL are described as follows:

   SYSPRINT
   A listing that shows the processed setups and error messages, if applicable.

   TABIN
   Conversion tables for character conversion and case conversion. They are supplied by IBM; in this example, they are in data set *hlq*.SCUNTBL. The image transforms the conversion tables into an internal format and stores them in the conversion image.

   SYSIMG
   Output is a single image of the entire conversion environment. The

conversion image is built according to the specification in the SYSIN DD name. The conversion image resides in either a sequential data set or a member of a partitioned data set with a fixed-block 80-byte format. In this example, the image resides in the partitioned data set member *hlq*.IMAGES(CUNIMG00).

SYSIN

Two types of statements are recognized in this DD statement: case conversion, which is identified by the CASE control statement, and character conversion, which is identified by the CONVERSION control statement.

CASE control statement:

Case conversion is defined as converting Unicode characters (for example, UTF-8) to their uppercase equivalent or their lowercase equivalent. In the preceding example, CASE NORMAL is specified, which means basic case conversion is provided. This basic case conversion is based on the UnicodeData.txt file that is provided by the Unicode consortium. It does not include special casing as described in the SpecialCasing.txt file that is provided by the Unicode consortium. Special casing typically includes characters that have significant differences in the case-based appearance. For example, the German "hard S", which appears as a flat B, appears as "SS" in uppercase German text. Because DB2 does not use the case conversion service, you do not need to specify a conversion.

CONVERSION control statement:

Character conversion is also referred to as conversion between specified CCSIDs. An application such as DB2 invokes the CUNLCNV function to convert characters between the specified code pages. You must identify the conversions that are possible on the CONVERSION control statement.

Important: Specify CONVERSION statements for DB2 as follows:

```
CONVERSION xxx,yyy,ER;
CONVERSION yyy,xxx,ER;
```

Many code page conversions are possible. However, when identifying the conversion that DB2 is to use, you need be concerned only with conversion for the national languages that you use and with conversion from these code pages to and from all Unicode CCSIDs.

Example: If you use an EBCDIC CCSID of 37 and an ASCII CCSID of 819, you need to use the following conversions:

```
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;

CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

Multiple conversion tables might be available for converting one CCSID to another. A technique search order can be used to specify which table should be used. The technique search order consists of up to eight technique characters. If you specify more than one technique character, the image generator tries to find a matching table for the leftmost technique character in the sequence of the technique-search-order. If one is not found, the search continues with the second one, and so on. Especially for mixed conversion, use more than one technique character because one of the subconversions might exist only in round-trip mode, and one might exist only in an enforced subset. In this case, a technique search order of 'RE' or 'ER' would be required. Technique search order is optional. If you do not specify a technique search order, RECLM is used.

Language products such as Enterprise Cobol might use the RECLM technique search order, while DB2 uses the ER technique search order. Therefore, you might also need to add the RECLM conversions, such as these:

```
CONVERSION 1047,850,RECLM;  /* EBCDIC -> ASCII */
CONVERSION 850,1047,RECLM;  /* ASCII -> EBCDIC */
```

The important technique characters for DB2 are E (enforced subset) and R (round-trip). Enforced subset conversions map only those characters from one CCSID to another that have a corresponding character in the second CCSID. All other characters are replaced by a substitution character. Round-trip conversions between two CCSIDs assure that all characters making the 'round trip' arrive as they were originally, even if the receiving CCSID does not support a given character. Round-trip conversions ensure that code points that are converted from CCSID A to CCSID B, and back to CCSID A are preserved, even if CCSID B is not capable of representing these code points.

After performing these steps, you should now have an updated CUNJIUTL JCL member.

3. Submit the batch job in the CUNJIUTL member. At completion, the batch job writes its output to the SYSPRINT DD (that is, SYSOUT in this example). Expect a return code of zero from the CUNJIUTL program. If you receive anything other than return code zero, refer to Return code meanings. This information helps you correct environmental, syntactical, and semantic errors that might occur.
4. After generating the conversion image, copy it to SYS1.PARMLIB or any other data set in the logical PARMLIB concatenation. In this example, you copy *hlq*.IMAGES(CUNIMG00) to SYS1.PARMLIB(CUNIMG00).
5. Calculate the storage that is needed for a conversion image. When the conversion image is created on disk, you need to determine the amount of virtual storage that the image is to occupy. You specify this number as the number of pages on the REALSTORAGE parameter in the CUNUNI*xx* PARMLIB member that you create in the next step. The REALSTORAGE parameter protects the system from a shortage of main storage caused by loading a conversion image that exceeds the amount of available storage. The minimum value for the REALSTORAGE parameter depends on how the image is activated.

- If the image is activated during IPL, the needed storage is the size of the image plus one page.
- If the image is activated using the SET UNI command (PARMLIB member with keyword IMAGE), the needed storage is the size of the currently active image plus the size of the new conversion image.

If you set up a conversion environment before or if you are not activating a conversion environment during IPL, you must determine the amount of storage that the currently active image occupies. To do this, issue the following command:

```
D UNI,STORAGE
```

The system displays the number of active pages.

To determine the storage that the new conversion image occupies, find the CUN1017I message in the SYSPRINT log that was created in the previous step. This message indicates the number of pages that are required for the new conversion image. For example:

```
CUN1017I GENERATED IMAGE SIZE 291 PAGES........
```

As an alternative, specify a REALSTORAGE value of zero, which indicates that unlimited storage is available. In this case, a value of 524 287 pages is used.

6. Create the PARMLIB member CUNUNI*xx* (PARMLIB member for activating a conversion environment). Normally the member is created in SYS1.PARMLIB, but in this case, you create it in another data set in the logical PARMLIB concatenation. This example uses SYS1.PARMLIB.

   The *xx* can be any two alphanumeric characters, or the special characters @, #, or $. This example uses 00. Here is the sample PARMLIB member in SYS1.PARMLIB(CUNUNI00):

   ```
   REALSTORAGE 292;
   IMAGE CUNIMG00;
   ```

   Because CUNMIUTL requires 291 pages, and an additional page is required during IPL, the REALSTORAGE statement indicates that 292 pages of real storage are required.

   The IMAGE parameter indicates that the system searches in SYS1.PARMLIB (or a data set in the logical concatenation) member CUNIMG00 for the conversion image.

   You can create a PARMLIB member to delete a current conversion environment.

7. Take one of the following actions:
   - Edit IEASYSxx.

     This parameter specifies one or more CUNUNI*xx* PARMLIB members that contain the keywords that configure the conversion environment. Each suffix *xx* identifies one CUNUNI*xx* member in the PARMLIB concatenation. If several PARMLIB members are specified, they are concatenated in the specified sequence. The concatenated contents is handled internally as a single member. This means that the lines are numbered consecutively, and error messages about syntax errors refer to the concatenated text. Restrictions for keywords apply for the entire concatenated text.
   - Check parameter MAXCAD in IEASYSxx. It limits the amount of common data spaces in a system. If MAXCAD is specified, consider that z/OS support for Unicode creates up to two common data spaces.
8. Initialize the conversion environment with an IPL.
9. After the system is initialized, you can use the DISPLAY UNI system command to show the current z/OS Unicode status or use the SET UNI system command to change the conversion environment.

Related concepts:

Calling the character conversion services (z/OS Support for Unicode: Using Unicode Services)

Related reference:

Creating a Unicode Services environment (z/OS Support for Unicode: Using Unicode Services)

EBCDIC and ASCII support

DB2 supports both EBCDIC and ASCII encoding of data.

If you specify MIXED DATA = NO on panel DSNTIPF, you can use any compatible SBCS CCSID in the EBCDIC CCSID and ASCII CCSID fields. The first table below lists a selection of common SBCS CCSIDs that might be used as source or target CCSIDs for EBCDIC or ASCII data. DB2 does not support the storing of data into all of these CCSIDs. That is, not all of the numbers listed in the first table below are supported as target CCSIDs in conversion. When you choose CCSIDs, you should choose an EBCDIC CCSID and an ASCII CCSID that are listed in the same row in the first table below.

To determine which combinations DB2 supports with SYSSTRINGS, issue the following SQL statement:

GUPI

```
SELECT * FROM SYSIBM.SYSSTRINGS;
```

GUPI

If you specify MIXED DATA=NO on installation panel DSNTIPF, specify an SBCS CCSID from the second table below in the EBCDIC CCSID field on DSNTIPF. You must specify a SBCS CCSID from the third table below in the ASCII CCSID field on DSNTIPF. Mixed character data and graphic data cannot be defined on a system when you specify MIXED DATA=NO.

If you specify MIXED DATA=YES on installation panel DSNTIPF, specify a mixed data CCSID from the second table below in the EBCDIC CCSID field on DSNTIPF. You must specify a mixed CCSID from the third table below in the ASCII CCSID field. The second and third tables below show the associated CCSIDs that DB2 assigns for SBCS and DBCS data when you specify a specific MCCSID.

Then specify a mixed CCSID from the third table in the ASCII CCSID field on DSNTIPF. By specifying a CCSID for mixed data (an MCCSID), you also receive system CCSIDs for SBCS and DBCS (graphic) data.

In the second and third tables, four CCSIDs are listed for Japanese to allow for all possible combinations of two single-byte code pages and two double-byte character sets. The difference between the single-byte code pages is in the code points for lowercase Latin letters and Katakana characters. In the code page for Japanese (Extended English, SCCSID 1027), lowercase letters have the same code points as other EBCDIC code pages.

In these tables, the terms are used as follows:

SCCSID
single-byte coded character set identifier

MCCSID
mixed coded character set identifier

GCCSID
graphic coded character set identifier

Table 171. Single-byte coded character set identifiers (CCSIDs)

Country or national language	EBCDIC	ASCII PC	ASCII AIX®	ASCII Windows
Australia (U.S. English)	37/1140*	437	819	1252/5348*
Austria (German)	273/1141*	850/858*	819	1252/5348*
Belarus (Cyrillic)	1025			1251/5347*
Belgium	500/1148*	850/858*	819	1252/5348*
Bosnia and Herzegovina (Cyrillic)	1025			1251/5347*
Bosnia and Herzegovina (Latin)	870	852	912	1250/5346*
Brazil (U.S. English)	37/1140*	850/858*	819	1252/5348*
Bulgaria (Cyrillic Multilingual)	1025			1251/5347*
Canada (U.S. English)	37/1140*	850/858*	819	1252/5348*
Croatia	870	852	912	1250/5346*
Czech Republic	870	852	912	1250/5346*
Denmark	277/1142*	850/858*	819	1252/5348*
Finland (Swedish)	278/1143*	850/858*	819	1252/5348*
France	297/1147*	850/858*	819	1252/5348*
Germany	273/1141*	850/858*	819	1252/5348*
Greece	875 or 423	869	813	1253/5349*
Hungary	870	852	912	1250/5346*
Iceland	871/1149*	850/858*	819	1252/5348*
International Latin-1	500/1148*			
Israel	424	862	916	1255/5351*
Italy	280/1144*	850/858*	819	1252/5348*
Latin America (Spanish)	284/1145*	850/858*	819	1252/5348*
FYR Macedonia	1025			1251/5347*
Netherlands (U.S. English)	37/1140*	850/858*	819	1252/5348*
New Zealand (U.S. English)	37/1140*	437	819	1252/5348*
Norway	277/1142*	850/858*	819	1252/5348*

Table 171. Single-byte coded character set identifiers (CCSIDs) (continued)

Country or national language	EBCDIC	ASCII PC	ASCII AIX®	ASCII Windows
Poland	870	852	912	1250/5346*
Portugal (U.S. English)	37/1140*	850/858*	819	1252/5348*
Russia (Cyrillic)	1025			1251/5347*
Serbia and Montenegro (Cyrillic)	1025			1251/5347*
Serbia and Montenegro (Latin)	870	852	912	1250/5346*
Spain	284/1145*	850/858*	819	1252/5348*
Sweden	278/1143*	850/858*	819	1252/5348*
Switzerland	500 /1148*	850/858*	819	1252/5348*
Thailand	838			
Turkey (Latin 5)	1026	857	920	1254/5350*
United Kingdom	285/1146*	850/858*	819	1252/5348*
U.S.A. (U.S. English)	37/1140*	437	819	1252/5348*

Note: * This number represents the equivalent CCSIDs using the euro symbol.

Table 172. EBCDIC double-byte coded character set identifiers (CCSIDs)

National language	MCCSID	SCCSID	GCCSID	User-defined characters
Japanese (Extended Katakana)	930	290	300	4370
Japanese (Katakana-Kanji)	1390	8482	16684	6205
Japanese (Extended Katakana)	5026	290	4396	1880
Japanese (Extended English)	939	1027	300	4370
Japanese (Latin-Kanji)	1399	5123	16684	6205
Japanese (Extended English)	5035	1027	4396	1880
Korean	933	833	834	1880
Korean	1364	13121	4930	1880
Simplified Chinese	935	836	837	1880
Simplified Chinese	1388	13124	4933	1880
Traditional Chinese	937	28709	835	6204

Table 173. ASCII double-byte coded character set identifiers (CCSIDs)

National language	MCCSID	SCCSID	GCCSID	User-defined characters
Japanese	932	897	301	1880
Japanese (Extended)	942	1041*	301	1880
Japanese (Open environment)	943	1041*	941	1880
Japanese (HP)	5039	1041*	1351	940
Korean	949	1088	951	1880
Korean (EUC)	970	367	971	1880
Korean	1363	1126	1362	1880
Simplified Chinese	1381	1115	1380	1880
Simplified Chinese (EUC)	1383	367	1382	
Simplified Chinese	1386	5210	1385	1880
Traditional Chinese	938	904	927	6204
Traditional Chinese	948	1043	927	6204
Traditional Chinese (IBM Big-5)	950	1114	947	

Note: * The SCCSID 1041 is a superset of SCCSID 897.

Implications of character conversions

Some character conversions have implications that you should be aware of. For example, when a character string is converted to another encoding scheme, the converted string might have a different length from the source string. In this case, the conversion is an *expanding conversion* or a *contracting conversion*.

Expanding conversions: An *expanding conversion* occurs when the length of the converted string is greater than that of the source string. For example, an expanding conversion occurs when an ASCII mixed data string that contains DBCS characters is converted to EBCDIC mixed data. Because of the addition of shift characters, an error occurs when an expanding conversion is performed on a fixed-length input host variable that requires conversion from ASCII mixed data to EBCDIC mixed data. The solution is to use a varying-length string variable with a maximum length that is sufficient to contain the expansion. Expanding conversions also can occur when string data is converted to or from Unicode.

Contracting conversions: A *contracting conversion* occurs when the length of the converted string is smaller than that of the source string. For example, a contracting conversion occurs when an EBCDIC mixed data string that contains DBCS characters is converted to ASCII mixed data due to the removal of shift characters. Contracting conversions also can occur when string data is converted to or from Unicode.

Euro symbol support

DB2 enables users to migrate to CCSIDs that support the euro symbol. This support is limited to conversion from specific CCSIDs that do not define the euro symbol to specific CCSIDs that define the euro symbol.

Attention: If you want to migrate to CCSIDs that support the euro symbol, contact IBM Software Support. Altering CCSIDs without specific guidance from IBM Software Support is strongly discouraged.

The tables below list the ASCII and EBCDIC CCSIDs that can be converted to CCSIDs that support the euro symbol. In most cases, the euro symbol replaces an existing code point, such as the International Currency Symbol (ICS). Unicode UTF-8 (1208) and UTF-16 (1200) support the euro symbol. Unicode SBCS data (367) does not support the euro symbol.

The CCSIDs that you can modify are listed according to the encoding scheme. The first table below lists the EBCDIC CCSIDs, and the second table below lists the ASCII CCSIDs.

Table 174. EBCDIC CCSID values that convert to euro CCSIDs

CCSIDs without euro symbol	CCSIDs with euro symbol
37	1140
273	1141
277	1142
278	1143
280	1144
284	1145
285	1146
297	1147
500	1148
871	1149

Table 175. ASCII CCSID values that convert to euro CCSIDs

CCSIDs without euro symbol	CCSIDs with euro symbol
850	858
1250	5346
1251	5347
1252	5348
1253	5349
1254	5350
1255	5351
1256	5352
1257	5353
874	4970

You cannot convert other CCSIDs to CCSIDs that support the euro symbol. All databases and all table spaces within an encoding scheme should be altered at the same time.

If you migrate to a euro-enabled CCSID on DB2 and you have DB2 Connect clients, you should set the DB2CONNECT_ENABLE_EURO_CODEPAGE environment variable on the client side.

Related concepts:

"The update process" on page 501

How an entry in SYSIBM.SYSSTRINGS works with character conversion

The catalog table SYSIBM.SYSSTRINGS contains information about character conversion. Each row in the table describes a conversion from one coded character set to another.

The following columns in SYSSTRINGS are relevant to character conversion.

INCCSID
The source CCSID of a character conversion.

OUTCCSID
The target CCSID of a character conversion.

TRANSTYPE
The type of conversion:

SS SBCS data to SBCS data
SM SBCS data to EBCDIC MIXED data
MS EBCDIC MIXED data to SBCS (EBCDIC and ASCII) data
PS ASCII MIXED data to SBCS (EBCDIC and ASCII) data
GG GRAPHIC data to GRAPHIC data
PM ASCII MIXED data to EBCDIC MIXED data
MM EBCDIC MIXED data to EBCDIC MIXED data
MP EBCDIC MIXED to ASCII MIXED data
PP ASCII MIXED to ASCII MIXED data
SP SBCS (ASCII and EBCDIC) to ASCII MIXED data

ERRORBYTE
Specifies the byte that is used in the conversion table (TRANSTAB) as an error indicator. For example, if ERRORBYTE is X'3E', that byte is used in the conversion table to indicate that no conversion is defined for code points that map to X'3E'. Null indicates the absence of an error indicator.

SUBBYTE
Specifies the byte that is used in the conversion table (TRANSTAB) as a substitution character. For example, if SUBBYTE is X'3F', that byte is used in the conversion table as a substitute for code points that map to X'3F'. A warning occurs when a code point maps to the value of SUBBYTE. Null indicates the absence of a substitution character.

TRANSPROC
The name of a module or a blank string. If IBMREQD is N, a non-blank value of TRANSPROC is the name of a user-provided conversion procedure. If IBMREQD is Y, a non-blank value of TRANSPROC is the name of a DB2 module that contains DBCS conversion tables.

IBMREQD
Y indicates that the row is provided by IBM. N indicates that the row has been inserted by the user.

TRANSTAB
A 256-byte conversion table or an empty string.

Each row of SYSSTRINGS contains information about the conversion of character strings from the coded character set that is identified by INCCSID to the coded

character set that is identified by OUTCCSID. The conversion function is automatically invoked when a conversion from the coded character set that is identified by the INCCSID column to the coded character set that is identified by the OUTCCSID column is required.

For example, the row of SYSSTRINGS in which the value of INCCSID is 500 and the value of OUTCCSID is 37 describes the conversion from CCSID 500 to CCSID 37. The row in which the value of INCCSID is 37 and the value of OUTCCSID is 500 describes the conversion from CCSID 37 to CCSID 500.

DB2 enforces a distinction between IBM-supplied rows and user-provided rows with the following constraints:

- Rows with IBMREQD=Y cannot be updated or deleted.
- Rows with IBMREQD=N can be inserted, updated, and deleted.
- The same pair of CCSIDs can be in two rows, if one is in an IBM-supplied row and the other is in a user-provided row. In this case, the user-provided row is used for the character conversion.

The following table lists the types of rows that are possible in SYSSTRINGS.

Table 176. Types of rows in SYSSTRINGS

The value of TRANSPROC is	The value of TRANSTAB is	The value of IBMREQD is	The result is
blank	an empty string	–	No conversion is performed.
not blank	–	NO	Conversion is performed by the conversion procedure module name identified in the TRANSPROC column
blank	not empty	–	Conversion is performed by the DB2 module using the conversion table identified in TRANSTAB

Be aware of the following rules for SYSSTRINGS entries:

- An INSERT, UPDATE, DELETE, or LOAD is allowed only if IBMREQD=N.
- The values in the INCCSID and OUTCCSID columns must be in the range of 1 to 65533.
- For any given row, the INCCSID and OUTCCSID columns cannot contain the same value.
- The value in the TRANSTYPE column must be SS, SM, MS, PS, MM, PM, GG, MP, PP, or SP.
- For any given row, the ERRORBYTE and SUBBYTE columns cannot contain the same nonnull value.
- The TRANSPROC column must either be blank or contain a string that conforms to the rules for z/OS program names.
- The length that is specified in the TRANSTAB column must be either 0 or 256.

Related reference:

z/OS XL C Support for the double-byte character set (XL C/C++ Programming Guide)

Conversion-related changes that require remote packages to be rebound

Certain conversion-related changes at the local DBMS or at a remote DBMS might force the rebinding of a package.

These include the following changes:

- The system CCSID at the remote DBMS was changed. In this case, always rebind the package.
- The system CCSID at the local DBMS was changed. This could happen, for example, if the wrong system CCSID was specified during installation. If so, string constants in static SQL statements might have been converted incorrectly during the binding of the package. Rebinding corrects the conversion. Other problems might also arise as a result of the change. In most cases, you should rebind the package.
- The subtype of a character column is changed at the remote DBMS. The pertinent changes are from BIT to either SBCS or MIXED, and from SBCS or MIXED to BIT. The change was probably made by modifying the FOREIGNKEY column of the SYSIBM.SYSCOLUMNS table in the remote system catalog. Alternatively, the change might have occurred if the table was dropped and re-created with a different subtype (BIT to either SBCS or MIXED, or either SBC or MIXED to BIT) after the application was bound. A statement that refers to a column with a modified subtype might fail with an SQLCODE of -333. If this occurs, rebind the package containing the statement.

Locales for uppercase and lowercase conversion of EBCDIC data

Rules for uppercase and lowercase usage vary according to language and country. A *locale* defines the subset of a user's environment that depends on language and cultural conventions.

DB2 uses the information that is associated with a locale to execute UPPER, LOWER, and TRANSLATE functions in a culturally correct manner. A locale consists of two components: the first component represents a specific language and country, and the second component is a CCSID.

Example: In the locale, Fr_CA.IBM-1047, Fr_CA represents the language and country (French Canadian), and IBM-1047 is the associated CCSID.

The symbol for euro currency is supported through the modifier @EURO.

Example: To display results in euro dollars instead of French Francs, specify Fr_FR@EURO.

DB2 uses the SCEELKED and SCEERUN Language Environment (LE) libraries. Both libraries are PDS libraries and are for non-XPLINK linkage only. If you need to customize the locales using Language Environment libraries, see Internationalization: Locales and Character Sets (XL C/C++ Programming Guide).

The following table shows a partial list of locales that are supplied with z/OS C/C++. For a more complete list of locales, see Compiled locales (LE locales) (XL C/C++ Programming Guide).

Table 177. Examples of locales supplied with z/OS C/C++. Excerpt of table from z/OS C/C++ Programming Guide

Locale	Language	Country	Code set	Load module name
De_CH.IBM-500	German	Switzerland	IBM-500	EDC$DCEO
De_CH.IBM-1047	German	Switzerland	IBM-1047	EDC$DCEY
De_DE.IBM-273	German	Germany	IBM-273	EDC$DDEB
De_DE.IBM-1047	German	Germany	IBM-1047	EDC$DDEY
Fr_CA.IBM-037	French	Canada	IBM-037	EDC$FCEA
Fr_CA.IBM-1047	French	Canada	IBM-1047	EDC$FCEY
It_IT.IBM-280	Italian	Italy	IBM-280	EDC$ITEG
It_IT.IBM-1047	Italian	Italy	IBM-1047	EDC$ITEY
Ja_JP.IBM-290	Japanese	Japan	IBM-290	EDC$JAEL
Ja_JP.IBM-930	Japanese	Japan	IBM-930	EDC$JAEU
Ja_JP.IBM-939	Japanese	Japan	IBM-939	EDC$JAEV
Ja_JP.IBM-1027	Japanese	Japan	IBM-1027	EDC$JAEX

Setting up z/OS to allow uppercase and lowercase conversion of Unicode and ASCII data

If you want to use the UPPER or LOWER built-in functions to process Unicode and ASCII data, you must complete additional setup. You must also complete additional setup if you want to use uppercase or lowercase conversions for characters other than A - Z or a - z.

Procedure

To allow uppercase and lowercase conversion of Unicode and ASCII data:

Add control statements to the configuration of z/OS Support for Unicode services.

Example

Assume that your EBCDIC CCSID is 37 and your ASCII CCSID is 819, and that you have defined the following conversions:

```
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;
CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

Modify your conversion image as follows:

```
CASE NORMAL;   /* normal casing */
CASE SPECIAL;  /* additional locale-independent casing */
CASE LOCALE;   /* additional locale-dependent casing */
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;
CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

The three additional CASE statements provide the necessary infrastructure for the UPPER and LOWER functions to process Unicode data according to the Unicode Standard.

Information resources for DB2 for z/OS and related products

Many information resources are available to help you use DB2 for z/OS and many related products. A large amount of technical information about IBM products is now available online in information centers or on library websites.

Disclaimer: Any web addresses that are included here are accurate at the time this information is being published. However, web addresses sometimes change. If you visit a web address that is listed here but that is no longer valid, you can try to find the current web address for the product information that you are looking for at either of the following sites:

- http://www.ibm.com/support/publications/us/library/index.shtml, which lists the IBM information centers that are available for various IBM products
- http://www.ibm.com/shop/publications/order, which is the IBM Publications Center, where you can download online PDF books or order printed books for various IBM products

DB2 for z/OS product information

The primary place to find and use information about DB2 for z/OS is the Information Management Software for z/OS Solutions Information Center (http://publib.boulder.ibm.com/infocenter/imzic), which also contains information about IMS, QMF, and many DB2 and IMS Tools products. This information center is also available as an installable information center that can run on a local system or on an intranet server. You can order the Information Management for z/OS Solutions Information Center DVD (SK5T-7377) for a low cost from the IBM Publications Center (http://www.ibm.com/shop/publications/order).

The majority of the DB2 for z/OS information in this information center is also available in the books that are identified in the following table. You can access these books at the DB2 for z/OS library website (http://www.ibm.com/software/data/db2/zos/library.html) or at the IBM Publications Center (http://www.ibm.com/shop/publications/order).

Table 178. DB2 10 for z/OS book titles

Title	Publication number	Available in information center	Available in PDF	Available in printed format
DB2 10 for z/OS Administration Guide	SC19-2968	X	X	
DB2 10 for z/OS Application Programming & SQL Guide	SC19-2969	X	X	
DB2 10 for z/OS Application Programming Guide and Reference for Java	SC19-2970	X	X	
DB2 10 for z/OS Codes	GC19-2971	X	X	
DB2 10 for z/OS Command Reference	SC19-2972	X	X	
DB2 10 for z/OS Data Sharing: Planning and Administration	SC19-2973	X	X	
DB2 10 for z/OS Diagnosis Guide and Reference [1]	LY37-3220		X	X

© Copyright IBM Corp. 1982, 2012

Table 178. DB2 10 for z/OS book titles (continued)

Title	Publication number	Available in information center	Available in PDF	Available in printed format
DB2 10 for z/OS Installation and Migration Guide	GC19-2974	X	X	
DB2 10 for z/OS Internationalization Guide	SC19-2975	X	X	
DB2 10 for z/OS Introduction to DB2	SC19-2976	X	X	
DB2 10 for z/OS Licensed Program Specifications	GC19-2977		X	X
DB2 10 for z/OS Managing Performance	SC19-2978	X	X	
DB2 10 for z/OS Managing Security	SC19-3496	X	X	
DB2 10 for z/OS Messages	GC19-2979	X	X	
DB2 10 for z/OS ODBC Guide and Reference	SC19-2980	X	X	
DB2 10 for z/OS Program Directory	GI10-8829		X	X
DB2 10 for z/OS pureXML Guide	SC19-2981	X	X	
DB2 10 for z/OS RACF Access Control Module Guide	SC19-2982	X	X	
DB2 10 for z/OS SQL Reference	SC19-2983	X	X	
DB2 10 for z/OS Utility Guide and Reference	SC19-2984	X	X	
DB2 10 for z/OS What's New?	GC19-2985	X	X	
IRLM Messages and Codes for IMS and DB2 for z/OS	GC19-2666	X	X	

Note:

1. *DB2 10 for z/OS Diagnosis Guide and Reference* is available in PDF format on the DB2 10 for z/OS Licensed Library Collection kit, LK5T-7390. You can order this Licensed Library Collection kit on the IBM Publications Center site (http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss). This book is also available in online format in DB2 data set DSNA10.SDSNIVPD(DSNDR).

Information resources for related products

In the following table, related product names are listed in alphabetic order, and the associated web addresses of product information centers or library web pages are indicated.

Table 179. Related product information resource locations

Related product	Information resources
C/C++ for z/OS	Library website: http://www.ibm.com/software/awdtools/czos/library/ This product is now called z/OS XL C/C++.
CICS Transaction Server for z/OS	Information center: http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp
COBOL	Information center: http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp This product is now called Enterprise COBOL for z/OS.
DB2 Connect	Information center: http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp This resource is for DB2 Connect 9.

Table 179. Related product information resource locations (continued)

Related product	Information resources
DB2 Database for Linux, UNIX, and Windows	Information center: http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp This resource is for DB2 9 for Linux, UNIX, and Windows.
DB2 Query Management Facility™	Information center: http://publib.boulder.ibm.com/infocenter/imzic
DB2 Server for VSE & VM	Product website: http://www.ibm.com/software/data/db2/vse-vm/
DB2 Tools	One of the following locations: • Information center: http://publib.boulder.ibm.com/infocenter/imzic • Library website: http://www.ibm.com/software/data/db2imstools/library.html These resources include information about the following products and others: • DB2 Administration Tool • DB2 Automation Tool • DB2 Log Analysis Tool • DB2 Object Restore Tool • DB2 Query Management Facility • DB2 SQL Performance Analyzer
DB2 Universal Database™ for iSeries®	Information center: http://www.ibm.com/systems/i/infocenter/
Debug Tool for z/OS	Information center: http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp
Enterprise COBOL for z/OS	Information center: http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp
Enterprise PL/I for z/OS	Information center: http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp
InfoSphere® Replication Server for z/OS	Information center: http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.swg.im.iis.db.prod.repl.nav.doc/dochome/iiyrcnav_dochome.html This product was also known as DB2 DataPropagator, DB2 Information Integrator Replication Edition for z/OS, and WebSphere Replication Server for z/OS.
IMS	Information center: http://publib.boulder.ibm.com/infocenter/imzic
IMS Tools	One of the following locations: • Information center: http://publib.boulder.ibm.com/infocenter/imzic • Library website: http://www.ibm.com/software/data/db2imstools/library.html These resources have information about the following products and others: • IMS Batch Terminal Simulator for z/OS • IMS Connect • IMS HALDB Conversion and Maintenance Aid • IMS High Performance Utility products • IMS DataPropagator • IMS Online Reorganization Facility • IMS Performance Analyzer

Table 179. Related product information resource locations (continued)

Related product	Information resources
Integrated Data Management products	Information center: http://publib.boulder.ibm.com/infocenter/idm/v2r2/index.jsp This information center has information about the following products and others: • IBM Data Studio • InfoSphere Data Architect • InfoSphere Warehouse • Optim™ Database Administrator • Optim Development Studio • Optim Query Tuner
PL/I	Information center: http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp This product is now called Enterprise PL/I for z/OS.
System z®	http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp
Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS	Information center: http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.omegamon.xe_db2.doc/ko2welcome_pe.htm In earlier releases, this product was called DB2 Performance Expert for z/OS.
WebSphere Application Server	Information center: http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp
WebSphere Message Broker with Rules and Formatter Extension	Information center: http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp The product is also known as WebSphere MQ Integrator Broker.
WebSphere MQ	Information center: http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp The resource includes information about MQSeries.
z/Architecture	Library Center site: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/

Table 179. Related product information resource locations (continued)

Related product	Information resources
z/OS	Library Center site: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/ This resource includes information about the following z/OS elements and components: • Character Data Representation Architecture • Device Support Facilities • DFSORT • Fortran • High Level Assembler • NetView • SMP/E for z/OS • SNA • TCP/IP • TotalStorage Enterprise Storage Server • VTAM • z/OS C/C++ • z/OS Communications Server • z/OS DCE • z/OS DFSMS • z/OS DFSMS Access Method Services • z/OS DFSMSdss • z/OS DFSMShsm • z/OS DFSMSdfp • z/OS ICSF • z/OS ISPF • z/OS JES3 • z/OS Language Environment • z/OS Managed System Infrastructure • z/OS MVS • z/OS MVS JCL • z/OS Parallel Sysplex • z/OS RMF • z/OS Security Server • z/OS UNIX System Services
z/OS XL C/C++	http://www.ibm.com/software/awdtools/czos/library/

The following information resources from IBM are not necessarily specific to a single product:

- The DB2 for z/OS Information Roadmap; available at: http://www.ibm.com/software/data/db2/zos/roadmap.html
- DB2 Redbooks® and Redbooks about related products; available at: http://www.ibm.com/redbooks
- IBM Educational resources:
  - Information about IBM educational offerings is available on the web at: http://www.ibm.com/software/sw-training/

  - A collection of glossaries of IBM terms in multiple languages is available on the IBM Terminology website at: http://www.ibm.com/software/globalization/terminology/index.jsp
- National Language Support information; available at the IBM Publications Center at: http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi
- *SQL Reference for Cross-Platform Development*; available at the following developerWorks® site: http://www.ibm.com/developerworks/db2/library/techarticle/0206sqlref/0206sqlref.html

The following information resources are not published by IBM but can be useful to users of DB2 for z/OS and related products:

- Database design topics:
  - *DB2 for z/OS and OS/390 Development for Performance Volume I*, by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-605-2
  - *DB2 for z/OS and OS/390 Development for Performance Volume II*, by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-606-0
  - *Handbook of Relational Database Design*, by C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8
- Distributed Relational Database Architecture (DRDA) specifications; http://www.opengroup.org
- Domain Name System: *DNS and BIND*, Third Edition, Paul Albitz and Cricket Liu, O'Reilly, ISBN 0-59600-158-4
- Microsoft Open Database Connectivity (ODBC) information; http://msdn.microsoft.com/library/
- Unicode information; http://www.unicode.org

How to obtain DB2 information

You can access the official information about the DB2 product in a number of ways.

- "DB2 on the web"
- "DB2 product information"
- "DB2 education" on page 962
- "How to order the DB2 library" on page 962

DB2 on the web

Stay current with the latest information about DB2 by visiting the DB2 home page on the web:

http://www.ibm.com/software/db2zos

On the DB2 home page, you can find links to a wide variety of information resources about DB2. You can read news items that keep you informed about the latest enhancements to the product. Product announcements, press releases, fact sheets, and technical articles help you plan and implement your database management strategy.

DB2 product information

The official DB2 for z/OS information is available in various formats and delivery methods. IBM provides mid-version updates to the information in the information center and in softcopy updates that are available on the web and on CD-ROM.

Information Management Software for z/OS Solutions Information Center

DB2 product information is viewable in the information center, which is the primary delivery vehicle for information about DB2 for z/OS, IMS, QMF, and related tools. This information center enables you to search across related product information in multiple languages for data management solutions for the z/OS environment and print individual topics or sets of related topics. You can also access, download, and print PDFs of the publications that are associated with the information center topics. Product technical information is provided in a format that offers more options and tools for accessing, integrating, and customizing information resources. The information center is based on Eclipse open source technology.

The Information Management Software for z/OS Solutions Information Center is viewable at the following website:

http://publib.boulder.ibm.com/infocenter/imzic

CD-ROMs and DVD

Books for DB2 are available on a CD-ROM that is included with your product shipment:

- DB2 10 for z/OS Licensed Library Collection, LK5T-7390, in English

The CD-ROM contains the collection of books for DB2 10 for z/OS in PDF format. Periodically, IBM refreshes the books on subsequent editions of this CD-ROM.

© Copyright IBM Corp. 1982, 2012

The books for DB2 for z/OS are also available on the following DVD collection kit, which contains online books for many IBM products:

- IBM z/OS Software Products DVD Collection, SK3T–4271, in English

PDF format

Many of the DB2 books are available in PDF (Portable Document Format) for viewing or printing from CD-ROM or the DB2 home page on the web or from the information center. Download the PDF books to your intranet for distribution throughout your enterprise.

DB2 education

IBM Education and Training offers a wide variety of classroom courses to help you quickly and efficiently gain DB2 expertise. IBM schedules classes in cities all over the world. You can find class information, by country, at the IBM Learning Services website:

http://www.ibm.com/services/learning

IBM also offers classes at your location, at a time that suits your needs. IBM can customize courses to meet your exact requirements. For more information, including the current local schedule, contact your IBM representative.

How to order the DB2 library

To order books, visit the IBM Publication Center on the web:

http://www.ibm.com/shop/publications/order

From the IBM Publication Center, you can go to the Publication Notification System (PNS). PNS users receive electronic notifications of updated publications in their profiles. You have the option of ordering the updates by using the publications direct ordering application or any other IBM publication ordering channel. The PNS application does not send automatic shipments of publications. You will receive updated publications and a bill for them if you respond to the electronic notification.

You can also order DB2 publications and CD-ROMs from your IBM representative or the IBM branch office that serves your locality. If your location is within the United States or Canada, you can place your order by calling one of the toll-free numbers:

- In the U.S., call 1-800-879-2755.
- In Canada, call 1-800-426-4968.

To order additional copies of licensed publications, specify the SOFTWARE option. To order additional publications or CD-ROMs, specify the PUBLICATIONS option. Be prepared to give your customer number, the product number, and either the feature codes or order numbers that you want.

How to use the DB2 library

Titles of books in the library begin with DB2 10 for z/OS. However, references from one book in the library to another are shortened and do not include the product name, version, and release. Instead, they point directly to the section that holds the information. The primary place to find and use information about DB2 for z/OS is the Information Management Software for z/OS Solutions Information Center (http://publib.boulder.ibm.com/infocenter/imzic).

If you are new to DB2 for z/OS, *Introduction to DB2 for z/OS* provides a comprehensive introduction to DB2 10 for z/OS. Topics included in this book explain the basic concepts that are associated with relational database management systems in general, and with DB2 for z/OS in particular.

The most rewarding task associated with a database management system is asking questions of it and getting answers, the task called *end use*. Other tasks are also necessary—defining the parameters of the system, putting the data in place, and so on. The tasks that are associated with DB2 are grouped into the following major categories.

Installation

If you are involved with installing DB2, you will need to use a variety of resources, such as:

- *DB2 Program Directory*
- *DB2 Installation and Migration Guide*
- *DB2 Administration Guide*
- *DB2 Application Programming Guide and Reference for Java*
- *DB2 Codes*
- *DB2 Internationalization Guide*
- *DB2 Messages*
- *DB2 Managing Performance*
- *DB2 Managing Security*
- *DB2 RACF Access Control Module Guide*
- *DB2 Utility Guide and Reference*

If you will be using data sharing capabilities you also need *DB2 Data Sharing: Planning and Administration*, which describes installation considerations for data sharing.

If you will be installing and configuring DB2 ODBC, you will need *DB2 ODBC Guide and Reference*.

If you are installing IBM Spatial Support for DB2 for z/OS, you will need *IBM Spatial Support for DB2 for z/OS User's Guide and Reference*.

If you are enabling the text search feature for DB2 for z/OS, you will need *IBM Text Search for DB2 for z/OS Installation, Administration, and Reference*.

© Copyright IBM Corp. 1982, 2012

End use

End users issue SQL statements to retrieve data. They can also insert, update, or delete data, with SQL statements. They might need an introduction to SQL, detailed instructions for using SPUFI, and an alphabetized reference to the types of SQL statements. This information is found in *DB2 Application Programming and SQL Guide*, and *DB2 SQL Reference*.

End users can also issue SQL statements through the DB2 Query Management Facility (QMF) or some other program, and the library for that licensed program might provide all the instruction or reference material they need.

Application programming

Some users access DB2 without knowing it, using programs that contain SQL statements. DB2 application programmers write those programs. Because they write SQL statements, they need the same resources that end users do.

Application programmers also need instructions for many other topics:

- How to transfer data between DB2 and a host program—written in Java, C, or COBOL, for example
- How to prepare to compile a program that embeds SQL statements
- How to process data from two systems simultaneously, for example, DB2 and IMS or DB2 and CICS
- How to write distributed applications across operating systems
- How to write applications that use Open Database Connectivity (ODBC) to access DB2 servers
- How to write applications that use JDBC and SQLJ with the Java programming language to access DB2 servers
- How to write applications to store XML data on DB2 servers and retrieve XML data from DB2 servers.

The material needed for writing a host program containing SQL is in *DB2 Application Programming and SQL Guide*.

The material needed for writing applications that use JDBC and SQLJ to access DB2 servers is in *DB2 Application Programming Guide and Reference for Java*. The material needed for writing applications that use DB2 CLI or ODBC to access DB2 servers is in *DB2 ODBC Guide and Reference*. The material needed for working with XML data in DB2 is in *DB2 pureXML Guide*. For handling errors, see *DB2 Messages* and *DB2 Codes*.

System and database administration

Administration covers almost everything else. *DB2 Administration Guide* divides some of those tasks among the following sections:

- Designing a database: Discusses the decisions that must be made when designing a database and tells how to implement the design by creating and altering DB2 objects, loading data, and adjusting to changes.
- Operation and recovery: Describes the steps in normal day-to-day operation and discusses the steps one should take to prepare for recovery in the event of some failure.

DB2 Managing Performance explains how to monitor the performance of the DB2 system and its parts. It also lists things that can be done to make some parts run faster.

DB2 Managing Security describes ways of controlling access to the DB2 system and to data within DB2, ways to audit aspects of DB2 usage, and ways to answer other security and auditing concerns.

If you will be using the RACF access control module for DB2 authorization checking, you will need *DB2 RACF Access Control Module Guide*.

If you are involved with DB2 only to design the database, or plan operational procedures, you need *DB2 Administration Guide*. If you also want to carry out your own plans by creating DB2 objects, granting privileges, running utility jobs, and so on, you also need:

- *DB2 SQL Reference*, which describes the SQL statements you use to create, alter, and drop objects and grant and revoke privileges
- *DB2 Utility Guide and Reference*, which explains how to run utilities
- *DB2 Command Reference*, which explains how to run commands

If you will be using data sharing, you need *DB2 Data Sharing: Planning and Administration*, which describes how to plan for and implement data sharing.

Additional information about system and database administration can be found in *DB2 Messages* and *DB2 Codes*, which list messages and codes issued by DB2, with explanations and suggested responses.

Diagnosis

Diagnosticians detect and describe errors in the DB2 program. They might also recommend or apply a remedy. The documentation for this task is in *DB2 Diagnosis Guide and Reference*, *DB2 Messages*, and *DB2 Codes*.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

© Copyright IBM Corp. 1982, 2012

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This information is intended to help you to plan for and administer DB2 10 for z/OS. This information also documents General-use Programming Interface and Associated Guidance Information and Product-sensitive Programming Interface and Associated Guidance Information provided by DB2 10 for z/OS.

General-use Programming Interface and Associated Guidance Information

General-use Programming Interfaces allow the customer to write programs that obtain the services of DB2 10 for z/OS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs by the following markings:

GUPI General-use Programming Interface and Associated Guidance Information... GUPI

Product-sensitive Programming Interface and Associated Guidance Information

Product-sensitive Programming Interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this IBM software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive Programming Interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs by the following markings:

PSPI Product-sensitive Programming Interface and Associated Guidance Information... PSPI

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered marks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at http://www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Glossary

abend See abnormal end of task.

abend reason code
A 4-byte hexadecimal code that uniquely identifies a problem with DB2.

abnormal end of task (abend)
Termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve during execution.

access method services
The facility that is used to define, alter, delete, print, and reproduce VSAM key-sequenced data sets.

access path
The path that is used to locate data that is specified in SQL statements. An access path can be indexed or sequential.

access path stability
A characteristic of an access path that defines reliability for dynamic or static queries. Access paths are not regenerated unless there is a schema change or manual intervention.

active log
The portion of the DB2 log to which log records are written as they are generated. The active log always contains the most recent log records. See also archive log.

address space
A range of virtual storage pages that is identified by a number (ASID) and a collection of segment and page tables that map the virtual pages to real pages of the computer's memory.

address space connection
The result of connecting an allied address space to DB2. See also allied address space and task control block.

address space identifier (ASID)
A unique system-assigned identifier for an address space.

AFTER trigger
A trigger that is specified to be activated after a defined trigger event (an insert, update, or delete operation on the table that is specified in a trigger definition). Contrast with BEFORE trigger and INSTEAD OF trigger.

agent In DB2, the structure that associates all processes that are involved in a DB2 unit of work. See also allied agent and system agent.

aggregate function
An operation that derives its result by using values from one or more rows. Contrast with scalar function.

alias An alternative name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 subsystem. An alias can be qualified with a schema qualifier and can thereby be referenced by other users. Contrast with synonym.

allied address space
An area of storage that is external to DB2 and that is connected to DB2. An allied address space can request DB2 services. See also address space.

allied agent
An agent that represents work requests that originate in allied address spaces. See also system agent.

allied thread
A thread that originates at the local DB2 subsystem and that can access data at a remote DB2 subsystem.

allocated cursor
A cursor that is defined for a stored procedure result set by using the SQL ALLOCATE CURSOR statement.

ambiguous cursor
A database cursor for which DB2 cannot determine whether it is used for update or read-only purposes.

APAR See authorized program analysis report.

APF See authorized program facility.

API See application programming interface.

APPL A VTAM network definition statement that is used to define DB2 to VTAM as an application program that uses SNA LU 6.2 protocols.

© Copyright IBM Corp. 1982, 2012

application
A program or set of programs that performs a task; for example, a payroll application.

application period
A pair of columns with application-maintained values that indicates the period of time when a row is valid. See also application-period temporal table.

application-period temporal table
A table that includes an application period. See also application period and bitemporal table.

application plan
The control structure that is produced during the bind process. DB2 uses the application plan to process SQL statements that it encounters during statement execution.

application process
The unit to which resources and locks are allocated. An application process involves the execution of one or more programs.

application programming interface (API)
A functional interface that is supplied by the operating system or by a separately ordered licensed program that allows an application program that is written in a high-level language to use specific data or functions of the operating system or licensed program.

application requester
The component on a remote system that generates DRDA requests for data on behalf of an application.

application server
The target of a request from a remote application. In the DB2 environment, the application server function is provided by the distributed data facility and is used to access DB2 data from remote applications.

archive log
The portion of the DB2 log that contains log records that have been copied from the active log. See also active log.

ASCII An encoding scheme that is used to represent strings in many environments, typically on personal computers and workstations. Contrast with EBCDIC and Unicode.

ASID See address space identifier.

attachment facility
An interface between DB2 and TSO, IMS, CICS, or batch address spaces. An attachment facility allows application programs to access DB2.

attribute
A characteristic of an entity. For example, in database design, the phone number of an employee is an attribute of that employee.

authorization ID
A string that can be verified for connection to DB2 and to which a set of privileges is allowed. An authorization ID can represent an individual or an organizational group.

authorized program analysis report (APAR)
A report of a problem that is caused by a suspected defect in a current release of an IBM supplied program.

authorized program facility (APF)
A facility that allows an installation to identify system or user programs that can use sensitive system functions.

automatic bind
(More correctly *automatic rebind*.) A process by which SQL statements are bound automatically (without a user issuing a BIND command) when an application process begins execution and the bound application plan or package it requires is not valid.

automatic query rewrite
A process that examines an SQL statement that refers to one or more base tables or materialized query tables, and, if appropriate, rewrites the query so that it performs better.

auxiliary index
An index on an auxiliary table in which each index entry refers to a LOB or XML document.

auxiliary table
A table that contains columns outside the actual table in which they are defined. Auxiliary tables can contain either LOB or XML data.

backout
The process of undoing uncommitted changes that an application process made.

A backout is often performed in the event of a failure on the part of an application process, or as a result of a deadlock situation.

backward log recovery
The final phase of restart processing during which DB2 scans the log in a backward direction to apply UNDO log records for all aborted changes.

base table
A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with clone table, materialized query table, result table, temporary table, and transition table.

base table space
A table space that contains base tables.

basic row format
A row format in which values for columns are stored in the row in the order in which the columns are defined by the CREATE TABLE statement. Contrast with reordered row format.

basic sequential access method (BSAM)
An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential-access or a direct-access device.

BEFORE trigger
A trigger that is specified to be activated before a defined trigger event (an insert, an update, or a delete operation on the table that is specified in a trigger definition). Contrast with AFTER trigger and INSTEAD OF trigger.

begin column
In a system period or an application period, the column that indicates the beginning of the period. See also period.

binary large object (BLOB)
A binary string data type that contains a sequence of bytes that can range in size from 0 bytes to 2 GB, less 1 byte. This string does not have an associated code page and character set. BLOBs can contain, for example, image, audio, or video data. In general, BLOB values are used whenever a binary string might exceed the limits of the VARBINARY type.

binary string
A sequence of bytes that is not associated with a CCSID. Binary string data type can be further classified as BINARY, VARBINARY, or BLOB.

binary XML format
A representation of XML data that uses binary values, an approach that facilitates more efficient storage and exchange.

bind A process by which a usable control structure with SQL statements is generated; the structure is often called an access plan, an application plan, or a package. During this bind process, access paths to the data are selected, and some authorization checking is performed. See also automatic bind.

bit data

- Data with character type CHAR or VARCHAR that is defined with the FOR BIT DATA clause. Note that using BINARY or VARBINARY rather than FOR BIT DATA is highly recommended.
- Data with character type CHAR or VARCHAR that is defined with the FOR BIT DATA clause.
- A form of character data. Binary data is generally more highly recommended than character-for-bit data.

bitemporal table
A table that is both a system-period temporal table and an application-period temporal table. See also application-period temporal table and system-period temporal table.

BLOB See binary large object.

block fetch
A capability in which DB2 can retrieve, or fetch, a large set of rows together. Using block fetch can significantly reduce the number of messages that are being sent across the network. Block fetch applies only to non-rowset cursors that do not update data.

bootstrap data set (BSDS)
A VSAM data set that contains name and status information for DB2 and RBA range specifications for all active and archive log data sets. The BSDS also contains passwords for the DB2 directory

and catalog, and lists of conditional restart and checkpoint records.

BSAM
See basic sequential access method.

BSDS See bootstrap data set.

buffer pool
An area of memory into which data pages are read, modified, and held during processing.

built-in data type
A data type that IBM supplies. Among the built-in data types for DB2 for z/OS are string, numeric, XML, ROWID, and datetime. Contrast with distinct type.

built-in function
A function that is generated by DB2 and that is in the SYSIBM schema. Contrast with user-defined function. See also function, cast function, external function, sourced function, and SQL function.

business dimension
A category of data, such as products or time periods, that an organization might want to analyze.

cache structure
A coupling facility structure that stores data that can be available to all members of a Sysplex. A DB2 data sharing group uses cache structures as group buffer pools.

CAF See call attachment facility.

call attachment facility (CAF)
A DB2 attachment facility for application programs that run in TSO or z/OS batch. The CAF is an alternative to the DSN command processor and provides greater control over the execution environment. Contrast with Resource Recovery Services attachment facility.

call-level interface (CLI)
A callable application programming interface (API) for database access, which is an alternative to using embedded SQL.

cascade delete
A process by which DB2 enforces referential constraints by deleting all descendent rows of a deleted parent row.

CASE expression
An expression that is selected based on the evaluation of one or more conditions.

cast function
A function that is used to convert instances of a (source) data type into instances of a different (target) data type.

castout
The DB2 process of writing changed pages from a group buffer pool to disk.

castout owner
The DB2 member that is responsible for casting out a particular page set or partition.

catalog
In DB2, a collection of tables that contains descriptions of objects such as tables, views, and indexes.

catalog table
Any table in the DB2 catalog.

CCSID
See coded character set identifier.

CDB See communications database.

CDRA
See Character Data Representation Architecture.

CEC See central processor complex.

central electronic complex (CEC)
See central processor complex.

central processor complex (CPC)
A physical collection of hardware that consists of main storage, one or more central processors, timers, and channels.

central processor (CP)
The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

CFRM See coupling facility resource management.

CFRM policy
The allocation rules for a coupling facility structure that are declared by a z/OS administrator.

character conversion
The process of changing characters from one encoding scheme to another.

Character Data Representation Architecture (CDRA)
An architecture that is used to achieve

consistent representation, processing, and interchange of string data.

character large object (CLOB)
A character string data type that contains a sequence of bytes that represent characters (single-byte, multibyte, or both) that can range in size from 0 bytes to 2 GB, less 1 byte. In general, CLOB values are used whenever a character string might exceed the limits of the VARCHAR type.

character set
A defined set of characters.

character string
A sequence of bytes that represent bit data, single-byte characters, or a mixture of single-byte and multibyte characters. Character data can be further classified as CHARACTER, VARCHAR, or CLOB.

check constraint
A user-defined constraint that specifies the values that specific columns of a base table can contain.

check integrity
The condition that exists when each row in a table conforms to the check constraints that are defined on that table.

check pending
A state of a table space or partition that prevents its use by some utilities and by some SQL statements because of rows that violate referential constraints, check constraints, or both.

checkpoint
A point at which DB2 records status information on the DB2 log; the recovery process uses this information if DB2 abnormally terminates.

child lock
For explicit hierarchical locking, a lock that is held on either a table, page, row, or a large object (LOB). Each child lock has a parent lock. See also parent lock.

CI See control interval.

CICS Represents (in this information): CICS Transaction Server for z/OS: Customer Information Control System Transaction Server for z/OS.

CICS attachment facility
A facility that provides a multithread connection to DB2 to allow applications that run in the CICS environment to execute DB2 statements.

claim A notification to DB2 that an object is being accessed. Claims prevent drains from occurring until the claim is released, which usually occurs at a commit point. Contrast with drain.

claim class
A specific type of object access that can be one of the following isolation levels:

- Cursor stability (CS)
- Repeatable read (RR)
- Write

class of service
A VTAM term for a list of routes through a network, arranged in an order of preference for their use.

clause In SQL, a distinct part of a statement, such as a SELECT clause or a WHERE clause.

CLI See call-level interface.

client See requester.

CLOB See character large object.

clone object
An object that is associated with a clone table, including the clone table itself and check constraints, indexes, and BEFORE triggers on the clone table.

clone table
A table that is structurally identical to a base table. The base and clone table each have separate underlying VSAM data sets, which are identified by their data set instance numbers. Contrast with base table.

closed application
An application that requires exclusive use of certain statements on certain DB2 objects, so that the objects are managed solely through the external interface of that application.

clustering index
An index that determines how rows are physically ordered (*clustered*) in a table space. If a clustering index on a partitioned table is not a partitioning index, the rows are ordered in cluster sequence within each data partition instead of spanning partitions.

CM See conversion mode.

CM* See conversion mode*.

C++ member
A data object or function in a structure, union, or class.

C++ member function
An operator or function that is declared as a member of a class. A member function has access to the private and protected data members and to the member functions of objects in its class. Member functions are also called methods.

C++ object
A region of storage. An object is created when a variable is defined or a new function is invoked.

An instance of a class.

coded character set
A set of unambiguous rules that establish a character set and the one-to-one relationships between the characters of the set and their coded representations.

coded character set identifier (CCSID)
A 16-bit number that uniquely identifies a coded representation of graphic characters. It designates an encoding scheme identifier and one or more pairs that consist of a character set identifier and an associated code page identifier.

code page
A set of assignments of characters to code points. Within a code page, each code point has only one specific meaning. In EBCDIC, for example, the character *A* is assigned code point X'C1', and character *B* is assigned code point X'C2'.

code point
In CDRA, a unique bit pattern that represents a character in a code page.

code unit
The fundamental binary width in a computer architecture that is used for representing character data, such as 7 bits, 8 bits, 16 bits, or 32 bits. Depending on the character encoding form that is used, each code point in a coded character set can be represented by one or more code units.

coexistence
During migration, the period of time in which two releases exist in the same data sharing group.

cold start
A process by which DB2 restarts without processing any log records. Contrast with warm start.

collection
A group of packages that have the same qualifier.

column
The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer).

column function
See aggregate function.

"come from" checking
An LU 6.2 security option that defines a list of authorization IDs that are allowed to connect to DB2 from a partner LU.

command
A DB2 operator command or a DSN subcommand. A command is distinct from an SQL statement.

command prefix
A 1- to 8-character command identifier. The command prefix distinguishes the command as belonging to an application or subsystem rather than to z/OS.

command recognition character (CRC)
A character that permits a z/OS console operator or an IMS subsystem user to route DB2 commands to specific DB2 subsystems.

command scope
The scope of command operation in a data sharing group.

commit
The operation that ends a unit of work by releasing locks so that the database changes that are made by that unit of work can be perceived by other processes. Contrast with rollback.

commit point
A point in time when data is considered consistent.

common service area (CSA)
In z/OS, a part of the common area that

contains data areas that are addressable by all address spaces. Most DB2 use is in the extended CSA, which is above the 16-MB line.

communications database (CDB)
A set of tables in the DB2 catalog that are used to establish conversations with remote database management systems.

comparison operator
A token (such as =, >, or <) that is used to specify a relationship between two values.

compatibility mode
See conversion mode.

compatibility mode* (CM*)
See conversion mode*.

composite key
An ordered set of key columns or expressions of the same table.

compression dictionary
The dictionary that controls the process of compression and decompression. This dictionary is created from the data in the table space or table space partition.

concurrency
The shared use of resources by more than one application process at the same time.

conditional restart
A DB2 restart that is directed by a user-defined conditional restart control record (CRCR).

connection
In SNA, the existence of a communication path between two partner LUs that allows information to be exchanged (for example, two DB2 subsystems that are connected and communicating by way of a conversation).

connection context
In SQLJ, a Java object that represents a connection to a data source.

connection declaration clause
In SQLJ, a statement that declares a connection to a data source.

connection handle
The data object containing information that is associated with a connection that DB2 ODBC manages. This includes general status information, transaction status, and diagnostic information.

connection ID
An identifier that is supplied by the attachment facility and that is associated with a specific address space connection.

consistency token
A timestamp that is used to generate the version identifier for an application. See also version.

constant
A language element that specifies an unchanging value. Constants are classified as string constants or numeric constants. Contrast with variable.

constraint
A rule that limits the values that can be inserted, deleted, or updated in a table. See referential constraint, check constraint, and unique constraint.

context
An application's logical connection to the data source and associated DB2 ODBC connection information that allows the application to direct its operations to a data source. A DB2 ODBC context represents a DB2 thread.

contracting conversion
A process that occurs when the length of a converted string is smaller than that of the source string. For example, this process occurs when an EBCDIC mixed-data string that contains DBCS characters is converted to ASCII mixed data; the converted string is shorter because the shift codes are removed.

control interval (CI)
- A unit of information that VSAM transfers between virtual and auxiliary storage.
- In a key-sequenced data set or file, the set of records that an entry in the sequence-set index record points to.

conversation
Communication, which is based on LU 6.2 or Advanced Program-to-Program Communication (APPC), between an application and a remote transaction program over an SNA logical unit-to-logical unit (LU-LU) session that allows communication while processing a transaction.

conversion mode* (CM*)
A stage of the version-to-version migration process that applies to a DB2 subsystem or data sharing group that was in enabling-new-function mode (ENFM), enabling-new-function mode* (ENFM*), or new-function mode (NFM) at one time. Fallback to a prior version is not supported. When in conversion mode*, a DB2 data sharing group cannot coexist with members that are still at the prior version level. Contrast with conversion mode, enabling-new-function mode, enabling-new-function mode*, and new-function mode.

Previously known as compatibility mode* (CM*).

conversion mode (CM)
The first stage of the version-to-version migration process. In a DB2 data sharing group, members in conversion mode can coexist with members that are still at the prior version level. Fallback to the prior version is also supported. When in conversion mode, the DB2 subsystem cannot use most new functions of the new version. Contrast with conversion mode*, enabling-new-function mode, enabling-new-function mode*, and new-function mode.

Previously known as compatibility mode (CM).

coordinator
The system component that coordinates the commit or rollback of a unit of work that includes work that is done on one or more other systems.

coprocessor
See SQL statement coprocessor.

copy pool
A collection of names of storage groups that are processed collectively for fast replication operations.

copy target
A named set of SMS storage groups that are to be used as containers for copy pool volume copies. A copy target is an SMS construct that lets you define which storage groups are to be used as containers for volumes that are copied by using FlashCopy functions.

copy version
A point-in-time FlashCopy copy that is managed by HSM. Each copy pool has a version parameter that specifies the number of copy versions to be maintained on disk.

correlated columns
A relationship between the value of one column and the value of another column.

correlated subquery
A subquery (part of a WHERE or HAVING clause) that is applied to a row or group of rows of a table or view that is named in an outer subselect statement.

correlation ID
An identifier that is associated with a specific thread. In TSO, it is either an authorization ID or the job name.

correlation name
An identifier that is specified and used within a single SQL statement as the exposed name for objects such as a table, view, table function reference, nested table expression, or result of a data change statement. Correlation names are useful in an SQL statement to allow two distinct references to the same base table and to allow an alternative name to be used to represent an object.

cost category
A category into which DB2 places cost estimates for SQL statements at the time the statement is bound. The cost category is externalized in the COST_CATEGORY column of the DSN_STATEMNT_TABLE when a statement is explained.

coupling facility
A special PR/SM logical partition (LPAR) that runs the coupling facility control program and provides high-speed caching, list processing, and locking functions in a Parallel Sysplex.

coupling facility resource management (CFRM)
A component of z/OS that provides the services to manage coupling facility resources in a Parallel Sysplex. This management includes the enforcement of CFRM policies to ensure that the coupling facility and structure requirements are satisfied.

CP See central processor.

CPC See central processor complex.

CRC See command recognition character.

created temporary table
A persistent table that holds temporary data and is defined with the SQL statement CREATE GLOBAL TEMPORARY TABLE. Information about created temporary tables is stored in the DB2 catalog and can be shared across application processes. Contrast with declared temporary table. See also temporary table.

cross-system coupling facility (XCF)
A component of z/OS that provides functions to support cooperation between authorized programs that run within a Sysplex.

cross-system extended services (XES)
A set of z/OS services that allow multiple instances of an application or subsystem, running on different systems in a Sysplex environment, to implement high-performance, high-availability data sharing by using a coupling facility.

CS See cursor stability.

CSA See common service area.

CT See cursor table.

current data
Data within a host structure that is current with (identical to) the data within the base table.

current status rebuild
The second phase of restart processing during which the status of the subsystem is reconstructed from information on the log.

cursor A control structure that an application program uses to point to a single row or multiple rows within some ordered set of rows of a result table. A cursor can be used to retrieve, update, or delete rows from a result table.

cursor sensitivity
The degree to which database updates are visible to the subsequent FETCH statements in a cursor.

cursor stability (CS)
The isolation level that provides maximum concurrency without the ability to read uncommitted data. With cursor stability, a unit of work holds locks only on its uncommitted changes and on the current row of each of its cursors. See also read stability, repeatable read, and uncommitted read.

cursor table (CT)
The internal representation of a cursor.

cycle A set of tables that can be ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table. A self-referencing table is a cycle with a single member. See also referential cycle.

database
A collection of tables, or a collection of table spaces and index spaces.

database access thread (DBAT)
A thread that accesses data at the local subsystem on behalf of a remote subsystem.

database administrator (DBA)
An individual who is responsible for designing, developing, operating, safeguarding, maintaining, and using a database.

database alias
The name of the target server if it is different from the location name. The database alias is used to provide the name of the database server as it is known to the network.

database descriptor (DBD)
An internal representation of a DB2 database definition, which reflects the data definition that is in the DB2 catalog. The objects that are defined in a database descriptor are table spaces, tables, indexes, index spaces, relationships, check constraints, and triggers. A DBD also contains information about accessing tables in the database.

database exception status
In a data sharing environment, an indication that something is wrong with a database.

database identifier (DBID)
An internal identifier of the database.

database management system (DBMS)
A software system that controls the

creation, organization, and modification of a database and the access to the data that is stored within it.

database request module (DBRM)
A data set member that is created by the DB2 precompiler and that contains information about SQL statements. DBRMs are used in the bind process.

database server
The target of a request from a local application or a remote intermediate database server.

data currency
The state in which the data that is retrieved into a host variable in a program is a copy of the data in the base table.

data-dependent pagination
The process used when applications need to access part of a DB2 result set that is based on a logical key value.

data dictionary
A repository of information about an organization's application programs, databases, logical data models, users, and authorizations.

data partition
A VSAM data set that is contained within a partitioned table space.

data-partitioned secondary index (DPSI)
A secondary index that is partitioned according to the underlying data. Contrast with nonpartitioned secondary index.

data set instance number
A number that indicates the data set that contains the data for an object.

data sharing
A function of DB2 for z/OS that enables applications on different DB2 subsystems to read from and write to the same data concurrently.

data sharing group
A collection of one or more DB2 subsystems that directly access and change the same data while maintaining data integrity.

data sharing member
A DB2 subsystem that is assigned by XCF services to a data sharing group.

data source
A local or remote relational or non-relational data manager that is capable of supporting data access via an ODBC driver that supports the ODBC APIs. In the case of DB2 for z/OS, the data sources are always relational database managers.

data type
An attribute of columns, constants, variables, parameters, special registers, and the results of functions and expressions.

data warehouse
A system that provides critical business information to an organization. The data warehouse system cleanses the data for accuracy and currency, and then presents the data to decision makers so that they can interpret and use it effectively and efficiently.

DBA See database administrator.

DBAT See database access thread.

DB2 catalog
A collection of tables that are maintained by DB2 and contain descriptions of DB2 objects, such as tables, views, and indexes.

DBCLOB
See double-byte character large object.

DB2 command
An instruction to the DB2 subsystem that a user enters to start or stop DB2, to display information on current users, to start or stop databases, to display information on the status of databases, and so on.

DBCS See double-byte character set.

DBD See database descriptor.

DB2I See DB2 Interactive.

DBID See database identifier.

DB2 Interactive (DB2I)
An interactive service within DB2 that facilitates the execution of SQL statements, DB2 (operator) commands, and programmer commands, and the invocation of utilities.

DBMS
See database management system.

DBRM
See database request module.

DB2 thread
The database manager structure that describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to the database manager resources and services. Most DB2 for z/OS functions execute under a thread structure.

DCLGEN
See declarations generator.

DDF See distributed data facility.

deadlock
Unresolved contention for the use of a resource, such as a table or an index.

declarations generator (DCLGEN)
A subcomponent of DB2 that generates SQL table declarations and COBOL, C, or PL/I data structure declarations that conform to the table. The declarations are generated from DB2 system catalog information.

declared temporary table
A non-persistent table that holds temporary data and is defined with the SQL statement DECLARE GLOBAL TEMPORARY TABLE. Information about declared temporary tables is not stored in the DB2 catalog and can be used only by the application process that issued the DECLARE statement. Contrast with created temporary table. See also temporary table.

default value
A predetermined value, attribute, or option that is assumed when no other value is specified. A default value can be defined for column data in DB2 tables by specifying the DEFAULT keyword in an SQL statement that changes data (such as INSERT, UPDATE, and MERGE).

deferred embedded SQL
SQL statements that are neither fully static nor fully dynamic. These statements are embedded within an application and are prepared during the execution of the application.

deferred write
The process of asynchronously writing changed data pages to disk.

degree of parallelism
The number of concurrently executed operations that are initiated to process a query.

delete hole
The location on which a cursor is positioned when a row in a result table is refetched and the row no longer exists on the base table. See also update hole.

delete rule
The rule that tells DB2 what to do to a dependent row when a parent row is deleted. Delete rules include CASCADE, RESTRICT, SET NULL, or NO ACTION.

delete trigger
A trigger that is defined with the triggering delete SQL operation.

delimited identifier
A sequence of one or more characters enclosed by escape characters, such as quotation marks ("").

delimiter token
A string constant, a delimited identifier, an operator symbol, or any of the special characters that are shown in DB2 syntax diagrams.

denormalization
The intentional duplication of columns in multiple tables to increase data redundancy. Denormalization is sometimes necessary to minimize performance problems. Contrast with normalization.

dependent
An object (row, table, or table space) that has at least one parent. The object is also said to be a dependent (row, table, or table space) of its parent. See also parent row, parent table, and parent table space.

dependent row
A row that contains a foreign key that matches the value of a primary key in the parent row.

dependent table
A table that is a dependent in at least one referential constraint.

descendent
An object that is a dependent of an object or is the dependent of a descendent of an object.

descendent row
A row that is dependent on another row, or a row that is a descendent of a dependent row.

descendent table
A table that is a dependent of another table, or a table that is a descendent of a dependent table.

deterministic function
A user-defined function whose result is dependent on the values of the input arguments. That is, successive invocations with the same input values produce the same answer. Sometimes referred to as a *not-variant* function. Contrast with nondeterministic function (sometimes called a *variant function*).

dimension
A data category such as time, products, or markets. The elements of a dimension are referred to as members. See also dimension table.

dimension table
The representation of a dimension in a star schema. Each row in a dimension table represents all of the attributes for a particular member of the dimension. See also dimension, star schema, and star join.

directory
The DB2 system database that contains internal objects such as database descriptors and skeleton cursor tables.

disk A direct-access storage device that records data magnetically.

distinct type
A user-defined data type that is represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

distributed data
Data that resides on a DBMS other than the local system.

distributed data facility (DDF)
A set of DB2 components through which DB2 communicates with another relational database management system.

Distributed Relational Database Architecture (DRDA)
A connection protocol for distributed relational database processing that is used by IBM relational database products. DRDA includes protocols for communication between an application and a remote relational database management system, and for communication between relational database management systems. See also DRDA access.

DNS See domain name server.

DOCID
See document ID.

document ID
A value that uniquely identifies a row that contains an XML column. This value is stored with the row and never changes.

domain
The set of valid values for an attribute.

domain name
The name by which TCP/IP applications refer to a TCP/IP host within a TCP/IP network.

domain name server (DNS)
A special TCP/IP network server that manages a distributed directory that is used to map TCP/IP host names to IP addresses.

double-byte character large object (DBCLOB)
A graphic string data type in which a sequence of bytes represent double-byte characters that range in size from 0 bytes to 2 GB, less 1 byte. In general, DBCLOB values are used whenever a double-byte character string might exceed the limits of the VARGRAPHIC type.

double-byte character set (DBCS)
A set of characters, which are used by national languages such as Japanese and Chinese, that have more symbols than can be represented by a single byte. Each character is 2 bytes in length. Contrast with single-byte character set and multibyte character set.

double-precision floating point number
A 64-bit approximate representation of a real number.

DPSI See data-partitioned secondary index.

drain The act of acquiring a locked resource by quiescing access to that object. Contrast with claim.

drain lock
A lock on a claim class that prevents a claim from occurring.

DRDA
See Distributed Relational Database Architecture.

DRDA access
An open method of accessing distributed data that you can use to connect to another database server to execute packages that were previously bound at the server location.

DSN

- The default DB2 subsystem name.
- The name of the TSO command processor of DB2.
- The first three characters of DB2 module and macro names.

dynamic cursor
A named control structure that an application program uses to change the size of the result table and the order of its rows after the cursor is opened. Contrast with static cursor.

dynamic dump
A dump that is issued during the execution of a program, usually under the control of that program.

dynamic SQL
SQL statements that are prepared and executed at run time. In dynamic SQL, the SQL statement is contained as a character string in a host variable or as a constant, and it is not precompiled.

EA-enabled table space
A table space or index space that is enabled for extended addressability and that contains individual partitions (or pieces, for LOB table spaces) that are greater than 4 GB.

EB See exabyte.

EBCDIC
Extended binary coded decimal interchange code. An encoding scheme that is used to represent character data in the z/OS, VM, VSE, and iSeries environments. Contrast with ASCII and Unicode.

embedded SQL
SQL statements that are coded within an application program. See static SQL.

enabling-new-function mode* (ENFM*)
A transitional stage of the version-to-version migration process that applies to a DB2 subsystem or data sharing group that was in new-function mode (NFM) at one time. When in enabling-new-function mode*, a DB2 subsystem or data sharing group is preparing to use the new functions of the new version but cannot yet use them. A data sharing group that is in enabling-new-function mode* cannot coexist with members that are still at the prior version level. Fallback to a prior version is not supported. Contrast with conversion mode, conversion mode*, enabling-new-function mode, and new-function mode.

enabling-new-function mode (ENFM)
A transitional stage of the version-to-version migration process during which the DB2 subsystem or data sharing group is preparing to use the new functions of the new version. When in enabling-new-function mode, a DB2 data sharing group cannot coexist with members that are still at the prior version level. Fallback to a prior version is not supported, and most new functions of the new version are not available for use in enabling-new-function mode. Contrast with conversion mode, conversion mode*, enabling-new-function mode*, and new-function mode.

enclave
In Language Environment , an independent collection of routines, one of which is designated as the main routine. An enclave is similar to a program or run unit. See also WLM enclave.

encoding scheme
A set of rules to represent character data (ASCII, EBCDIC, or Unicode).

end column
In a system period or an application period, the column that indicates the end of the period. See also period.

ENFM See enabling-new-function mode.

ENFM*
See enabling-new-function mode*.

entity A person, object, or concept about which information is stored. In a relational database, entities are represented as tables. A database includes information about the entities in an organization or business, and their relationships to each other.

enumerated list
A set of DB2 objects that are defined with a LISTDEF utility control statement in which pattern-matching characters (*, %;, _, or ?) are not used.

environment
A collection of names of logical and physical resources that are used to support the performance of a function.

environment handle
A handle that identifies the global context for database access. All data that is pertinent to all objects in the environment is associated with this handle.

equijoin
A join operation in which the join-condition has the form *expression* = *expression*. See also join, full outer join, inner join, left outer join, outer join, and right outer join.

error page range
A range of pages that are considered to be physically damaged. DB2 does not allow users to access any pages that fall within this range.

escape character
The symbol, a double quotation (") for example, that is used to enclose an SQL delimited identifier.

exabyte
A unit of measure for processor, real and virtual storage capacities, and channel volume that has a value of 1 152 921 504 606 846 976 bytes or 2^{60}.

exception
An SQL operation that involves the EXCEPT set operator, which combines two result tables. The result of an exception operation consists of all of the rows that are in only one of the result tables.

exception table
A table that holds rows that violate referential constraints or check constraints that the CHECK DATA utility finds.

exclusive lock
A lock that prevents concurrently executing application processes from reading or changing data. Contrast with share lock.

executable statement
An SQL statement that can be embedded in an application program, dynamically prepared and executed, or issued interactively.

execution context
In SQLJ, a Java object that can be used to control the execution of SQL statements.

exit routine
A user-written (or IBM-provided default) program that receives control from DB2 to perform specific functions. Exit routines run as extensions of DB2.

expanding conversion
A process that occurs when the length of a converted string is greater than that of the source string. For example, this process occurs when an ASCII mixed-data string that contains DBCS characters is converted to an EBCDIC mixed-data string; the converted string is longer because shift codes are added.

explicit hierarchical locking
Locking that is used to make the parent-child relationship between resources known to IRLM. This kind of locking avoids global locking overhead when no inter-DB2 interest exists on a resource.

explicit privilege
A privilege that has a name and is held as the result of an SQL GRANT statement and revoked as the result of an SQL REVOKE statement. For example, the SELECT privilege.

exposed name
A correlation name or a table or view name for which a correlation name is not specified.

expression
An operand or a collection of operators and operands that yields a single value.

Extended Recovery Facility (XRF)
A facility that minimizes the effect of failures in z/OS, VTAM, the host processor, or high-availability applications during sessions between high-availability applications and designated terminals. This facility provides an alternative subsystem to take over sessions from the failing subsystem.

Extensible Markup Language (XML)
A standard metalanguage for defining markup languages that is a subset of Standardized General Markup Language (SGML).

external function
A function that has its functional logic implemented in a programming language application that resides outside the database, in the file system of the database server. The association of the function with the external code application is specified by the EXTERNAL clause in the CREATE FUNCTION statement. External functions can be classified as external scalar functions and external table functions. Contrast with sourced function, built-in function, and SQL function.

external procedure
A procedure that has its procedural logic implemented in an external programming language application. The association of the procedure with the external application is specified by a CREATE PROCEDURE statement with a LANGUAGE clause that has a value other than SQL and an EXTERNAL clause that implicitly or explicitly specifies the name of the external application. Contrast with external SQL procedure and native SQL procedure.

external routine
A user-defined function or stored procedure that is based on code that is written in an external programming language.

external SQL procedure
An SQL procedure that is processed using a generated C program that is a representation of the procedure. When an external SQL procedure is called, the C program representation of the procedure is executed in a stored procedures address space. Contrast with external procedure and native SQL procedure.

failed member state
A state of a member of a data sharing group in which the member's task, address space, or z/OS system terminates before the state changes from active to quiesced.

fallback
The process of returning to a previous release of DB2 after attempting or completing migration to a current release. Fallback is supported only from a subsystem that is in conversion mode.

false global lock contention
A contention indication from the coupling facility that occurs when multiple lock names are hashed to the same indicator and when no real contention exists.

fan set
A direct physical access path to data, which is provided by an index, hash, or link; a fan set is the means by which DB2 supports the ordering of data.

federated database
The combination of a DB2 server (in Linux, UNIX, and Windows environments) and multiple data sources to which the server sends queries. In a federated database system, a client application can use a single SQL statement to join data that is distributed across multiple database management systems and can view the data as if it were local.

fetch orientation
The specification of the desired placement of the cursor as part of a FETCH statement. The specification can be before or after the rows of the result table (with BEFORE or AFTER). The specification can also have either a single-row fetch orientation (for example, NEXT, LAST, or ABSOLUTE *n*) or a rowset fetch orientation (for example, NEXT ROWSET, LAST ROWSET, or ROWSET STARTING AT ABSOLUTE *n*).

field procedure
A user-written exit routine that is designed to receive a single value and transform (encode or decode) it in any way the user can specify.

file reference variable
A host variable that is declared with one of the derived data types (BLOB_FILE, CLOB_FILE, DBCLOB_FILE); file reference variables direct the reading of a LOB from a file or the writing of a LOB into a file.

filter factor
A number between zero and one that estimates the proportion of rows in a table for which a predicate is true.

fixed-length string
A character, graphic, or binary string whose length is specified and cannot be changed. Contrast with varying-length string.

FlashCopy
A function on the IBM Enterprise Storage Server that can, in conjunction with the BACKUP SYSTEM utility, create a point-in-time copy of data while an application is running.

foreign key
A column or set of columns in a dependent table of a constraint relationship. The key must have the same number of columns, with the same descriptions, as the primary key of the parent table. Each foreign key value must either match a parent key value in the related parent table or be null.

forest An ordered set of subtrees of XML nodes.

forward log recovery
The third phase of restart processing during which DB2 processes the log in a forward direction to apply all REDO log records.

free space
The total amount of unused space in a page; that is, the space that is not used to store records or control information is free space.

full outer join
The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of both tables. See also join, equijoin, inner join, left outer join, outer join, and right outer join.

fullselect
A subselect, a fullselect in parentheses, or a number of both that are combined by set operators. Fullselect specifies a result table. If a set operator is not used, the result of the fullselect is the result of the specified subselect or fullselect.

fully escaped mapping
A mapping from an SQL identifier to an XML name when the SQL identifier is a column name.

function
A mapping, which is embodied as a program (the function body) that can be invoked by means of zero or more input values (arguments) to a single value (the result). See also aggregate function and scalar function.

Functions can be user-defined, built-in, or generated by DB2. (See also built-in function, cast function, external function, sourced function, SQL function, and user-defined function.)

function definer
The authorization ID of the owner of the schema of the function that is specified in the CREATE FUNCTION statement.

function package
A package that results from binding the DBRM for a function program.

function package owner
The authorization ID of the user who binds the function program's DBRM into a function package.

function signature
The logical concatenation of a fully qualified function name with the data types of all of its parameters.

GB Gigabyte. A value of (1 073 741 824 bytes).

GBP See group buffer pool.

GBP-dependent
The status of a page set or page set partition that is dependent on the group buffer pool. Either read/write interest is active among DB2 subsystems for this page set, or the page set has changed pages in the group buffer pool that have not yet been cast out to disk.

generalized trace facility (GTF)
A z/OS service program that records significant system events such as I/O

interrupts, SVC interrupts, program interrupts, or external interrupts.

generated column
A column for which the database manager assigns the value. An example of a generated column is an identity column, row change timestamp column, or row-begin column. See also generated expression column.

generated expression column
A generated column that is defined using an expression. See also generated column.

generic resource name
A name that VTAM uses to represent several application programs that provide the same function in order to handle session distribution and balancing in a Sysplex environment.

getpage
An operation in which DB2 accesses a data page.

global lock
A lock that provides concurrency control within and among DB2 subsystems. The scope of the lock is across all DB2 subsystems of a data sharing group.

global lock contention
Conflicts on locking requests between different DB2 members of a data sharing group when those members are trying to serialize shared resources.

governor
See resource limit facility.

graphic string
A sequence of DBCS characters. Graphic data can be further classified as GRAPHIC, VARGRAPHIC, or DBCLOB.

GRECP
See group buffer pool recovery pending.

gross lock
The *shared*, *update*, or *exclusive* mode locks on a table, partition, or table space.

group buffer pool duplexing
The ability to write data to two instances of a group buffer pool structure: a primary group buffer pool and a secondary group buffer pool. z/OS publications refer to these instances as the "old" (for primary) and "new" (for secondary) structures.

group buffer pool (GBP)
A coupling facility cache structure that is used by a data sharing group to cache data and to ensure that the data is consistent for all members.

group buffer pool recovery pending (GRECP)
The state that exists after the buffer pool for a data sharing group is lost. When a page set is in this state, changes that are recorded in the log must be applied to the affected page set before the page set can be used.

group level
The release level of a data sharing group, which is established when the first member migrates to a new release.

group name
The z/OS XCF identifier for a data sharing group.

group restart
A restart of at least one member of a data sharing group after the loss of either locks or the shared communications area.

GTF See generalized trace facility.

handle
In DB2 ODBC, a variable that refers to a data structure and associated resources. See also statement handle, connection handle, and environment handle.

hash access
Access to a table using the hash value of a key that is defined by the *organization-clause* of a CREATE TABLE statement or ALTER TABLE statement.

hash overflow index
A DB2 index used to track data rows that do not fit into the fixed hash space, and therefore, reside in the hash overflow space. DB2 accesses the hash overflow index to fetch rows from the hash overflow area.

help panel
A screen of information that presents tutorial text to assist a user at the workstation or terminal.

heuristic damage
The inconsistency in data between one or more participants that results when a heuristic decision to resolve an indoubt

LUW at one or more participants differs from the decision that is recorded at the coordinator.

heuristic decision
A decision that forces indoubt resolution at a participant by means other than automatic resynchronization between coordinator and participant.

histogram statistics
A way of summarizing data distribution. This technique divides up the range of possible values in a data set into intervals, such that each interval contains approximately the same percentage of the values. A set of statistics are collected for each interval.

historical row
A row in a history table. See also history table.

history table
A table that is used by the database manager to store historical versions of the rows from the associated system-period temporal table. See also historical row and system-period temporal table.

hole A row of the result table that cannot be accessed because of a delete or an update that has been performed on the row. See also delete hole and update hole.

home address space
The area of storage that z/OS currently recognizes as *dispatched*.

host The set of programs and resources that are available on a given TCP/IP instance.

host expression
A Java variable or expression that is referenced by SQL clauses in an SQLJ application program.

host identifier
A name that is declared in the host program.

host language
A programming language in which you can embed SQL statements.

host program
An application program that is written in a host language and that contains embedded SQL statements.

host structure
In an application program, a structure that is referenced by embedded SQL statements.

host variable
In an application program written in a host language, an application variable that is referenced by embedded SQL statements.

host variable array
An array of elements, each of which corresponds to a value for a column. The dimension of the array determines the maximum number of rows for which the array can be used.

IBM System z9 Integrated Processor (zIIP)
A specialized processor that can be used for some DB2 functions.

IDCAMS
An IBM program that is used to process access method services commands. It can be invoked as a job or jobstep, from a TSO terminal, or from within a user's application program.

IDCAMS LISTCAT
A facility for obtaining information that is contained in the access method services catalog.

identity column
A generated column that is defined with the AS IDENTITY clause. An identity column provides a way for the database manager to automatically generate a numeric value for each row that is inserted into a table. A table can have no more than one identity column.

IFCID See instrumentation facility component identifier.

IFI See instrumentation facility interface.

IFI call
An invocation of the instrumentation facility interface (IFI) by means of one of its defined functions.

image copy
An exact reproduction of all or part of a table space. DB2 provides utility programs to make full image copies (to copy the entire table space) or incremental image copies (to copy only those pages that have been modified since the last image copy).

IMS attachment facility
A DB2 subcomponent that uses z/OS subsystem interface (SSI) protocols and cross-memory linkage to process requests from IMS to DB2 and to coordinate resource commitment.

in-abort
A status of a unit of recovery. If DB2 fails after a unit of recovery begins to be rolled back, but before the process is completed, DB2 continues to back out the changes during restart.

in-commit
A status of a unit of recovery. If DB2 fails after beginning its phase 2 commit processing, it "knows," when restarted, that changes made to data are consistent. Such units of recovery are termed *in-commit*.

independent
An object (row, table, or table space) that is neither a parent nor a dependent of another object.

index A set of pointers that are logically ordered by the values of a key. Indexes can provide faster access to data and can enforce uniqueness on the rows in a table.

index-controlled partitioning
A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are specified on the CREATE INDEX statement. Partition limits are saved in the LIMITKEY column of the SYSIBM.SYSINDEXPART catalog table.

index key
The set of columns in a table that is used to determine the order of index entries.

index partition
A VSAM data set that is contained within a partitioning index space.

index space
A page set that is used to store the entries of one index.

indicator column
A 4-byte value that is stored in a base table in place of a LOB column.

indicator variable
A variable that is used to represent the null value in an application program. If the value for the selected column is null, a negative value is placed in the indicator variable.

indoubt
A status of a unit of recovery. If DB2 fails after it has finished its phase 1 commit processing and before it has started phase 2, only the commit coordinator knows if an individual unit of recovery is to be committed or rolled back. At restart, if DB2 lacks the information it needs to make this decision, the status of the unit of recovery is *indoubt* until DB2 obtains this information from the coordinator. More than one unit of recovery can be indoubt at restart.

indoubt resolution
The process of resolving the status of an indoubt logical unit of work to either the committed or the rollback state.

inflight
A status of a unit of recovery. If DB2 fails before its unit of recovery completes phase 1 of the commit process, it merely backs out the updates of its unit of recovery at restart. These units of recovery are termed *inflight*.

inheritance
The passing downstream of class resources or attributes from a parent class in the class hierarchy to a child class.

initialization file
For DB2 ODBC applications, a file containing values that can be set to adjust the performance of the database manager.

inline copy
A copy that is produced by the LOAD or REORG utility. The data set that the inline copy produces is logically equivalent to a full image copy that is produced by running the COPY utility with read-only access (SHRLEVEL REFERENCE).

inline SQL PL
A subset of SQL procedural language that can be used in SQL functions and dynamic compound statements. See also SQL procedural language.

inner join
The result of a join operation that includes only the matched rows of both tables that are being joined. See also join,

equijoin, full outer join, left outer join, outer join, and right outer join.

inoperative package
In DB2 Version 9.1 for z/OS and earlier, a package that cannot be used because one or more user-defined functions or procedures that the package depends on were dropped. Such a package must be explicitly rebound. Contrast with invalid package.

insensitive cursor
A cursor that is not sensitive to inserts, updates, or deletes that are made to the underlying rows of a result table after the result table has been materialized.

insert trigger
A trigger that is defined with the triggering SQL operation, an insert.

install The process of preparing a DB2 subsystem to operate as a z/OS subsystem.

INSTEAD OF trigger
A trigger that is associated with a single view and is activated by an insert, update, or delete operation on the view and that can define how to propagate the insert, update, or delete operation on the view to the underlying tables of the view. Contrast with BEFORE trigger and AFTER trigger.

instrumentation facility component identifier (IFCID)
A value that names and identifies a trace record of an event that can be traced. As a parameter on the START TRACE and MODIFY TRACE commands, it specifies that the corresponding event is to be traced.

instrumentation facility interface (IFI)
A programming interface that enables programs to obtain online trace data about DB2, to submit DB2 commands, and to pass data to DB2.

Interactive System Productivity Facility (ISPF)
An IBM licensed program that provides interactive dialog services in a z/OS environment.

inter-DB2 R/W interest
A property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members.

intermediate database server
The target of a request from a local application or a remote application requester that is forwarded to another database server.

internal resource lock manager (IRLM)
A z/OS subsystem that DB2 uses to control communication and database locking.

intersection
An SQL operation that involves the INTERSECT set operator, which combines two result tables. The result of an intersection operation consists of all of the rows that are in both result tables.

invalid package
In DB2 Version 9.1 for z/OS and earlier, a package that depends on an object (other than a user-defined function) that is dropped. Such a package is implicitly rebound on invocation. Contrast with inoperative package.

IP address
A value that uniquely identifies a TCP/IP host.

IRLM See internal resource lock manager.

isolation level
The degree to which a unit of work is isolated from the updating operations of other units of work. See also cursor stability, read stability, repeatable read, and uncommitted read.

ISPF See Interactive System Productivity Facility.

iterator
In SQLJ, an object that contains the result set of a query. An iterator is equivalent to a cursor in other host languages.

iterator declaration clause
In SQLJ, a statement that generates an iterator declaration class. An iterator is an object of an iterator declaration class.

JAR See Java Archive.

Java Archive (JAR)
A file format that is used for aggregating many files into a single file.

JDBC A Sun Microsystems database application

programming interface (API) for Java that allows programs to access database management systems by using callable SQL.

join A relational operation that allows retrieval of data from two or more tables based on matching column values. See also equijoin, full outer join, inner join, left outer join, outer join, and right outer join.

KB Kilobyte. A value of 1024 bytes.

Kerberos
A network authentication protocol that is designed to provide strong authentication for client/server applications by using secret-key cryptography.

Kerberos ticket
A transparent application mechanism that transmits the identity of an initiating principal to its target. A simple ticket contains the principal's identity, a session key, a timestamp, and other information, which is sealed using the target's secret key.

key A column, an ordered collection of columns, or an expression that is identified in the description of a table, index, or referential constraint. The same column or expression can be part of more than one key.

key-sequenced data set (KSDS)
A VSAM file or data set whose records are loaded in key sequence and controlled by an index.

KSDS See key-sequenced data set.

large object (LOB)
A sequence of bytes representing bit data, single-byte characters, double-byte characters, or a mixture of single- and double-byte characters. A LOB can be up to 2 GB minus 1 byte in length. See also binary large object, character large object, and double-byte character large object.

last agent optimization
An optimized commit flow for either presumed-nothing or presumed-abort protocols in which the last agent, or final participant, becomes the commit coordinator. This flow saves at least one message.

latch A DB2 mechanism for controlling concurrent events or the use of system resources.

LCID See log control interval definition.

LDS See linear data set.

leaf page
An index page that contains pairs of keys and RIDs and that points to actual data. Contrast with nonleaf page.

left outer join
The result of a join operation that includes the matched rows of both tables that are being joined, and that preserves the unmatched rows of the first table. See also join, equijoin, full outer join, inner join, outer join, and right outer join.

limit key
The highest value of the index key for a partition.

linear data set (LDS)
A VSAM data set that contains data but no control information. A linear data set can be accessed as a byte-addressable string in virtual storage.

linkage editor
A computer program for creating load modules from one or more object modules or load modules by resolving cross references among the modules and, if necessary, adjusting addresses.

link-edit
The action of creating a loadable computer program using a linkage editor.

list A type of object, which DB2 utilities can process, that identifies multiple table spaces, multiple index spaces, or both. A list is defined with the LISTDEF utility control statement.

list structure
A coupling facility structure that lets data be shared and manipulated as elements of a queue.

L-lock See logical lock.

load module
A program unit that is suitable for loading into main storage for execution. The output of a linkage editor.

LOB See large object.

LOB locator
A mechanism that allows an application program to manipulate a large object value in the database system. A LOB locator is a fullword integer value that represents a single LOB value. An application program retrieves a LOB locator into a host variable and can then apply SQL operations to the associated LOB value using the locator.

LOB lock
A lock on a LOB value.

LOB table space
A table space that contains all the data for a particular LOB column in the related base table.

local A way of referring to any object that the local DB2 subsystem maintains. A *local table*, for example, is a table that is maintained by the local DB2 subsystem. Contrast with remote.

locale The definition of a subset of a user's environment that combines a CCSID and characters that are defined for a specific language and country.

local lock
A lock that provides intra-DB2 concurrency control, but not inter-DB2 concurrency control; that is, its scope is a single DB2.

local subsystem
The unique relational DBMS to which the user or application program is directly connected (in the case of DB2, by one of the DB2 attachment facilities).

location
The unique name of a database server. An application uses the location name to access a DB2 database server. A database alias can be used to override the location name when accessing a remote server.

location alias
Another name by which a database server identifies itself in the network. Applications can use this name to access a DB2 database server.

lock A means of controlling concurrent events or access to data. DB2 locking is performed by the IRLM.

lock duration
The interval over which a DB2 lock is held.

lock escalation
The promotion of a lock from a row, page, or LOB lock to a table space lock because the number of page locks that are concurrently held on a given resource exceeds a preset limit.

locking
The process by which the integrity of data is ensured. Locking prevents concurrent users from accessing inconsistent data. See also claim, drain, and latch.

lock mode
A representation for the type of access that concurrently running programs can have to a resource that a DB2 lock is holding.

lock object
The resource that is controlled by a DB2 lock.

lock promotion
The process of changing the size or mode of a DB2 lock to a higher, more restrictive level.

lock size
The amount of data that is controlled by a DB2 lock on table data; the value can be a row, a page, a LOB, a partition, a table, or a table space.

lock structure
A coupling facility data structure that is composed of a series of lock entries to support shared and exclusive locking for logical resources.

log A collection of records that describe the events that occur during DB2 execution and that indicate their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

log control interval definition
A suffix of the physical log record that tells how record segments are placed in the physical control interval.

logical claim
A claim on a logical partition of a nonpartitioning index.

logical index partition
The set of all keys that reference the same data partition.

logical lock (L-lock)
The lock type that transactions use to control intra- and inter-DB2 data concurrency between transactions. Contrast with physical lock (P-lock).

logically complete
A state in which the concurrent copy process is finished with the initialization of the target objects that are being copied. The target objects are available for update.

logical page list (LPL)
A list of pages that are in error and that cannot be referenced by applications until the pages are recovered. The page is in *logical error* because the actual media (coupling facility or disk) might not contain any errors. Usually a connection to the media has been lost.

logical partition
A set of key or RID pairs in a nonpartitioning index that are associated with a particular partition.

logical recovery pending (LRECP)
The state in which the data and the index keys that reference the data are inconsistent.

logical unit (LU)
An access point through which an application program accesses the SNA network in order to communicate with another application program. See also LU name.

logical unit of work
The processing that a program performs between synchronization points.

logical unit of work identifier (LUWID)
A name that uniquely identifies a thread within a network. This name consists of a fully-qualified LU network name, an LUW instance number, and an LUW sequence number.

log initialization
The first phase of restart processing during which DB2 attempts to locate the current end of the log.

log record header (LRH)
A prefix, in every log record, that contains control information.

log record sequence number (LRSN)
An identifier for a log record that is associated with a data sharing member. DB2 uses the LRSN for recovery in the data sharing environment.

log truncation
A process by which an explicit starting RBA is established. This RBA is the point at which the next byte of log data is to be written.

LPL See logical page list.

LRECP
See logical recovery pending.

LRH See log record header.

LRSN See log record sequence number.

LU See logical unit.

LU name
Logical unit name, which is the name by which VTAM refers to a node in a network.

LUW See logical unit of work.

LUWID
See logical unit of work identifier.

mapping table
A table that the REORG utility uses to map the associations of the RIDs of data records in the original copy and in the shadow copy. This table is created by the user.

mass delete
The deletion of all rows of a table.

materialize

- The process of putting rows from a view or nested table expression into a work file for additional processing by a query.
- The placement of a LOB value into contiguous storage. Because LOB values can be very large, DB2 avoids materializing LOB data until doing so becomes absolutely necessary.

materialized query table
A table that is used to contain information

that is derived and can be summarized from one or more source tables. Contrast with base table.

MB Megabyte (1 048 576 bytes).

MBCS See multibyte character set.

member name
The z/OS XCF identifier for a particular DB2 subsystem in a data sharing group.

menu A displayed list of available functions for selection by the operator. A menu is sometimes called a *menu panel*.

metalanguage
A language that is used to create other specialized languages.

migration
The process of converting a subsystem with a previous release of DB2 to an updated or current release. In this process, you can acquire the functions of the updated or current release without losing the data that you created on the previous release.

mixed data string
A character string that can contain both single-byte and double-byte characters.

mode name
A VTAM name for the collection of physical and logical characteristics and attributes of a session.

modify locks
An L-lock or P-lock with a MODIFY attribute. A list of these active locks is kept at all times in the coupling facility lock structure. If the requesting DB2 subsystem fails, that DB2 subsystem's modify locks are converted to retained locks.

multibyte character set (MBCS)
A character set that represents single characters with more than a single byte. UTF-8 is an example of an MBCS. Characters in UTF-8 can range from 1 to 4 bytes in DB2. Contrast with single-byte character set and double-byte character set. See also Unicode.

multidimensional analysis
The process of assessing and evaluating an enterprise on more than one level.

Multiple Virtual Storage (MVS)
An element of the z/OS operating system. This element is also called the Base Control Program (BCP).

multisite update
Distributed relational database processing in which data is updated in more than one location within a single unit of work.

multithreading
Multiple TCBs that are executing one copy of DB2 ODBC code concurrently (sharing a processor) or in parallel (on separate central processors).

MVS See Multiple Virtual Storage.

native SQL procedure
An SQL procedure that is processed by converting the procedural statements to a native representation that is stored in the database directory, as is done with other SQL statements. When a native SQL procedure is called, the native representation is loaded from the directory, and DB2 executes the procedure. Contrast with external procedure and external SQL procedure.

nested table expression
A fullselect in a FROM clause (surrounded by parentheses).

network identifier (NID)
The network ID that is assigned by IMS or CICS, or if the connection type is RRSAF, the RRS unit of recovery ID (URID).

new-function mode (NFM)
The normal mode of operation that exists after successful completion of a version-to-version migration. At this stage, all new functions of the new version are available for use. A DB2 data sharing group cannot coexist with members that are still at the prior version level, and fallback to a prior version is not supported. Contrast with conversion mode, conversion mode*, enabling-new-function mode, and enabling-new-function mode*.

NFM See new-function mode.

NID See network identifier.

node ID index
See XML node ID index.

nondeterministic function
A user-defined function whose result is

not solely dependent on the values of the input arguments. That is, successive invocations with the same argument values can produce a different answer. This type of function is sometimes called a *variant* function. Contrast with deterministic function (sometimes called a *not-variant function*).

nonleaf page
A page that contains keys and page numbers of other pages in the index (either leaf or nonleaf pages). Nonleaf pages never point to actual data. Contrast with leaf page.

nonpartitioned index
An index that is not physically partitioned. Both partitioning indexes and secondary indexes can be nonpartitioned.

nonpartitioned secondary index (NPSI)
An index on a partitioned table space that is not the partitioning index and is not partitioned. Contrast with data-partitioned secondary index.

nonpartitioning index
See secondary index.

nonscrollable cursor
A cursor that can be moved only in a forward direction. Nonscrollable cursors are sometimes called forward-only cursors or serial cursors.

normalization
A key step in the task of building a logical relational database design. Normalization helps you avoid redundancies and inconsistencies in your data. An entity is normalized if it meets a set of constraints for a particular normal form (first normal form, second normal form, and so on). Contrast with denormalization.

not-variant function
See deterministic function.

NPSI See nonpartitioned secondary index.

NUL The null character ('\0'), which is represented by the value X'00'. In C, this character denotes the end of a string.

null A special value that indicates the absence of information.

null terminator
In C, the value that indicates the end of a string. For EBCDIC, ASCII, and Unicode UTF-8 strings, the null terminator is a single-byte value (X'00'). For Unicode UTF-16 or UCS-2 (wide) strings, the null terminator is a double-byte value (X'0000').

ODBC
See Open Database Connectivity.

ODBC driver
A dynamically-linked library (DLL) that implements ODBC function calls and interacts with a data source.

OLAP See online analytical processing.

online analytical processing (OLAP)
The process of collecting data from one or many sources; transforming and analyzing the consolidated data quickly and interactively; and examining the results across different dimensions of the data by looking for patterns, trends, and exceptions within complex relationships of that data.

Open Database Connectivity (ODBC)
A Microsoft database application programming interface (API) for C that allows access to database management systems by using callable SQL. ODBC does not require the use of an SQL preprocessor. In addition, ODBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time. This means that applications no longer need to be directly linked to the modules of all the database management systems that are supported.

ordinary identifier
An uppercase letter followed by zero or more characters, each of which is an uppercase letter, a digit, or the underscore character. An ordinary identifier must not be a reserved word.

ordinary token
A numeric constant, an ordinary identifier, a host identifier, or a keyword.

originating task
In a parallel group, the primary agent that receives data from other execution units (referred to as *parallel tasks*) that are executing portions of the query in parallel.

outer join
The result of a join operation that includes the matched rows of both tables that are being joined and preserves some or all of the unmatched rows of the tables that are being joined. See also join, equijoin, full outer join, inner join, left outer join, and right outer join.

overloaded function
A function name for which multiple function instances exist.

package
An object containing a set of SQL statements that have been statically bound and that is available for processing. A package is sometimes also called an *application package*.

package list
An ordered list of package names that may be used to extend an application plan.

package name
The name of an object that is used for an application package or an SQL procedure package. An application package is a bound version of a database request module (DBRM) that is created by a BIND PACKAGE or REBIND PACKAGE command. An SQL procedural language package is created by a CREATE or ALTER PROCEDURE statement for a native SQL procedure. The name of a package consists of a location name, a collection ID, a package ID, and a version ID.

page A unit of storage within a table space (4 KB, 8 KB, 16 KB, or 32 KB) or index space (4 KB, 8 KB, 16 KB, or 32 KB). In a table space, a page contains one or more rows of a table. In a LOB or XML table space, a LOB or XML value can span more than one page, but no more than one LOB or XML value is stored on a page.

page set
Another way to refer to a table space or index space. Each page set consists of a collection of VSAM data sets.

page set recovery pending (PSRCP)
A restrictive state of an index space. In this case, the entire page set must be recovered. Recovery of a logical part is prohibited.

panel A predefined display image that defines the locations and characteristics of display fields on a display surface (for example, a *menu panel*).

parallel complex
A cluster of machines that work together to handle multiple transactions and applications.

parallel group
A set of consecutive operations that execute in parallel and that have the same number of parallel tasks.

parallel I/O processing
A form of I/O processing in which DB2 initiates multiple concurrent requests for a single user query and performs I/O processing concurrently (in *parallel*) on multiple data partitions.

parallelism assistant
In Sysplex query parallelism, a DB2 subsystem that helps to process parts of a parallel query that originates on another DB2 subsystem in the data sharing group.

parallelism coordinator
In Sysplex query parallelism, the DB2 subsystem from which the parallel query originates.

Parallel Sysplex
A set of z/OS systems that communicate and cooperate with each other through certain multisystem hardware components and software services to process customer workloads.

parallel task
The execution unit that is dynamically created to process a query in parallel. A parallel task is implemented by a z/OS service request block.

parameter marker
A question mark (?) that appears in a statement string of a dynamic SQL statement. The question mark can appear where a variable could appear if the statement string were a static SQL statement.

parameter-name
An SQL identifier that designates a parameter in a routine that is written by a user. Parameter names are required for SQL procedures and SQL functions, and they are used in the body of the routine

to refer to the values of the parameters. Parameter names are optional for external routines.

parent key
A primary key or unique key in the parent table of a referential constraint. The values of a parent key determine the valid values of the foreign key in the referential constraint.

parent lock
For explicit hierarchical locking, a lock that is held on a resource that might have child locks that are lower in the hierarchy. A parent lock is usually the table space lock or the partition intent lock. See also child lock.

parent row
A row whose primary key value is the foreign key value of a dependent row.

parent table
A table whose primary key is referenced by the foreign key of a dependent table.

parent table space
A table space that contains a parent table. A table space containing a dependent of that table is a dependent table space.

participant
An entity other than the commit coordinator that takes part in the commit process. The term participant is synonymous with agent in SNA.

partition
A portion of a page set. Each partition corresponds to a single, independently extendable data set. The maximum size of a partition depends on the number of partitions in the partitioned page set. All partitions of a given page set have the same maximum size.

partition-by-growth table space
A table space whose size can grow to accommodate data growth. DB2 for z/OS manages partition-by-growth table spaces by automatically adding new data sets when the database needs more space to satisfy an insert operation. Contrast with range-partitioned table space. See also universal table space.

partitioned data set (PDS)
A data set in disk storage that is divided into partitions, which are called members. Each partition can contain a program, part of a program, or data. A program library is an example of a partitioned data set.

partitioned index
An index that is physically partitioned. Both partitioning indexes and secondary indexes can be partitioned.

partitioned page set
A partitioned table space or an index space. Header pages, space map pages, data pages, and index pages reference data only within the scope of the partition.

partitioned table space
A table space that is based on a single table and that is subdivided into partitions, each of which can be processed independently by utilities. Contrast with segmented table space and universal table space.

partitioning index
An index in which the leftmost columns are the partitioning columns of the table. The index can be partitioned or nonpartitioned.

partner logical unit
An access point in the SNA network that is connected to the local DB2 subsystem by way of a VTAM conversation.

path See SQL path.

PDS See partitioned data set.

period In a table, an interval of time that is defined by two datetime columns. A period contains a begin column and an end column. See also begin column and end column.

physical consistency
The state of a page that is not in a partially changed state.

physical lock (P-lock)
A type of lock that DB2 acquires to provide consistency of data that is cached in different DB2 subsystems. Physical locks are used only in data sharing environments. Contrast with logical lock (L-lock).

physically complete
The state in which the concurrent copy process is completed and the output data set has been created.

piece A data set of a nonpartitioned page set.

plan See application plan.

plan allocation
The process of allocating DB2 resources to a plan in preparation for execution.

plan member
The bound copy of a DBRM that is identified in the member clause.

plan name
The name of an application plan.

P-lock See physical lock.

point of consistency
A time when all recoverable data that an application accesses is consistent with other data. The term point of consistency is synonymous with sync point or commit point.

policy See CFRM policy.

postponed abort UR
A unit of recovery that was inflight or in-abort, was interrupted by system failure or cancellation, and did not complete backout during restart.

precision
In SQL, the total number of digits in a decimal number (called the *size* in the C language). In the C language, the number of digits to the right of the decimal point (called the scale in SQL). The DB2 information uses the SQL terms.

precompilation
A processing of application programs containing SQL statements that takes place before compilation. SQL statements are replaced with statements that are recognized by the host language compiler. Output from this precompilation includes source code that can be submitted to the compiler and the database request module (DBRM) that is input to the bind process.

predicate
An element of a search condition that expresses or implies a comparison operation.

prefix A code at the beginning of a message or record.

preformat
The process of preparing a VSAM linear data set for DB2 use, by writing specific data patterns.

prepare
The first phase of a two-phase commit process in which all participants are requested to prepare for commit.

prepared SQL statement
A named object that is the executable form of an SQL statement that has been processed by the PREPARE statement.

primary authorization ID
The authorization ID that is used to identify the application process to DB2.

primary group buffer pool
For a duplexed group buffer pool, the structure that is used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The z/OS equivalent is *old* structure. Compare with secondary group buffer pool.

primary index
An index that enforces the uniqueness of a primary key.

primary key
In a relational database, a unique, nonnull key that is part of the definition of a table. A table cannot be defined as a parent unless it has a unique key or primary key.

principal
An entity that can communicate securely with another entity. In Kerberos, principals are represented as entries in the Kerberos registry database and include users, servers, computers, and others.

principal name
The name by which a principal is known to the DCE security services.

privilege
The capability of performing a specific function, sometimes on a specific object. See also explicit privilege.

privilege set
- For the installation SYSADM ID, the set of all possible privileges.

- For any other authorization ID, including the PUBLIC authorization ID, the set of all privileges that are recorded for that ID in the DB2 catalog.

process
In DB2, the unit to which DB2 allocates resources and locks. Sometimes called an application process, a process involves the execution of one or more programs. The execution of an SQL statement is always associated with some process. The means of initiating and terminating a process are dependent on the environment.

program
A single, compilable collection of executable statements in a programming language.

program temporary fix (PTF)
A solution or bypass of a problem that is diagnosed as a result of a defect in a current unaltered release of a licensed program. An authorized program analysis report (APAR) fix is corrective service for an existing problem. A PTF is preventive service for problems that might be encountered by other users of the product. A PTF is *temporary*, because a permanent fix is usually not incorporated into the product until its next release.

protected conversation
A VTAM conversation that supports two-phase commit flows.

PSRCP
See page set recovery pending.

PTF See program temporary fix.

QSAM
See queued sequential access method.

query A component of certain SQL statements that specifies a result table.

query block
The part of a query that is represented by one of the FROM clauses. Each FROM clause can have multiple query blocks, depending on DB2 processing of the query.

query CP parallelism
Parallel execution of a single query, which is accomplished by using multiple tasks. See also Sysplex query parallelism.

query I/O parallelism
Parallel access of data, which is accomplished by triggering multiple I/O requests within a single query.

queued sequential access method (QSAM)
An extended version of the basic sequential access method (BSAM). When this method is used, a queue of data blocks is formed. Input data blocks await processing, and output data blocks await transfer to auxiliary storage or to an output device.

quiesce point
A point at which data is consistent as a result of running the DB2 QUIESCE utility.

RACF Resource Access Control Facility. A component of the z/OS Security Server.

range-partitioned table space
A type of universal table space that is based on partitioning ranges and that contains a single table. Contrast with partition-by-growth table space. See also universal table space.

RBA See relative byte address.

RCT See resource control table.

RDO See resource definition online.

read stability (RS)
An isolation level that is similar to repeatable read but does not completely isolate an application process from all other concurrently executing application processes. See also cursor stabilityrepeatable read, and uncommitted read.

rebind
The creation of a new application plan for an application program that has been bound previously. If, for example, you have added an index for a table that your application accesses, you must rebind the application in order to take advantage of that index.

rebuild
The process of reallocating a coupling facility structure. For the shared communications area (SCA) and lock structure, the structure is repopulated; for the group buffer pool, changed pages are usually cast out to disk, and the new

structure is populated only with changed pages that were not successfully cast out.

record The storage representation of a row or other data.

record identifier (RID)
A unique identifier that DB2 uses to identify a row of data in a table. Compare with row identifier.

record identifier (RID) pool
An area of main storage that is used for sorting record identifiers during list-prefetch processing.

record length
The sum of the length of all the columns in a table, which is the length of the data as it is physically stored in the database. Records can be fixed length or varying length, depending on how the columns are defined. If all columns are fixed-length columns, the record is a fixed-length record. If one or more columns are varying-length columns, the record is a varying-length record.

Resource Recovery Services attachment facility (RRSAF)
A DB2 subcomponent that uses Resource Recovery Services to coordinate resource commitment between DB2 and all other resource managers that also use RRS in a z/OS system.

recovery
The process of rebuilding databases after a system failure.

recovery log
A collection of records that describes the events that occur during DB2 execution and indicates their sequence. The recorded information is used for recovery in the event of a failure during DB2 execution.

recovery manager
A subcomponent that supplies coordination services that control the interaction of DB2 resource managers during commit, abort, checkpoint, and restart processes. The recovery manager also supports the recovery mechanisms of other subsystems (for example, IMS) by acting as a participant in the other subsystem's process for protecting data that has reached a point of consistency.

A coordinator or a participant (or both), in the execution of a two-phase commit, that can access a recovery log that maintains the state of the logical unit of work and names the immediate upstream coordinator and downstream participants.

recovery pending (RECP)
A condition that prevents SQL access to a table space that needs to be recovered.

recovery token
An identifier for an element that is used in recovery (for example, NID or URID).

RECP See recovery pending.

redo A state of a unit of recovery that indicates that changes are to be reapplied to the disk media to ensure data integrity.

reentrant code
Executable code that can reside in storage as one shared copy for all threads. Reentrant code is not self-modifying and provides separate storage areas for each thread. See also threadsafe.

referential constraint
The requirement that nonnull values of a designated foreign key are valid only if they equal values of the primary key of a designated table.

referential cycle
A set of referential constraints such that each base table in the set is a descendent of itself. The tables that are involved in a referential cycle are ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table.

referential integrity
The state of a database in which all values of all foreign keys are valid. Maintaining referential integrity requires the enforcement of referential constraints on all operations that change the data in a table on which the referential constraints are defined.

referential structure
A set of tables and relationships that includes at least one table and, for every table in the set, all the relationships in which that table participates and all the tables to which it is related.

refresh age
The time duration between the current

time and the time during which a materialized query table was last refreshed.

registry
See registry database.

registry database
A database of security information about principals, groups, organizations, accounts, and security policies.

relational database
A database that can be perceived as a set of tables and manipulated in accordance with the relational model of data.

relational database management system (RDBMS)
A collection of hardware and software that organizes and provides access to a relational database.

relational schema
See SQL schema.

relationship
A defined connection between the rows of a table or the rows of two tables. A relationship is the internal representation of a referential constraint.

relative byte address (RBA)
The offset of a data record or control interval from the beginning of the storage space that is allocated to the data set or file to which it belongs.

remigration
The process of returning to a current release of DB2 following a fallback to a previous release. This procedure constitutes another migration process.

remote
Any object that is maintained by a remote DB2 subsystem (that is, by a DB2 subsystem other than the local one). A *remote view*, for example, is a view that is maintained by a remote DB2 subsystem. Contrast with local.

remote subsystem
Any relational DBMS, except the local subsystem, with which the user or application can communicate. The subsystem need not be remote in any physical sense, and might even operate on the same processor under the same z/OS system.

reoptimization
The DB2 process of reconsidering the access path of an SQL statement at run time; during reoptimization, DB2 uses the values of host variables, parameter markers, or special registers.

reordered row format
A row format that facilitates improved performance in retrieval of rows that have varying-length columns. DB2 rearranges the column order, as defined in the CREATE TABLE statement, so that the fixed-length columns are stored at the beginning of the row and the varying-length columns are stored at the end of the row. Contrast with basic row format.

REORG pending (REORP)
A condition that restricts SQL access and most utility access to an object that must be reorganized.

REORP
See REORG pending.

repeatable read (RR)
The isolation level that provides maximum protection from other executing application programs. When an application program executes with repeatable read protection, rows that the program references cannot be changed by other programs until the program reaches a commit point. See also cursor stability, read stability, and uncommitted read.

repeating group
A situation in which an entity includes multiple attributes that are inherently the same. The presence of a repeating group violates the requirement of first normal form. In an entity that satisfies the requirement of first normal form, each attribute is independent and unique in its meaning and its name. See also normalization.

replay detection mechanism
A method that allows a principal to detect whether a request is a valid request from a source that can be trusted or whether an untrustworthy entity has captured information from a previous exchange and is replaying the information exchange to gain access to the principal.

request commit
The vote that is submitted to the prepare phase if the participant has modified data and is prepared to commit or roll back.

requester
The source of a request to access data at a remote server. In the DB2 environment, the requester function is provided by the distributed data facility.

resource
The object of a lock or claim, which could be a table space, an index space, a data partition, an index partition, or a logical partition.

resource allocation
The part of plan allocation that deals specifically with the database resources.

resource control table
A construct of previous versions of the CICS attachment facility that defines authorization and access attributes for transactions or transaction groups. Beginning in CICS Transaction Server Version 1.3, resources are defined by using resource definition online instead of the resource control table. See also resource definition online.

resource definition online (RDO)
The recommended method of defining resources to CICS by creating resource definitions interactively, or by using a utility, and then storing them in the CICS definition data set. In earlier releases of CICS, resources were defined by using the resource control table (RCT), which is no longer supported.

resource limit facility (RLF)
A portion of DB2 code that prevents dynamic manipulative SQL statements from exceeding specified time limits. The resource limit facility is sometimes called the governor.

resource limit specification table (RLST)
A site-defined table that specifies the limits to be enforced by the resource limit facility.

resource manager

- A function that is responsible for managing a particular resource and that guarantees the consistency of all updates made to recoverable resources within a logical unit of work. The resource that is being managed can be physical (for example, disk or main storage) or logical (for example, a particular type of system service).
- A participant, in the execution of a two-phase commit, that has recoverable resources that could have been modified. The resource manager has access to a recovery log so that it can commit or roll back the effects of the logical unit of work to the recoverable resources.

restart pending (RESTP)
A restrictive state of a page set or partition that indicates that restart (backout) work needs to be performed on the object.

RESTP
See restart pending.

result set
The set of rows that a stored procedure returns to a client application.

result set locator
A 4-byte value that DB2 uses to uniquely identify a query result set that a stored procedure returns.

result table
The set of rows that are specified by a SELECT statement.

retained lock
A MODIFY lock that a DB2 subsystem was holding at the time of a subsystem failure. The lock is retained in the coupling facility lock structure across a DB2 for z/OS failure.

RID See record identifier.

RID pool
See record identifier pool.

right outer join
The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of the second join operand. See also join, equijoin, full outer join, inner join, left outer join, and outer join.

RLF See resource limit facility.

RLST See resource limit specification table.

role A database entity that groups together one or more privileges and that can be

assigned to a primary authorization ID or to PUBLIC. The role is available only in a trusted context.

rollback
The process of restoring data that was changed by SQL statements to the state at its last commit point. All locks are freed. Contrast with commit.

root page
The index page that is at the highest level (or the beginning point) in an index.

routine
A database object that encapsulates procedural logic and SQL statements, is stored on the database server, and can be invoked from an SQL statement or by using the CALL statement. The main classes of routines are procedures and functions.

row The horizontal component of a table. A row consists of a sequence of values, one for each column of the table.

row-begin column
A generated column that is defined with the AS ROW BEGIN clause. The value is assigned whenever a row is inserted into the table or any column in the row is updated. A row-begin column is intended for use as the first column of a SYSTEM_TIME period. See also generated column, row-end column, and transaction-start-ID column.

row change timestamp column
A generated column that is defined with the AS ROW CHANGE TIMESTAMP clause. A row change timestamp column provides a way for the database manager to automatically generate and maintain a timestamp value for each row that is inserted or updated in a table. A table can have no more than one row change timestamp column.

row-end column
A generated column that is defined with the AS ROW END clause. The value is assigned whenever a row is inserted into the table or any column in the row is updated. A row-end column is intended for use as the second column of a SYSTEM_TIME period. See also generated column, row-begin column, and transaction-start-ID column.

rowid A value that uniquely identifies a row in a table and does not change.

row lock
A lock on a single row of data.

row-positioned fetch orientation
The specification of the desired placement of the cursor as part of a FETCH statement, with respect to a single row (for example, NEXT, LAST, or ABSOLUTE *n*). Contrast with rowset-positioned fetch orientation.

rowset
A set of rows for which a cursor position is established.

rowset cursor
A cursor that is defined so that one or more rows can be returned as a rowset for a single FETCH statement, and the cursor is positioned on the set of rows that is fetched.

rowset-positioned fetch orientation
The specification of the desired placement of the cursor as part of a FETCH statement, with respect to a rowset (for example, NEXT ROWSET, LAST ROWSET, or ROWSET STARTING AT ABSOLUTE *n*). Contrast with row-positioned fetch orientation.

row trigger
A trigger that is defined with the trigger granularity FOR EACH ROW.

RRSAF
See Resource Recovery Services attachment facility.

RS See read stability.

savepoint
A named entity that represents the state of data and schemas at a particular point in time within a unit of work.

SBCS See single-byte character set.

SCA See shared communications area.

scalar function
An SQL operation that produces a single value from another value and is expressed as a function name, followed by a list of arguments that are enclosed in parentheses.

scale In SQL, the number of digits to the right of the decimal point (called the precision

in the C language). The DB2 information uses the SQL definition.

schema
The organization or structure of a database.

A collection of, and a way of qualifying, database objects such as tables, views, routines, indexes or triggers that define a database. A database schema provides a logical classification of database objects.

scrollability
The ability to use a cursor to fetch in either a forward or backward direction. The FETCH statement supports multiple fetch orientations to indicate the new position of the cursor. See also fetch orientation.

scrollable cursor
A cursor that can be moved in both a forward and a backward direction.

search condition
A criterion for selecting rows from a table. A search condition consists of one or more predicates.

secondary authorization ID
An authorization ID that has been associated with a primary authorization ID by an authorization exit routine.

secondary group buffer pool
For a duplexed group buffer pool, the structure that is used to back up changed pages that are written to the primary group buffer pool. No page registration or cross-invalidation occurs using the secondary group buffer pool. The z/OS equivalent is *new* structure.

secondary index
A nonpartitioning index that is useful for enforcing a uniqueness constraint, for clustering data, or for providing access paths to data for queries. A secondary index can be partitioned or nonpartitioned. See also data-partitioned secondary index (DPSI) and nonpartitioned secondary index (NPSI).

section
The segment of a plan or package that contains the executable structures for a single SQL statement. For most SQL statements, one section in the plan exists for each SQL statement in the source program. However, for cursor-related statements, the DECLARE, OPEN, FETCH, and CLOSE statements reference the same section because they each refer to the SELECT statement that is named in the DECLARE CURSOR statement. SQL statements such as COMMIT, ROLLBACK, and some SET statements do not use a section.

security label
A classification of users' access to objects or data rows in a multilevel security environment."

segment
A group of pages that holds rows of a single table. See also segmented table space.

segmented table space
A table space that is divided into equal-sized groups of pages called segments. Segments are assigned to tables so that rows of different tables are never stored in the same segment. Contrast with partitioned table space and universal table space.

self-referencing constraint
A referential constraint that defines a relationship in which a table is a dependent of itself.

self-referencing table
A table with a self-referencing constraint.

sensitive cursor
A cursor that is sensitive to changes that are made to the database after the result table has been materialized.

sequence
A user-defined object that generates a sequence of numeric values according to user specifications.

sequential data set
A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 database utilities require sequential data sets.

sequential prefetch
A mechanism that triggers consecutive asynchronous I/O operations. Pages are fetched before they are required, and several pages are read with a single I/O operation.

serialized profile
A Java object that contains SQL statements and descriptions of host variables. The SQLJ translator produces a serialized profile for each connection context.

server The target of a request from a remote requester. In the DB2 environment, the server function is provided by the distributed data facility, which is used to access DB2 data from remote applications.

service class
An eight-character identifier that is used by the z/OS Workload Manager to associate user performance goals with a particular DDF thread or stored procedure. A service class is also used to classify work on parallelism assistants.

service request block
A unit of work that is scheduled to execute.

session
A link between two nodes in a VTAM network.

session protocols
The available set of SNA communication requests and responses.

set operator
The SQL operators UNION, EXCEPT, and INTERSECT corresponding to the relational operators union, difference, and intersection. A set operator derives a result table by combining two other result tables.

shared communications area (SCA)
A coupling facility list structure that a DB2 data sharing group uses for inter-DB2 communication.

share lock
A lock that prevents concurrently executing application processes from changing data, but not from reading data. Contrast with exclusive lock.

shift-in character
A special control character (X'0F') that is used in EBCDIC systems to denote that the subsequent bytes represent SBCS characters. See also shift-out character.

shift-out character
A special control character (X'0E') that is used in EBCDIC systems to denote that the subsequent bytes, up to the next shift-in control character, represent DBCS characters. See also shift-in character.

sign-on
A request that is made on behalf of an individual CICS or IMS application process by an attachment facility to enable DB2 to verify that it is authorized to use DB2 resources.

simple page set
A nonpartitioned page set. A simple page set initially consists of a single data set (page set piece). If and when that data set is extended to 2 GB, another data set is created, and so on, up to a total of 32 data sets. DB2 considers the data sets to be a single contiguous linear address space containing a maximum of 64 GB. Data is stored in the next available location within this address space without regard to any partitioning scheme.

simple table space
A table space that is neither partitioned nor segmented. Creation of simple table spaces is not supported in DB2 Version 9.1 for z/OS. Contrast with partitioned table space, segmented table space, and universal table space.

single-byte character set (SBCS)
A set of characters in which each character is represented by a single byte. Contrast with double-byte character set or multibyte character set.

single-precision floating point number
A 32-bit approximate representation of a real number.

SMP/E
See System Modification Program/Extended.

SNA See Systems Network Architecture.

SNA network
The part of a network that conforms to the formats and protocols of Systems Network Architecture (SNA).

socket A callable TCP/IP programming interface that TCP/IP network applications use to communicate with remote TCP/IP partners.

sourced function
A function that is implemented by another built-in or user-defined function

that is already known to the database manager. This function can be a scalar function or an aggregate function; it returns a single value from a set of values (for example, MAX or AVG). Contrast with built-in function, external function, and SQL function.

source program
A set of host language statements and SQL statements that is processed by an SQL precompiler.

source table
A table that can be a base table, a view, a table expression, or a user-defined table function.

source type
An existing type that DB2 uses to represent a distinct type.

space A sequence of one or more blank characters.

special register
A storage area that DB2 defines for an application process to use for storing information that can be referenced in SQL statements. Examples of special registers are SESSION_USER and CURRENT DATE.

specific function name
A particular user-defined function that is known to the database manager by its specific name. Many specific user-defined functions can have the same function name. When a user-defined function is defined to the database, every function is assigned a specific name that is unique within its schema. Either the user can provide this name, or a default name is used.

SPUFI See SQL Processor Using File Input.

SQL See Structured Query Language.

SQL authorization ID (SQL ID)
The authorization ID that is used for checking dynamic SQL statements in some situations.

SQLCA
See SQL communication area.

SQL communication area (SQLCA)
A structure that is used to provide an application program with information about the execution of its SQL statements.

SQL connection
An association between an application process and a local or remote application server or database server.

SQLDA
See SQL descriptor area.

SQL descriptor area (SQLDA)
A structure that describes input variables, output variables, or the columns of a result table.

SQL escape character
The symbol that is used to enclose an SQL delimited identifier. This symbol is the double quotation mark ("). See also escape character.

SQL function
A user-defined function in which the CREATE FUNCTION statement contains the source code. The source code is a single SQL expression that evaluates to a single value. The SQL user-defined function can return the result of an expression. See also built-in function, external function, and sourced function.

SQL ID
See SQL authorization ID.

SQLJ Structured Query Language (SQL) that is embedded in the Java programming language.

SQL path
An ordered list of schema names that are used in the resolution of unqualified references to user-defined functions, distinct types, and stored procedures. In dynamic SQL, the SQL path is found in the CURRENT PATH special register. In static SQL, it is defined in the PATH bind option.

SQL PL
See SQL procedural language.

SQL procedural language (SQL PL)
A language extension of SQL that consists of statements and language elements that can be used to implement procedural logic in SQL statements. SQL PL provides statements for declaring variables and condition handlers, assigning values to variables, and for implementing procedural logic. See also inline SQL PL.

SQL procedure
A user-written program that can be

invoked with the SQL CALL statement. An SQL procedure is written in the SQL procedural language. Two types of SQL procedures are supported: external SQL procedures and native SQL procedures. See also external procedure and native SQL procedure.

SQL processing conversation
Any conversation that requires access of DB2 data, either through an application or by dynamic query requests.

SQL Processor Using File Input (SPUFI)
A facility of the TSO attachment subcomponent that enables the DB2I user to execute SQL statements without embedding them in an application program.

SQL return code
Either SQLCODE or SQLSTATE.

SQL routine
A user-defined function or stored procedure that is based on code that is written in SQL.

SQL schema
A collection of database objects such as tables, views, indexes, functions, distinct types, schemas, or triggers that defines a database. An SQL schema provides a logical classification of database objects.

SQL statement coprocessor
An alternative to the DB2 precompiler that lets the user process SQL statements at compile time. The user invokes an SQL statement coprocessor by specifying a compiler option.

SQL string delimiter
A symbol that is used to enclose an SQL string constant. The SQL string delimiter is the apostrophe ('), except in COBOL applications, where the user assigns the symbol, which is either an apostrophe or a double quotation mark (").

SRB See service request block.

stand-alone
An attribute of a program that means that it is capable of executing separately from DB2, without using DB2 services.

star join
A method of joining a dimension column of a fact table to the key column of the corresponding dimension table. See also join, dimension, and star schema.

star schema
The combination of a fact table (which contains most of the data) and a number of dimension tables. See also star join, dimension, and dimension table.

statement handle
In DB2 ODBC, the data object that contains information about an SQL statement that is managed by DB2 ODBC. This includes information such as dynamic arguments, bindings for dynamic arguments and columns, cursor information, result values, and status information. Each statement handle is associated with the connection handle.

statement string
For a dynamic SQL statement, the character string form of the statement.

statement trigger
A trigger that is defined with the trigger granularity FOR EACH STATEMENT.

static cursor
A named control structure that does not change the size of the result table or the order of its rows after an application opens the cursor. Contrast with dynamic cursor.

static SQL
SQL statements, embedded within a program, that are prepared during the program preparation process (before the program is executed). After being prepared, the SQL statement does not change (although values of variables that are specified by the statement might change).

storage group
A set of storage objects on which DB2 for z/OS data can be stored. A storage object can have an SMS data class, a management class, a storage class, and a list of volume serial numbers.

stored procedure
A user-written application program that can be invoked through the use of the SQL CALL statement. Stored procedures are sometimes called procedures.

string See binary string, character string, or graphic string.

strong typing
A process that guarantees that only user-defined functions and operations that are defined on a distinct type can be applied to that type. For example, you cannot directly compare two currency types, such as Canadian dollars and U.S. dollars. But you can provide a user-defined function to convert one currency to the other and then do the comparison.

structure

- A name that refers collectively to different types of DB2 objects, such as tables, databases, views, indexes, and table spaces.
- A construct that uses z/OS to map and manage storage on a coupling facility. See also cache structure, list structure, or lock structure.

Structured Query Language (SQL)
A standardized language for defining and manipulating data in a relational database.

structure owner
In relation to group buffer pools, the DB2 member that is responsible for the following activities:

- Coordinating rebuild, checkpoint, and damage assessment processing
- Monitoring the group buffer pool threshold and notifying castout owners when the threshold has been reached

subcomponent
A group of closely related DB2 modules that work together to provide a general function.

subject table
The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

subquery
A SELECT statement within the WHERE or HAVING clause of another SQL statement; a nested SQL statement.

subselect
That form of a query that includes only a SELECT clause, FROM clause, and optionally a WHERE clause, GROUP BY clause, HAVING clause, ORDER BY clause, or FETCH FIRST clause.

substitution character
A unique character that is substituted during character conversion for any characters in the source program that do not have a match in the target coding representation.

subsystem
In z/OS, a service provider that performs one or many functions, but does nothing until a request is made. For example, each WebSphere MQ for z/OS queue manager or instance of a DB2 for z/OS database management system is a z/OS subsystem.

surrogate pair
A coded representation for a single character that consists of a sequence of two 16-bit code units, in which the first value of the pair is a high-surrogate code unit in the range U+D800 through U+DBFF, and the second value is a low-surrogate code unit in the range U+DC00 through U+DFFF. Surrogate pairs provide an extension mechanism for encoding 917 476 characters without requiring the use of 32-bit characters.

SVC dump
A dump that is issued when a z/OS or a DB2 functional recovery routine detects an error.

sync point
See commit point.

syncpoint tree
The tree of recovery managers and resource managers that are involved in a logical unit of work, starting with the recovery manager, that make the final commit decision.

synonym
In SQL, an alternative name for a table or view. Synonyms can be used to refer only to objects at the subsystem in which the synonym is defined. A synonym cannot be qualified and can therefore not be used by other users. Contrast with alias.

Sysplex
See Parallel Sysplex.

Sysplex query parallelism
Parallel execution of a single query that is accomplished by using multiple tasks on more than one DB2 subsystem. See also query CP parallelism.

system administrator
The person at a computer installation who designs, controls, and manages the use of the computer system.

system agent
A work request that DB2 creates such as prefetch processing, deferred writes, and service tasks. See also allied agent.

system authorization ID
The primary DB2 authorization ID that is used to establish a trusted connection. A system authorization ID is derived from the system user ID that is provided by an external entity, such as a middleware server.

system conversation
The conversation that two DB2 subsystems must establish to process system messages before any distributed processing can begin.

system-defined routine
In DB2 10 for z/OS and later, an object (function or procedure) for which system DBADM and SQLADM authorities have implicit execute privilege on the routine and any packages executed within the routine.

System Modification Program/Extended (SMP/E)
A z/OS tool for making software changes in programming systems (such as DB2) and for controlling those changes.

system period
A pair of columns with system-maintained values that indicates the period of time when a row is valid. See also system-period temporal table and system-period data versioning.

system-period data versioning
Automatic maintenance of historical data by the database manager by using a system period. See also system period and system-period temporal table.

system-period temporal table
A table that is defined with system-period data versioning. See also bitemporal table, system-period data versioning, and history table.

Systems Network Architecture (SNA)
The description of the logical structure, formats, protocols, and operational sequences for transmitting information through and controlling the configuration and operation of networks.

table
A named data object consisting of a specific number of columns and some number of unordered rows. See also base table or temporary table. Contrast with auxiliary table, clone table, materialized query table, result table, and transition table.

table-controlled partitioning
A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are defined in the CREATE TABLE statement.

table function
A function that receives a set of arguments and returns a table to the SQL statement that references the function. A table function can be referenced only in the FROM clause of a subselect.

table locator
A mechanism that allows access to trigger tables in SQL or from within user-defined functions. A table locator is a fullword integer value that represents a transition table.

table space
A page set that is used to store the records in one or more tables. See also partitioned table space, segmented table space, and universal table space.

table space set
A set of table spaces and partitions that should be recovered together for one of the following reasons:

- Each of them contains a table that is a parent or descendent of a table in one of the others.
- The set contains a base table and associated auxiliary tables.

A table space set can contain both types of relationships.

task control block (TCB)
A z/OS control block that is used to communicate information about tasks within an address space that is connected to a subsystem. See also address space connection.

TB
Terabyte. A value of 1 099 511 627 776 bytes.

TCB See task control block.

TCP/IP
A network communication protocol that computer systems use to exchange information across telecommunication links.

TCP/IP port
A 2-byte value that identifies an end user or a TCP/IP network application within a TCP/IP host.

template
A DB2 utilities output data set descriptor that is used for dynamic allocation. A template is defined by the TEMPLATE utility control statement.

temporal table
A table that records the period of time when a row is valid.

temporary table
A table that holds temporary data. Temporary tables are useful for holding or sorting intermediate results from queries that contain a large number of rows. The two types of temporary table, which are created by different SQL statements, are the created temporary table and the declared temporary table. Contrast with result table. See also created temporary table and declared temporary table.

textual XML format
A representation of XML data that uses character values, an approach that allows for direct reading by people.

thread See DB2 thread.

threadsafe
A characteristic of code that allows multithreading both by providing private storage areas for each thread, and by properly serializing shared (global) storage areas.

three-part name
The full name of a table, view, or alias. It consists of a location name, a schema name, and an object name, separated by a period.

time A three-part value that designates a time of day in hours, minutes, and seconds.

timeout
Abnormal termination of either the DB2 subsystem or of an application because of the unavailability of resources. Installation specifications are set to determine both the amount of time DB2 is to wait for IRLM services after starting, and the amount of time IRLM is to wait if a resource that an application requests is unavailable. If either of these time specifications is exceeded, a timeout is declared.

Time-Sharing Option (TSO)
An option in z/OS that provides interactive time sharing from remote terminals.

timestamp
A seven-part value that consists of a date and time. The timestamp is expressed in years, months, days, hours, minutes, seconds, and microseconds.

timestamp with time zone
A two-part value that consists of a timestamp and time zone. The timestamp with time zone is expressed in years, months, days, hours, minutes, seconds, microseconds, time zone hours, and time zone minutes.

trace A DB2 facility that provides the ability to monitor and collect DB2 monitoring, auditing, performance, accounting, statistics, and serviceability (global) data.

transaction
An atomic series of SQL statements that make up a logical unit of work. All of the data modifications made during a transaction are either committed together as a unit or rolled back as a unit.

transaction lock
A lock that is used to control concurrent execution of SQL statements.

transaction program name
In SNA LU 6.2 conversations, the name of the program at the remote logical unit that is to be the other half of the conversation.

transaction-start-ID column
A generated column that is defined with the AS TRANSACTION START ID clause. The value is assigned whenever a row is inserted into the table or any column in the row is updated. A transaction-start-ID column is intended for use in a

system-period temporal table. See also generated column, row-begin column, and row-end column.

transition table
A temporary table that contains all the affected rows of the subject table in their state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the table of changed rows in the old state or the new state. Contrast with auxiliary table, base table, clone table, and materialized query table.

transition variable
A variable that contains a column value of the affected row of the subject table in its state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the set of old values or the set of new values.

tree structure
A data structure that represents entities in nodes, with a most one parent node for each node, and with only one root node.

trigger
A database object that is associated with a single base table or view and that defines a rule. The rule consists of a set of SQL statements that run when an insert, update, or delete database operation occurs on the associated base table or view.

trigger activation
The process that occurs when the trigger event that is defined in a trigger definition is executed. Trigger activation consists of the evaluation of the triggered action condition and conditional execution of the triggered SQL statements.

trigger activation time
An indication in the trigger definition of whether the trigger should be activated before or after the triggered event.

trigger body
The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. A trigger body is also called triggered SQL statements.

trigger cascading
The process that occurs when the triggered action of a trigger causes the activation of another trigger.

triggered action
The SQL logic that is performed when a trigger is activated. The triggered action consists of an optional triggered action condition and a set of triggered SQL statements that are executed only if the condition evaluates to true.

triggered action condition
An optional part of the triggered action. This Boolean condition appears as a WHEN clause and specifies a condition that DB2 evaluates to determine if the triggered SQL statements should be executed.

triggered SQL statements
The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. Triggered SQL statements are also called the trigger body.

trigger granularity
In SQL, a characteristic of a trigger, which determines whether the trigger is activated:

- Only once for the triggering SQL statement
- Once for each row that the SQL statement modifies

triggering event
The specified operation in a trigger definition that causes the activation of that trigger. The triggering event is comprised of a triggering operation (insert, update, or delete) and a subject table or view on which the operation is performed.

triggering SQL operation
The SQL operation that causes a trigger to be activated when performed on the subject table or view.

trigger package
A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

trust attribute
An attribute on which to establish trust. A trusted relationship is established based on one or more trust attributes.

trusted connection
A database connection whose attributes match the attributes of a unique trusted context defined at the DB2 database server.

trusted connection reuse
The ability to switch the current user ID on a trusted connection to a different user ID.

trusted context
A database security object that enables the establishment of a trusted relationship between a DB2 database management system and an external entity.

trusted context default role
A role associated with a trusted context. The privileges granted to the trusted context default role can be acquired only when a trusted connection based on the trusted context is established or reused.

trusted context user
A user ID to which switching the current user ID on a trusted connection is permitted.

trusted context user-specific role
A role that is associated with a specific trusted context user. It overrides the trusted context default role if the current user ID on the trusted connection matches the ID of the specific trusted context user.

trusted relationship
A privileged relationship between two entities such as a middleware server and a database server. This relationship allows for a unique set of interactions between the two entities that would be impossible otherwise.

TSO See Time-Sharing Option.

TSO attachment facility
A DB2 facility consisting of the DSN command processor and DB2I. Applications that are not written for the CICS or IMS environments can run under the TSO attachment facility.

typed parameter marker
A parameter marker that is specified along with its target data type. It has the general form:

```
CAST(? AS data-type)
```

type 2 indexes
Indexes that are created on a release of DB2 after Version 7 or that are specified as type 2 indexes in Version 4 or later.

UCS-2 Universal Character Set, coded in 2 octets, which means that characters are represented in 16-bits per character.

UDF See user-defined function.

UDT User-defined data type. In DB2 for z/OS, the term distinct type is used instead of user-defined data type. See distinct type.

uncommitted read (UR)
The isolation level that allows an application to read uncommitted data. See also cursor stability, read stability, and repeatable read.

underlying view
The view on which another view is directly or indirectly defined.

undo A state of a unit of recovery that indicates that the changes that the unit of recovery made to recoverable DB2 resources must be backed out.

Unicode
A standard that parallels the ISO-10646 standard. Several implementations of the Unicode standard exist, all of which have the ability to represent a large percentage of the characters that are contained in the many scripts that are used throughout the world.

union An SQL operation that involves the UNION set operator, which combines the results of two SELECT statements. Unions are often used to merge lists of values that are obtained from two tables.

unique constraint
An SQL rule that no two values in a primary key, or in the key of a unique index, can be the same.

unique index
An index that ensures that no identical key values are stored in a column or a set of columns in a table.

unit of recovery (UOR)
A recoverable sequence of operations within a single resource manager, such as an instance of DB2. Contrast with unit of work.

unit of work (UOW)
A recoverable sequence of operations within an application process. At any

time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations. In a multisite update operation, a single unit of work can include several *units of recovery*. Contrast with unit of recovery.

universal table space
A table space that is both segmented and partitioned. Contrast with partitioned table space, segmented table space, partition-by-growth table space, and range-partitioned table space.

unlock
The act of releasing an object or system resource that was previously locked and returning it to general availability within DB2.

untyped parameter marker
A parameter marker that is specified without its target data type. It has the form of a single question mark (?).

updatability
The ability of a cursor to perform positioned updates and deletes. The updatability of a cursor can be influenced by the SELECT statement and the cursor sensitivity option that is specified on the DECLARE CURSOR statement.

update hole
The location on which a cursor is positioned when a row in a result table is fetched again and the new values no longer satisfy the search condition. See also delete hole.

update trigger
A trigger that is defined with the triggering SQL operation update.

UR See uncommitted read.

user-defined data type (UDT)
See distinct type.

user-defined function (UDF)
A function that is defined to DB2 by using the CREATE FUNCTION statement and that can be referenced thereafter in SQL statements. A user-defined function can be an external function, a sourced function, or an SQL function. Contrast with built-in function.

user view
In logical data modeling, a model or representation of critical information that the business requires.

UTF-16
Unicode Transformation Format, 16-bit encoding form, which is designed to provide code values for over a million characters and a superset of UCS-2. The CCSID value for data in UTF-16 format is 1200. DB2 for z/OS supports UTF-16 in graphic data fields.

UTF-8 Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems. The CCSID value for data in UTF-8 format is 1208. DB2 for z/OS supports UTF-8 in mixed data fields.

value The smallest unit of data that is manipulated in SQL.

variable
A data element that specifies a value that can be changed. A COBOL elementary data item is an example of a host variable. Contrast with constant.

variant function
See nondeterministic function.

varying-length string
A character, graphic, or binary string whose length varies within set limits. Contrast with fixed-length string.

version
A member of a set of similar programs, DBRMs, packages, or LOBs.

- **A version of a program** is the source code that is produced by precompiling the program. The program version is identified by the program name and a timestamp (consistency token).
- **A version of an SQL procedural language routine** is produced by issuing the CREATE or ALTER PROCEDURE statement for a native SQL procedure.
- **A version of a DBRM** is the DBRM that is produced by precompiling a program. The DBRM version is identified by the same program name and timestamp as a corresponding program version.

- **A version of an application package** is the result of binding a DBRM within a particular database system. The application package version is identified by the same program name and consistency token as the DBRM.
- **A version of a LOB** is a copy of a LOB value at a point in time. The version number for a LOB is stored in the auxiliary index entry for the LOB.
- **A version of a record** is a copy of the record at a point in time.

view
A logical table that consists of data that is generated by a query. A view can be based on one or more underlying base tables or views, and the data in a view is determined by a SELECT statement that is run on the underlying base tables or views.

Virtual Storage Access Method (VSAM)
An access method for direct or sequential processing of fixed- and varying-length records on disk devices.

Virtual Telecommunications Access Method (VTAM)
An IBM licensed program that controls communication and the flow of data in an SNA network (in z/OS).

volatile table
A table for which SQL operations choose index access whenever possible.

VSAM
See Virtual Storage Access Method.

VTAM
See Virtual Telecommunications Access Method.

warm start
The normal DB2 restart process, which involves reading and processing log records so that data that is under the control of DB2 is consistent. Contrast with cold start.

WLM application environment
A z/OS Workload Manager attribute that is associated with one or more stored procedures. The WLM application environment determines the address space in which a given DB2 stored procedure runs.

WLM enclave
A construct that can span multiple dispatchable units (service request blocks and tasks) in multiple address spaces, allowing them to be reported on and managed by WLM as part of a single work request.

write to operator (WTO)
An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that might need to be corrected (in z/OS).

WTO
See write to operator.

WTOR
Write to operator (WTO) with reply.

XCF
See cross-system coupling facility.

XES
See cross-system extended services.

XML
See Extensible Markup Language.

XML attribute
A name-value pair within a tagged XML element that modifies certain features of the element.

XML column
A column of a table that stores XML values and is defined using the data type XML. The XML values that are stored in XML columns are internal representations of well-formed XML documents.

XML data type
A data type for XML values.

XML element
A logical structure in an XML document that is delimited by a start and an end tag. Anything between the start tag and the end tag is the content of the element.

XML index
An index on an XML column that provides efficient access to nodes within an XML document by providing index keys that are based on XML patterns.

XML lock
A column-level lock for XML data. The operation of XML locks is similar to the operation of LOB locks.

XML node
The smallest unit of valid, complete structure in a document. For example, a node can represent an element, an attribute, or a text string.

XML node ID index
An implicitly created index, on an XML table that provides efficient access to XML documents and navigation among multiple XML data rows in the same document.

XML pattern
A slash-separated list of element names, an optional attribute name (at the end), or kind tests, that describe a path within an XML document in an XML column. The pattern is a restrictive form of path expressions, and it selects nodes that match the specifications. XML patterns are specified to create indexes on XML columns in a database.

XML publishing function
A function that returns an XML value from SQL values. An XML publishing function is also known as an XML constructor.

XML schema
In XML, a mechanism for describing and constraining the content of XML files by indicating which elements are allowed and in which combinations. XML schemas are an alternative to document type definitions (DTDs) and can be used to extend functionality in the areas of data typing, inheritance, and presentation.

XML schema repository (XSR)
A repository that allows the DB2 database system to store XML schemas. When registered with the XSR, these objects have a unique identifier and can be used to validate XML instance documents.

XML serialization function
A function that returns a serialized XML string from an XML value.

XML table
An auxiliary table that is implicitly created when an XML column is added to a base table. This table stores the XML data, and the column in the base table points to it.

XML table space
A table space that is implicitly created when an XML column is added to a base table. The table space stores the XML table. If the base table is partitioned, one partitioned table space exists for each XML column of data.

X/Open
An independent, worldwide open systems organization that is supported by most of the world's largest information systems suppliers, user organizations, and software companies. X/Open's goal is to increase the portability of applications by combining existing and emerging standards.

XRF See Extended Recovery Facility.

XSR See XML schema repository.

zIIP See IBM System z9 Integrated Processor.

z/OS An operating system for the System z product line that supports 64-bit real and virtual storage.

z/OS Distributed Computing Environment (z/OS DCE) A set of technologies that are provided by the Open Software Foundation to implement distributed computing.

Index

Numerics

0268 177
32K SORT PRIMARY SPACE field of panel DSNTIP9 327
32K SORT WORK SEG SIZE field of panel DSNTIP9 327
32K SORT WORK TS'S field of panel DSNTIP9 327
32K TEMP PRIMARY SPACE field of panel DSNTIP9 329
32K TEMP WORK SEG SIZE field of panel DSNTIP9 331
32K TEMP WORK TS'S field of panel DSNTIP9 330
4K SORT PRIMARY SPACE field of panel DSNTIP9 325
4K SORT WORK SEG SIZE field of panel DSNTIP9 326
4K SORT WORK TS'S field of panel DSNTIP9 326
4K TEMP PRIMARY SPACE field of panel DSNTIP9 328
4K TEMP WORK SEG SIZE field of panel DSNTIP9 329
4K TEMP WORK TS'S field of panel DSNTIP9 329

A

ABIND subsystem parameter 354
ACBNAME option of APPL statement 892
ACCEL subsystem parameter 253
ACCEPT job 216
ACCESS CONTROL field of panel DSNTIPO3 360
access paths 577
accessibility
  keyboard xix
  shortcut keys xix
active log
  data set
    prefix 293
    sharing DASD 567
    storage requirements 32
  dual logging 290
  installation 514
  size
    estimating 32, 313
adding
  data sharing members 691
  employee information at remote locations 823
adding data sharing members
  CLIST 692
  connecting CICS 698
  user authorization exit routines 695
additional capabilities
  for DB2 833
address spaces
  administrative task scheduler
    storage requirements 48
  allied agent 48
  database services
    description 48
    working calculation 60
    working storage calculation 61
  DDF 46
  DSN1DIST 46
  IRLM 47
  system services 47
  WLM-established stored procedures 48
  WLM-managed 839
ADMIN SCHEDULER field of panel DSNTIPX 458
administrative enablement routines
  enabling 854
administrative task scheduler
  address space 512, 557, 633
    storage requirements 48
  address space startup procedure 557, 633
  routines 557, 633
    enabling 854
  set up 557, 633
  table spaces 557, 633
ADMTPROC subsystem parameter 253
ADVANCED CONFIGURATION OPTIONS field of panel DSNTIPR1 462
advantages
  non-volatile coupling facility 174
AGGREGATION FIELDS field of panel DSNTIPN 350
ALCUNIT subsystem parameter 434
alias
  server location 885
allied agent address space 48
ALLOCATION UNITS field of panel DSNTIPA 434
ALTER GROUPBUFFERPOOL 185
AMDPRECT module 508
APN option of DSNMAPN macro 874, 875
APPC option of APPL statement 891
APPL DEFAULTS field of panel DSNTIPO3 359
APPL REGISTRATION TABLE field of panel DSNTIPZ 488
APPL statement 888, 902
  example 888
  options
    description 888
    DSESLIM 909
    LUNAME 916
application defaults load module
  installation 516
application defaults parameters 678
APPLICATION ENCODING field of panel DSNTIPF 370
application environment
  WLM 839
APPLICATION LOAD field of panel DSNTIPT 297
archive log
  cataloging options 276
  data set
    prefix 294
    sharing DASD 567
    storage requirements 45
  dual logging 290
ARCHIVE LOG FREQ field of panel DSNTIPL 427
ARCHIVE LOG RACF field of panel DSNTIPP 410
ART/ORT ESCAPE CHARACTER field of panel DSNTIPZ 487
ASCII
  case conversion 953
  conversion table 945
ASCII CODED CHARACTER SET field of panel DSNTIPF 369
ASSISTANT field of panel DSNTIPK 289
ATCSTRxx member of SYS1.VTAMLST library 901
attachment facilities
  CICS 529
    adding data sharing members 715

© Copyright IBM Corp. 1982, 2012

attachment facilities *(continued)*
CICS *(continued)*
installation 529
migration 588
IMS
adding data sharing members 698
installation 529, 869
migrating data sharing members 715
migration 587
TSO 521
connecting for data sharing 686
installation 521
migration 584
AUDIT TRACE field of panel DSNTIPN 344
AUTH EXIT LIMIT field of panel DSNTIPP 412
AUTH MEMBER field of panel DSNTIPM 424
AUTH option
APPL statement 891
AUTH SEQUENCE field of panel DSNTIPM 425
AUTHID column
MODESELECT table 905
USERNAMES catalog table 929
authorization ID
RACF 520
TSO 520
AUTO BIND field of panel DSNTIPO 354, 647
AUTO START field of panel DSNTIPI 388
automatic
rebind, considerations in a coexistence environment 727
rebuild of coupling facility structures 172
recall 352
automatic rebinds 727
considerations in a coexistence environment 726
automatic recovery 171
automatic remigration rebinds 728
automatic restart 167
advantage 167
policy 167
creating 168
AUTOREC (YES) 171
AUTOSES option of APPL statement 889
availability
catalog and directory 177
coupling facility 169
planning 166
volatility of coupling facility 174

B

BACKODUR subsystem parameter 432
BACKOUT DURATION field of panel DSNTIPL1 432
backup
migration 590
BIF COMPATIBILITY field of panel DSNTIPX 456
BIF_COMPATIBILITY subsystem parameter 456
BIND NEW PACKAGE field of panel DSNTIPP1 419
binding
data sharing environment 201
installation 534
migration
DCLGEN 606
SPUFI 606
recommendations for a coexistence environment 728
remote package
plan name for 905
relation to character conversion 952
BLOCK SIZE field of panel DSNTIPA 437
BookManager-based online help 107
bootstrap data sets (BSDS) 592
BP0 - BP29 fields of panel DSNTIP1 342
BP30 - BP32K9 fields of panel DSNTIP2 343
BSDS (bootstrap data set)
adding second 592
increasing size 196
installation 514
naming recommendation 671
storage requirements 35
BUFFER POOL SIZE field of panel DSNTIPC 494
buffer pools
fallback implications 638
group sizes 181
storage requirement 52
VTAM IOBUF storage requirements 908

C

C/370 COMPILER MESSAGES field of panel DSNTIPU 307
C/CPP COMPILER LIBRARY field of panel DSNTIPU 306
C/CPP COMPILER MODULE field of panel DSNTIPU 305
CACHE DYNAMIC SQL field of panel DSNTIP8 380
CACHE option of CREATE SEQUENCE statement 197
cache sequencing 197
cache structure 161
cache structure sizes 186
CACHEDYN_FREELOCAL subsystem parameter 253
caching all data 184
CAF (call attachment facility)
coexistence considerations 726
catalog alias 153
CATALOG ALIAS field of panel DSNTIPA2 276
CATALOG DATA field of panel DSNTIPA 435
catalog level 724
catalog tables
SYSSTRINGS
description 950
catalog, DB2
checking 603
installing 514
storage requirements 30
catalog, integrated catalog facility 514
CATDDACL subsystem parameter 279
CATDMGCL subsystem parameter 279
CATDSTCL subsystem parameter 279
CATMAINT utility
migration from Version 9.1 531, 602
CATXDACL subsystem parameter 280
CATXMGCL subsystem parameter 280
CATXSTCL subsystem parameter 280
CCSID (coded character set identifier)
code page 945
definition 940
installation option 368
specifying 940
CD-ROM, books on 961
CDB (communications database)
creating during installation 534
description 895, 925
dropping while DDF is active 907
IPNAMES table 928
LOCATIONS table 877, 896, 900, 926
LULIST table 895
LUMODES table 900
LUNAMES table 897
MODESELECT table 900

CDB (communications database) *(continued)*
populating while
connecting DB2 subsystems 895, 925
installing 556
updating while DDF is active 907
USERNAMES table 899, 929
CFRM policy 175, 176
description 149
REBUILDPERCENT parameter 173
CFSizer tool 180
change data capture 605
change log inventory utility
changing
generic LU name 916
location name 916
LU name 916
VTAM password 916
changed data capture 34
channel-to-channel (CTC) 912
character conversion
code page 945
code point 945
contracting conversion 948
description 939
distributed data 939
Euro symbol 949
expanding conversion 948
locale
definition 952
specifying 952
support for euro currency 952
lowercase 953
rebinding remote packages 952
SYSIBM.SYSSTRINGS catalog table 950
Unicode 940
setup 941
uppercase 953
character strings
transmitting 939
CHECK INDEX utility
migration preparation 580
CHECK_FASTREPLICATION subsystem parameter 400
checklists
conversion mode from Version 8 15
conversion mode from Version 9.1 25
premigration from Version 8 14
premigration from Version 9.1 25
checkpoint frequency 34
CHKTYPE field of panel DSNTIPL1 428
CHKTYPE subsystem parameter 428
CICS
adding data sharing members 715
applications, migration considerations in a Sysplex 198
CICSPlex SM 198
connecting to DB2
adding data sharing members 698
installation 529
preparing for data sharing 194
migration 588
operating
starting application 777
testing 774
reconnecting 643
storage handling facilities 778
CICS COBOL LIBRARY field of panel DSNTIPW 313
CICS EXCI LIBRARY field of panel DSNTIPW 313
CICS LOAD LIBRARY field of panel DSNTIPW 312
CICS MACRO LIBRARY field of panel DSNTIPW 313
CICS PL/I LIBRARY field of panel DSNTIPW 313
CICS transaction invocation stored procedure
creating 856
class of service
modifying 903
CLIST 219
adding data sharing members 692
enabling data sharing 680
general instructions 225
jobs tailored by 497
messages 496
migrating data sharing members 712
processing 496
tailoring installation jobs 499
update process 501
CLIST ALLOCATION field of panel DSNTIPA3 283
CLIST LIBRARY field of panel DSNTIPT 297
CLIST messages fields of panel DSNTIPC 495
CLIST messages fields of panel DSNTIPC1 496
CM8 5
CM8* 5
CM9 16
CM9* 16
CNOS (change number of sessions)
description 909
COBOL COMPILER LIBRARY field of panel DSNTIPU 308
coded character set 939
coexistence
requirements for data sharing 723
coexistence of DB2 releases 202
IRLM service 206
WLM-managed stored procedures 723
COLUMNS field of panel DSNTIPD 315
COMCRIT subsystem parameter 253
command line processor
overview 935
prerequisites 935
setting up 936
COMMAND PREFIX field of panel DSNTIPM 423
command recognition character (CRC) 870
COMMAND SCOPE field of panel DSNTIPM 425
commands
prefix
message format 195
common service area (CSA) 44, 49
calculating storage requirements 50
common SQL API stored procedures
enabling 857
communications database (CDB) 877, 897
COMPACT DATA field of panel DSNTIPA 440
compatibility mode
See conversion mode
compatibility mode*
See conversion mode*
COMPRESS SMF RECS field of panel DSNTIPN 351
COMPRESS SPT01 field of panel DSNTIPA2 280
COMPRESS_SPT01 subsystem parameter 253, 280
compression
archive log data set 440
COMPROT option of MODEENT macro 894
configuration
disk connectivity 152
Parallel Sysplex components 147
z/OS system 163
configuration control statements 849

connection
database management systems 877
DB2 896, 926
systems 877
DB2I panels to ISPF
installation 524
migration 586
migration of data sharing members 713
connection exit routine
installation 518
migration 598
connectivity failures 172
connectivity, checking 152
consumer
web service 868
contention
for conversation 889
contention, false
LOCK ENTRY SIZE parameter of installation panel DSNTIPJ 188
CONTRACT THREAD STG field of panel DSNTIPE 337
CONTROL ALL APPLICATIONS field of panel DSNTIPZ 486
conversation
contention 889
definition 883
SQL 904
system 897
conversion mode 65, 114
conversion mode from Version 8 5
conversion mode from Version 9.1 16
conversion mode* 65
conversion mode* from Version 8 5
conversion mode* from Version 9.1 16
CONVLIMIT column of LUMODES catalog table
CNOS negotiation 909
description 904
COORDINATOR field of panel DSNTIPK 289
COPY 1 NAME field of panel DSNTIPH 291
COPY 1 PREFIX field of panel DSNTIPH 292, 294
COPY 2 NAME field of panel DSNTIPH 292
COPY 2 PREFIX field of panel DSNTIPH 293, 294
COPY field of panel DSNTIP6 402
coupling facility
availability 169
channel failure 172
failure-isolated 169
failures 170
lock name 161
non-failure-isolated 169
non-volatile 174
placement 169
relation to INITSIZE 181
storage
group buffer pool 181
lock 187
SCA 189
structure sizer 180
structures
allocation size 180
changing size of 180, 190
estimating storage 179
policy definition 149
types 147
volatile 174
CPP AUTO CALL LIBRARY field of panel DSNTIPU 307
CPP CLASS LIB HEADERS field of panel DSNTIPU 307
CPP CLASS LIBRARY field of panel DSNTIPU 308
CPP PROCEDURE LIBRARY field of panel DSNTIPU 308
CRC (command recognition character)
installation 870
CREATE SEQUENCE statement 197
critical data 177
CROSS MEMORY
IRLM 47
cross-system coupling facility (XCF) 147
CSA (common service area) 44, 49
CTC (channel-to-channel)
considerations for MAXBFRU size 912
sample definitions for VTAM 912
X' sense code 912
CURRENT DEGREE field of panel DSNTIP8 379
CURRENT MAINT TYPES field of panel DSNTIP8 384
CURRENT REFRESH AGE field of panel DSNTIP81 383

D

data
compression 830
distributed 877
data capture changes 605
data compression
Huffman 830
Data Facility Sort (DFSORT) 509
DATA SET NAME PREFIX field of panel DSNTIPA1 271
DATA SET NAME SUFFIX field of panel DSNTIPA1 272
DATA SET(MEMBER) NAME field of panel DSNTIPA1 269
data sets
altering for data sharing 684
control block size calculation 59
defining for new members 694
storage requirements 29
data sharing
adding members 691
creating an image copy 702
default storage group 700
defining DB2 to z/OS 698
defining initialization parameters 694
defining system data sets 694
identifying DB2 to z/OS 692
recording data to SMF 696
REXX Language Support 700
running the CLIST 692
starting DB2 699
temporary work files 700
verifying subsystem 703
application design considerations 197
binds 201
catalog merging 703
disablement 736
disabling 737
enabling 679
altering active log data sets 684
connecting to TSO 686
creating an image copy 690
default storage group 688
defining DB2 to z/OS 686
defining initialization parameters 684
identifying DB2 to z/OS 682
introduction 194
recording data to SMF 685
REXX Language Support 688
running the CLIST 680
starting DB2 687
stopping DB2 681

data sharing *(continued)*
enabling *(continued)*
temporary work files 688
verifying subsystem 691
migrating
stopping DB2 715
migrating groups
first member 711
new-function mode 723
preparation 146
subsequent members 712
migrating members
running the CLIST 712
migrating transactions 200
Parallel Sysplex 147
procedures 669
re-enablement 736
re-enabling 740
release coexistence 203, 723
DATA SHARING field of panel DSNTIPA1 268
data sharing group
creating 669
database
DSNDB06 (DB2 catalog database) 514
DSNDB07 (work file database) 532
database descriptors (DBDs) 191
database services address space 48
DATABASES field of panel DSNTIPD 314
DATABASES field of panel DSNTIPE 333
DATASET STATS TIME field of panel DSNTIPN 348
DATE FORMAT field of panel DSNTIP4 375
DB2 books online 961
DB2 Connect
creating database during installation 535
storage estimation 45
DB2 GENERIC LUNAME
field of panel DSNTIPR 885
DB2 GENERIC LUNAME field of panel DSNTIPR 447
DB2 Information Center for z/OS solutions 961
DB2 LOCATION NAME field of panel DSNTIPR 444
DB2 NETWORK LUNAME field of panel DSNTIPR 444
DB2 NETWORK PASSWORD field of panel DSNTIPR 444
DB2 web services
enabling 568, 863
DB2-supplied
stored procedures 833
user-defined functions 833
DB2-supplied routines 833
DB2I (DB2 Interactive)
connecting to 586, 713
connecting to ISPF
installation 524
panels
verifying values 578
DB2SORT subsystem parameter 409
DBACRVW subsystem parameter 419
DBADM CREATE AUTH field of panel DSNTIPP1 419
DBALIAS column of LOCATIONS catalog table 896
DBCS (double-byte character set)
identifiers 945
DBD (database descriptor)
size 58
DBRM LIBRARY field of panel DSNTIPT 297, 299
DBRMLIB.DATA library data set
DASD volume 283
device type 282
DSNTIJIN job 514
DBRMLIB.DATA library data set *(continued)*
installing second DB2 subsystem 564, 566
naming considerations 212
DCLGEN (declarations generator)
installation 568
migration 606
DDCS (data definition control support)
creating database during installation 535
storage estimation 45
DDF (distributed data facility)
address space 46
installation
implications 882
overview 877
DDF STARTUP OPTION field of panel DSNTIPR 443
DDF THREADS field of panel DSNTIPR 445
DDF_COMPATIBILITY subsystem parameter 253
DDF/RRSAF ACCUM field of panel DSNTIPN 349
DDLTOX subsystem parameter 253
DDRAINL option of APPL statement 892
DEADLOCK CYCLE field of panel DSNTIPJ 395
deadlock cycles 395
DEADLOCK TIME field of panel DSNTIPJ 395
DEALLOC PERIOD field of panel DSNTIPA 438
DECFLOAT ROUNDING MODE field of panel DSNTIPF 371
DECIMAL ARITHMETIC field of panel DSNTIP4 373
DECIMAL POINT IS field of panel DSNTIPF 364
DECLARATION LIBRARY field of panel DSNTIPT 298
declared temporary table
for scrollable cursor 41
longest row size
calculating 42
DECPSSID parameter of DSNHDECP 678
DEF ENCODING SCHEME field of panel DSNTIPF 370
DEFAULT 16-KB BUFFER POOL FOR USER DATA field of panel DSNTIP1 340
DEFAULT 32-KB BUFFER POOL FOR USER DATA field of panel DSNTIP1 341
DEFAULT 4-KB BUFFER POOL FOR USER DATA field of panel DSNTIP1 340
DEFAULT 8-KB BUFFER POOL FOR USER DATA field of panel DSNTIP1 340
DEFAULT BUFFER POOL FOR USER INDEXES field of panel DSNTIP1 342
DEFAULT BUFFER POOL FOR USER LOB DATA field of panel DSNTIP1 341
DEFAULT BUFFER POOL FOR USER XML DATA field of panel DSNTIP1 341
default database (DSNDB04)
storage estimation 44
DEFAULT PARTITION SEGSIZE field of panel DSNTIP7 322
default storage group
adding data sharing members 700
enabling data sharing 688
installing DB2 532
DEFAULT TEMPLATE field of panel DSNTIP6 402
DEFCOLLID option of DSNTP2DP tool 880
DEFER field of panel DSNTIPS 442
DEFINE CATALOG field of panel DSNTIPA2 279
DEFINE DATA SETS field of panel DSNTIP7 319
defining initialization parameters
adding data sharing members 694
enabling data sharing 684
DEL CF STRUCTS field of panel DSNTIPK 290
DEL_CFSTRUCTS_ON_RESTART subsystem parameter 290
deleting
data sets, DSNTIJDE 514

deprecated functions
  Version 8 migration 105
  Version 9.1 migration 139
DESCRIBE FOR STATIC field of panel DSNTIP4 374
DESCSTAT subsystem parameter 374
DEVICE TYPE 1 field of panel DSNTIPA 436
DEVICE TYPE 2 field of panel DSNTIPA 436
DFSESL DD statement 210, 869
DFSMS (Data Facility Storage Management Subsystem)
  installation 410
DFSMShsm (Data Facility Hierarchical Storage Manager)
  RECALL command 352
DFSORT (Data Facility Sort)
  program library 509
directory
  installing 514
  panel field names 229
  panels 228
  storage requirements 31
DIRECTORY AND CATALOG DATA field of panel DSNTIPA2 279
DIRECTORY AND CATALOG INDEXES field of panel DSNTIPA2 280
disability xix
DISABLE_EDMRTS 253
disabling
  data sharing 737
    data recovery 740
DISABSCL subsystem parameter 253
DISALLOW_DEFAULT_COLLID subsystem parameter 253
DISCONNECT IRLM field of panel DSNTIPJ 397
disk, shared requirement 152
DISPLAY GROUP command
  group and member release level 724
DISPLAY NET,BFRUSE command of VTAM 901
DIST SQL STR DELIMTR field of panel DSNTIPF 367
distributed data
  moving to data sharing 704
  planning
    DRDA access 877
    number of systems that can be connected 877
  programming
    character conversion 939
distributed environments 202
Distributed naming convention 164
distributed organization application
  DRDA access 818
distributed unit of work 877
distribution libraries
  manage use DFSMShsm 352
  SMP/E 216
DL/I BATCH TIMEOUT field of panel DSNTIPI 391
DMINWNL option of APPL statement 889
DMINWNR option of APPL statement 889
domain name
  definition 918
domain name server
  definition 918
DPROP SUPPORT field of panel DSNTIPO 356
DPSEGSZ subsystem parameter 322
DRDA (Distributed Relational Database Architecture)
  enhancements
    from Version 8 881
    from Version 9 881
  release coexistence 202
  remote access control 878
DRDA access
  definition 878
  distributed organization application 818
  preparing for 879
  setting up 877
  specifying modes 904
  updating 820
DRDA PORT field of panel DSNTIP5 449
DRDA RDBNAM (relational database name) 885
DRESPL option of APPL statement 892
DSESLIM option of APPL statement
  CNOS negotiation 909
  description 889
DSMAX field of panel DSNTIPC 491
DSMAX limit on open data sets, description 59
DSMAX subsystem parameter 491
DSN1CHKR utility
  migration preparation 580
DSN1COPY utility
  merging DB2 subsystems 704
  migration preparation 580
DSN1DIST address space 46
DSN3@ATH connection exit routine 518
  adding data sharing members 695
DSN3@SGN sign-on exit routine 518
  adding data sharing members 695
DSN3EPX load module 509
DSN3INI load module 509
DSN6ARVP macro 516
DSN6FAC macro 516
DSN6GRP macro 516
DSN6LOGP macro 516
DSN6SPRM macro
  installation 516
DSN6SYSP macro 516
DSN8EAE1 exit routine 830
DSN8HUFF edit routine 830
DSNACEP1 job 216
DSNACICS stored procedure
  creating 856
DSNADMT startup procedure 512, 557, 633
DSNALLOC job 216
DSNAPPL1 job 216
DSNBIND plan name in SYSIBM.MODESELECT table 905
DSNC STRT command 194
DSNCONNS ISPF table 523
DSNDB06 database
  installation job DSNTIJIN 514
DSNDBRMS ISPF table 523
DSNDDF database 877
DSNEMCO1 CLIST 522
DSNEPRI panel of ISPF 524
DSNETBLS data set for ISPF tables 523
DSNHASM procedure 512
DSNHC procedure 512
DSNHCOB procedure 512
DSNHCOB2 procedure 512
DSNHCPP procedure 512
DSNHCPP2 procedure 512
DSNHDECP load module 265
  installation 516
  list of parameters 242
  migration 591, 717
  shared 678
DSNHFOR procedure 512
DSNHICB2 procedure 512
DSNHICOB procedure 512

DSNHMCID module 591
DSNHPLI procedure 512
DSNHSQL procedure 512
DSNJCNVB module 591
DSNJU003 job 925
DSNMAPN macro 874
DSNPLPKN ISPF table 523
DSNRECV1 job 216
DSNRECV2 job 216
DSNRECV3 job 216
DSNRECV4 job 216
DSNTEJ1 job 731
DSNTEJxx job
  installation 744
  migration 636
    data sharing members 722
    Version 8 7, 10, 11, 18
    Version 9.1 21, 22
DSNTESA job 765
DSNTESC job 765
DSNTESD data set member 731
DSNTESE job 765
DSNTESQ queries 582, 832
DSNTIAD sample program
  executes SQL statements 832
  invoked by DSNTEJ1 752
  run by DSNTIJTM 532, 605
DSNTIDXA member 272
DSNTIDXB member 272
DSNTIDxx member 274
DSNTIJAA job 216
DSNTIJAS job 937
DSNTIJCA job 514
DSNTIJCI job 659
DSNTIJDE job 514, 516
  adding data sharing members 694
DSNTIJEN job 655
DSNTIJES job 216
DSNTIJEX job
  installation 518
  migration 598
DSNTIJFT job 737
DSNTIJFV 734
DSNTIJFV job 643
DSNTIJGF job 737
DSNTIJIC job
  adding data sharing members 702
  enabling data sharing 690
  fallback 646
  installation 556
  migration 583, 632
DSNTIJID job 517
  adding data sharing members 695
DSNTIJIE job 517
DSNTIJIF job 517
DSNTIJIN job 515
  adding data sharing members 694
  enabling data sharing 684
  installation 514
  migration 597
DSNTIJMV job
  adding data sharing members 692
  enabling data sharing 682
  installation 508
  migration 594
    data sharing members 718
DSNTIJNF job 663
DSNTIJNG job 663
DSNTIJPD job 880
DSNTIJPM job 572, 608
DSNTIJRA job
  installation 557
  migration 633
DSNTIJRI job 604
DSNTIJRT job 549, 624
DSNTIJRV job 553, 628
DSNTIJRW job
  installation 547
  migration 622
DSNTIJSG job
  installation 534, 535
  migration 606
DSNTIJSS job
  installation 508
  migration 576
DSNTIJTC job 531, 602
DSNTIJTM job 532, 605
DSNTIJUD job 216
DSNTIJUZ job 516, 591, 717
  actions 591
  adding data sharing members 694
  enabling data sharing 684
DSNTIJVC job 522, 585
DSNTIJXA job 608, 664
DSNTIJXB job 608, 664
DSNTIJXC job 608, 664
DSNTIJXZ job 220
DSNTINST CLIST 2, 227
DSNTIP1 installation panel 339
DSNTIP2 installation panel 342
DSNTIP4 installation panel 372
DSNTIP5 installation panel 448
DSNTIP6 installation panel 397
DSNTIP61 installation panel 404
DSNTIP7 installation panel 318
DSNTIP8 installation panel 379
DSNTIP9 installation panel 323
DSNTIPA installation panel 434
DSNTIPA1 installation panel 266
  INSTALL TYPE field 267
DSNTIPA2 installation panel 276
DSNTIPA3 installation panel 281
DSNTIPB installation panel 502
DSNTIPC installation panel 490
DSNTIPC1 installation panel 495
DSNTIPD installation panel 313
DSNTIPE installation panel 333
DSNTIPF installation panel 362
DSNTIPH installation panel 290
DSNTIPI 167
DSNTIPI installation panel 386
DSNTIPJ installation panel 392
DSNTIPK installation panel 286
DSNTIPL installation panel 426
DSNTIPL1 installation panel 428
DSNTIPM installation panel 422
DSNTIPN installation panel 343
DSNTIPO installation panel 351
DSNTIPO1 installation panel 274
DSNTIPO2 installation panel 275
  LICENSE TERMS ACCEPTED 275
DSNTIPO3 installation panel 358
DSNTIPP installation panel 410
DSNTIPP1 installation panel 412

DSNTIPR installation panel 442
DSNTIPS installation panel 441
DSNTIPT installation panel 295
DSNTIPU installation panel 300
DSNTIPW installation panel 309
DSNTIPX installation panel 453
DSNTIPY installation panel 488
DSNTIPZ installation panel 485
DSNTNJxx jobs 216
DSNTP2DP tool
  authorization 880
  option descriptions 880
  outalias output 880
  outpkgs output 880
  outplans output 880
  sample output 880
  syntax 880
DSNTPSMP stored procedure
  setting up 852
DSNTRIN program 549, 624
DSNTRVFY program 553, 628
DSNTWLMB program 848
DSNTXAZP program 221
DSNUTILB entry in PPT 508
DSNUTILS stored procedure
  sample application 787
DSNWZP external module 579
DSNWZPR external module 579
DSNYASCP entry in PPT 508
DSNZPARM
  list of parameters 242
DSNZPxxx
  installing 671
  options 671
DSVCI subsystem parameter 321
dual logging
  specifying 290
dump
  data set size 44
DUMP CLASS NAME field of panel DSNTIP6 399
duplex 176
duplexed group buffer pools. 174
duplexing 174
  affect on CPU usage 177
  coupling facility storage 175
  how it works 175
  options 176
  performance implications 176
  primary and secondary structures 175
  rebuild status 177
  requirements 175
duplexing, benefits 166
DXRRLM00 entry in PPT 508
dynamic SQL
  DSNTESA 830
  DSNTESQ 830
  sample programs 832
  verification statements
    DSNTESA 830
    DSNTESQ 832

E

early code 208
EAS option of APPL statement 892
EBCDIC CCSID field of panel DSNTIPF 368
ECSA (extended common service area) 49
edit routine
  DSN8HUFF 830
EDM DBD CACHE field of panel DSNTIPC 492
EDM LIMIT BELOW THE BAR field of panel DSNTIPC 493
EDM pool
  data sharing storage 193
  size calculation 56
EDM SKELETON POOL SIZE field of panel DSNTIPC 493
EDM_SKELETON_POOL subsystem parameter 493
EDMPOOL STATEMENT CACHE field of panel
  DSNTIPC 492
EDMPOOL subsystem parameter 493
EDMSTMTC subsystem parameter 492
element names for ARM policy 168
employees
  adding at remote locations 823
  removing at remote locations 824
enabling data sharing 679
  CLIST 680
enabling-new-function mode
  actions 654
  from conversion mode 653
  migrating to 653
  returning to 667
enabling-new-function mode from Version 8 5
enabling-new-function mode from Version 9.1 16
enabling-new-function mode* from Version 8 5
enabling-new-function mode* from Version 9.1 16
ENCR option of APPL statement 891
ENCRYPTPSWDS column of LUNAMES catalog table 897
END option of DSNMAPN macro
  description 875
  installation format 874
ENFM8 5
ENFM8* 5
ENFM9 16
ENFM9* 16
erasing
  employee information at remote locations 824
ERLY code 208
estimating storage 179
euro currency support 952
Euro symbol
  character conversion 949
EVALUATE UNCOMMITTED field of panel DSNTIP8 382
EXECUTED STMTS field of panel DSNTIPD 316
EXIT LIBRARY field of panel DSNTIPT 299
exit routine
  description 830
exit routines
  adding a data sharing member 695
EXPLAIN
  stored procedure 857
EXPLAIN PROCESSING field of panel DSNTIPO 355
EXPLAIN tables
  converting after migration to conversion mode (from
    Version 8 or Version 9.1) 608
  converting after migration to new-function mode 664
  converting before migration 65
extended common service area (ECSA) 49
  calculating storage requirements 51
extended English code page 945
extended Katakana code page 945
EXTENDED SECURITY field of panel DSNTIPR 448
external SQL procedures
  configuration 852
  prerequisites for Data Studio 852

external SQL procedures *(continued)*
prerequisites for DSNTPSMP 852
external storage 28
EXTRA BLOCKS REQ field of panel DSNTIP5 451
EXTRA BLOCKS SRV field of panel DSNTIP5 451

F

failure-isolated coupling facility 169
fallback
automatic rebind 638
description 641
frozen objects 638
jobs 643
overview 638
release incompatibilities 638
remigration following 647
steps
Version 8 9
Version 9.1 20
updating your XSR_COMPLETE 647
falling back
data sharing group 734
false lock contention, preventing 188
FAST REPLICATION field of panel DSNTIP6 400
FAST RESTORE field of panel DSNTIP6 400
FCCOPYDDN subsystem parameter 402
fix categories 27
FIXCATs 27
FLASHCOPY PPRC field of panel DSNTIP6 401
FLASHCOPY_COPY subsystem parameter 402
FLASHCOPY_LOAD subsystem parameter 402
FLASHCOPY_PPRC subsystem parameter 401
FLASHCOPY_REBUILD_INDEX subsystem parameter 403
FLASHCOPY_REORG_TS subsystem parameter 403
FMPROF option of MODEENT macro 894
FOR DB2 MQSERIES ROUTINES field of panel DSNTIPR1 462
FOR DB2 REXX ROUTINES field of panel DSNTIPR1 461
FOR DB2 UTILITIES ROUTINES field of panel DSNTIPR1 460
FOR DB2 WEB SERVICES ROUTINES field of panel DSNTIPR1 462
FOR DB2 XML PROCESSING ROUTINES field of panel DSNTIPR1 460
FOR MOST DB2 JAVA ROUTINES field of panel DSNTIPR1 461
FOR MOST DB2 ROUTINES field of panel DSNTIPR1 459
FOR ROUTINES IDENTIFIED TO PROGRAM CONTROL field of panel DSNTIPR1 460
FOR SERIALIZED ROUTINES field of panel DSNTIPR1 460
FOR THE DB2 CICS ROUTINE field of panel DSNTIPR1 461
FOR UNIFIED DEBUGGER ROUTINES field of panel DSNTIPR1 461
FORMAT command in sample application 772, 773
FORTRAN COMPILER LIBRARY field of panel DSNTIPU 309
FORTRAN LINK EDIT LIB field of panel DSNTIPU 309
FROM RELEASE field of panel DSNTIPA1 269
From V8 symbols xvii
From V9.1 symbols xvii
frozen objects 638
function keys 807
function level of IRLM 725

G

GBPCACHE CHANGED clause
defining table spaces and indexes 197
GBPCACHE SYSTEM 185
GBPCACHE(ALL) 184
GBPCACHE(NO) 185
GBPCACHE(YES) 175
GDDM LOAD MODULES field of panel DSNTIPW 312
GDDM MACLIB field of panel DSNTIPW 312
general-use programming information, described 968
GENERIC column of LUNAMES catalog table
description 897
installation panel 885
generic LU name
choosing 153
GET_CONFIG
enabling 857
GET_MESSAGE
enabling 857
GET_SYSTEM_INFO
enabling 857
global deadlock cycle 395
global trace 345
governor (resource limit facility) 353
graphic coded character set identifiers 945
GROUP ATTACH field of panel DSNTIPK 288
group attachment name
after disabling data sharing 737
choosing 153
DECPSSID parameter of DSNHDECP load module 678
defining on IEFSSNxx parmlib member 195
DISPLAY GROUP command 724
migration considerations 155
group buffer pool
description 147
duplexing
description 174
how it works 175
performance 176
failures 170
recovery 172
storage requirement 181
group IP address
associated with data sharing group 153
group name
DB2 153
GROUP NAME field of panel DSNTIPK 287
GROUP parameter of ssnmMSTR startup procedure 679
group restart
recommendations for a coexistence environment 731
GUPI symbols 969

H

hardware requirements 27
HAVAIL option of APPL statement 891
Hierarchical Storage Manager (DFSMShsm) 352
HIGH LEVEL ASSEMBLER LIB field of panel DSNTIPU 305
HONOR_KEEPDICTIONARY subsystem parameter 253
Huffman compression
exit routine 830
HVSHARE parameter 49

I

IBM Data Server Driver
installing 869
IBM LE LINK EDIT LIB field of panel DSNTIPU 304
IBM LE PRELINK MSG LIB field of panel DSNTIPU 305
IBM LE RUNTIME LIBRARY field of panel DSNTIPU 304
IBMDB2LM mode 893
IBMRDB mode 893
IBMUSER authority, establishing authorization ID 520
IDENTIFY/AUTH field of panel DSNTIPO3 360
IDLE THREAD TIMEOUT field of panel DSNTIPR 447
IDRC feature 440
IEAAPFxx member of SYS1.PARMLIB
DSNTIJMV job
installation 509
migration 595
DSNTIPM panel 422
IEFSSN 740
IEFSSNxx member of SYS1.PARMLIB
DSNTIJMV job
installation 509
migration 595
DSNTIPM panel 422
IEFSSNxx parmlib member
command prefix 195
IFCID (instrumentation facility component identifier)
identifiers by number
0267 174
0268 174
IFCIDs 0267 177
IGNORE SORTNUM STAT field of panel DSNTIP61 408
image copy
adding data sharing members 702
data sharing 690
IMMEDIATE WRITE field of panel DSNTIP8 382
IMPDSDEF subsystem parameter 319
IMPDSSIZE subsystem parameter 253
IMPLICIT TIME ZONE field of panel DSNTIP4 377
IMPTKMOD subsystem parameter 253
IMPTSCMP subsystem parameter 320
IMS attachment facility
multiple subsystems 566
IMS BMP TIMEOUT field of panel DSNTIPI 390
IMS connection 869
adding data sharing members 698
data sharing 194
defining
new programs 873
new transactions 873
installation 529, 869
migrating data sharing members 715
migration 587
operating
starting 772
testing 769
reconnecting 643
IMS language interface module (DFSLI000)
generating 874
IMS RESLIB field of panel DSNTIPW 311
IMS.PROCLIB library
description 870
IMSCTRL macro 870
INCLUDE LIBRARY field of panel DSNTIPT 300
incompatibilities of releases 575
Version 8 migration 65
applications and SQL 66
commands 94
incompatibilities of releases *(continued)*
Version 8 migration *(continued)*
deprecated functions 105
other 98
security 95
storage 96
unsupported functions 107
utilities 92
Version 9.1 migration 114
applications and SQL 114
commands 132
deprecated functions 139
other 134
security 132
unsupported functions 141
utilities 130
INDEX SPACE ALLOCATION field of panel DSNTIP7 320
INDEX_IO_PARALLELISM subsystem parameter 253
indexes
type 1 572
initial program load (IPL) 529
adding data sharing members 698
enabling data sharing 686
INITSIZE parameter of CFRM policy
relation to SIZE 181
INLISTP subsystem parameter 253
INPUT MEMBER NAME field of panel DSNTIPA1 272
INSTALL DD CONTROL SUPT field of panel DSNTIPZ 486
INSTALL IRLM field of panel DSNTIPI 387
INSTALL TYPE field of panel DSNTIPA1 267
installation 507
additional capabilities 560
choosing options to enable data sharing 669
defining DB2 to z/OS 508
description 3
IRLM 386, 392
macros
DSN6ARVP 516
DSN6FAC 516
DSN6GRP 516
DSN6LOGP 516
DSN6SPRM 516
DSN6SYSP 516
output 225
overview 1, 3
panels
description 228
updating 503
preparing 27
second subsystem 561
steps 3
tailoring session 3
tapes or cartridges 207
tools 2
verification 560
buffer pool caching for data sharing 731
concurrency for data sharing 732
for data sharing 731
global lock serialization for data sharing 732
planning 748
Sysplex query parallelism for data sharing 733
testing batch environment 758
testing CICS environment 774
testing IMS environment 769
testing PL/I batch 764
testing SPUFI 764
testing TSO attachment facility 752

installation CLIST 219, 225
adding data sharing members 692
calculating disk requirements 46
enabling data sharing 680
general instructions 225
jobs tailored by 497
messages 496
migrating data sharing members 712
migration 2
processing 496
running 219
update process 501
installation jobs
description 2
DSNTIJCA 514
DSNTIJDE 514
DSNTIJEN 654
DSNTIJEX 518
DSNTIJIC 556
DSNTIJID 517
adding data sharing members 695
DSNTIJIE 517
DSNTIJIF 517
DSNTIJIN 514
DSNTIJMV 595
adding data sharing members 692
enabling data sharing 682
installation 508
migration 594
migration of data sharing members 718
DSNTIJSG 534, 535
DSNTIJTM 532, 605
adding data sharing members 700
enabling data sharing 688
DSNTIJUZ 516
DSNTIJVC 522
system affinity 224, 488
tailoring 499
installation libraries
performance implications 211
installation panels
DSNTIPR1 458
DSNTIPRA 462
DSNTIPRB 463
DSNTIPRC 464
DSNTIPRD 465
DSNTIPRE 466
DSNTIPRF 467
DSNTIPRH 469
DSNTIPRI 471
DSNTIPRJ 472
DSNTIPRK 474
DSNTIPRL 476
DSNTIPRM 479
DSNTIPRN 481
DSNTIPRO 482
DSNTIPRP 484
installation SYSADM authority
authorization IDs established 520
installation SYSOPR authority
authorization IDs established 520
installation verification procedure (IVP) 646
adding data sharing members 703
enabling data sharing 691
installation verification samples
ODBA stored procedure 792
stored procedure with result set 785
installation verification samples *(continued)*
stored procedure without result set 782
stored procedures 782
installing DB2 507
integrated catalog facility
DSNTIJCA job 514
invoking
CLIST 224
IOBUF buffer pool
calculating storage requirements 908
description 901
IP addresses
definition 918
IPADDR column of IPNAMES catalog table 928
IPL 740
IPL (initial program load)
adding data sharing members 698
enabling data sharing 686
installation 529
migration 599
data sharing members 720
IPv4 932
IPv6 932
IRLM
address space 512
fallback, stopping IRLM during 642
group naming, naming convention 286
installing second DB2 subsystem 561
libraries
load 295
sample 216
target 216
naming convention for group names 286
space
estimating maximum 490
SYS1.PARMLIB updates 509
z/OS system linkage index 508
IRLM (internal resource lock manager)
address space 47
assistant storage 192
automatic restart 167, 168
automatic start of diagnostic traces 671
coexistence 725
coordinator storage 192
coupling facility lock structure size 187
dump formatting module, AMDPRECT 508
entry in PPT 508
function level 725
LOCK ENTRY SIZE parameter of installation panel DSNTIPJ 188
lock storage 191
monitoring storage 192
names 160, 168
priority 192
starting
adding data sharing members 699
after fallback 644
during data sharing migration 720
during migration 600
enabling data sharing 687
installation 530
storage for Sysplex query parallelism 192
subsystem parameters 671
IRLM LOAD LIBRARY field of panel DSNTIPT 299
IRLM XCF GROUP NAME field of panel DSNTIPJ 396
IRLMAUT subsystem parameter 388
IRLMPROC 47, 512

IRLMRWT subsystem parameter 387
IRLMSID subsystem parameter 387
ISPF
  migration 586
  migration of data sharing members 713
ISPF (Interactive System Productivity Facility)
  primary option menu, panel connection 524
ISPF ISPLINK MODULE field of panel DSNTIPW 312
ISPF Skeleton Library (ISPSLIB) 220
ISTINCLM mode table 889, 893
IVP (installation verification procedure)
  adding data sharing members 703
  C language options 750
  C++ language options 750
  COBOL options 749
  enabling data sharing 691
  fallback 646
  installation 743
  migration 743
  phases 744
  PL/I options 751
  preparing for 578
  programs-phases relationship 744
IVP DATA LIBRARY field of panel DSNTIPT 300
IX_TB_PART_CONV_EXCLUDE subsystem parameter 253

J

Java support
  installing 561
JAVA_HOME 513
JCLLIB statement 224
JDBC
  IBM Data Server Driver 869
job card information fields of panel DSNTIPY 489
JOB statement
  description 207

L

LANGUAGE DEFAULT field of panel DSNTIPF 364
Language Environment (LE)
  overriding runtime options (installation) 537
  overriding runtime options (migration) 612
  setting up for stored procedures (installation) 537
  setting up for stored procedures (migration) 612
Language Environment runtime library 920
language interface token (LIT) 870
LBACKOUT subsystem parameter 431
LC_TYPE DECP value 371
LEMAX subsystem parameter 455
LEVEL ID UPDATE FREQ field of panel DSNTIPL1 433
library 961
  description 207
  distribution 216
  prefix name 212
  target 216
LIBRARY NAME PREFIX field of panel DSNTIPA1 270
LIBRARY NAME SUFFIX field of panel DSNTIPA1 271
LICENSE TERMS ACCEPTED field of panel DSNTIPO2 275
LIMIT BACKOUT field of panel DSNTIPL1 431
link checker 580
LINK LIST ENTRY field of panel DSNTIPM 425
LINK LIST LIBRARY field of panel DSNTIPT 298
LINK LIST SEQUENCE field of panel DSNTIPM 425
LINKNAME column
  IPNAMES catalog table 928
  LOCATIONS catalog table 878, 896, 926
  USERNAMES catalog table 929
list structure 147
LIT (language interface token) 870
LLA (LNKLST lookaside)
  description 208
  installation 509
  migration 595
LMDENT option of APPL statement 892
LNKLST lookaside (LLA) 208
LNKLSTxx member of SYS1.PARMLIB
  installation 422, 509
  migration 595
LOAD DISTRIBUTION field of panel DSNTIPT 299
LOAD field of panel DSNTIP6 402
LOAD LIBRARY field of panel DSNTIPT 298
load module 730
load module library
  application defaults 516, 591, 717
  DSNHDECP 516
  SDSNEXIT 208
  SDSNLINK 208
  SDSNLOAD 208
load modules
  TSO
    installation 522
    migration 585
loading
  distribution libraries 207
  target libraries 207
LOB INLINE LENGTH field of panel DSNTIPD 317
LOB_INLINE_LENGTH subsystem parameter 317
LOCAL DATE LENGTH field of panel DSNTIP4 377
local deadlock cycles 395
LOCAL TIME LENGTH field of panel DSNTIP4 377
locale
  definition 952
  specifying 952
  supporting Euros 952
LOCALE LC_CTYPE field of panel DSNTIPF 371
location name
  data sharing group 153
location names
  description 885
  updating BSDS 916
location statistics 98, 134, 347
LOCATIONS table
  column
    TCP/IP 926
  columns
    VTAM 896
lock
  structure
    size, determining 187
    storage requirement 187
LOCK ENTRY SIZE field of panel DSNTIPJ 396
lock structure 171
  storage estimate 189
locks
  usage, IRLM 47
LOCKS PER TABLE(SPACE) field of panel DSNTIPJ 393
LOCKS PER USER field of panel DSNTIPJ 394
log
  data set
    sharing DASD 567

log *(continued)*
storage examples 32
LOG COPY 1, BSDS 2 field of panel DSNTIPA3 285
LOG COPY 2, BSDS 1 field of DSNTIPA3 panel 285
log mode table 893
logical unit (LU) 883
LOGLOAD 34
LOGMODE option of MODEENT macro 894
logon mode table entries 893
LOWER function
ASCII
setup 953
Unicode
setup 953
LU 6.2 communications protocols 877
LUNAME column
LUMODES catalog table 897, 904
MODESELECT catalog table 905
LUNAME field of panel DSNTIPR 878
LUNAME option of APPL statement
coding values 889
naming conventions 885
updating BSDS 916

M

MACRO LIBRARY field of panel DSNTIPT 298
main storage 51
maintenance 27
MANAGE THREAD STORAGE field of panel DSNTIPE 338
MAX ABEND COUNT field of panel DSNTIPX 454
MAX BATCH CONNECT field of panel DSNTIPE 336
MAX DATA CACHING field of panel DSNTIP8 384
MAX DEGREE field of panel DSNTIP8 381
MAX KEPT DYN STMTS field of panel DSNTIPE 337
MAX OPEN CURSORS field of panel DSNTIPX 455
MAX OPEN FILE REFS field of panel DSNTIPE 339
MAX REMOTE ACTIVE field of panel DSNTIPE 334
MAX REMOTE CONNECTED field of panel DSNTIPE 335
MAX STORAGE FOR LOCKS field of panel DSNTIPJ 393
MAX STORED PROCS field of panel DSNTIPX 455
MAX TEMP RID field of panel DSNTIP9 332
MAX TEMP STG/AGENT field of panel DSNTIP9 331
MAX TSO CONNECT field of panel DSNTIPE 336
MAX TYPE 1 INACTIVE field of panel DSNTIPR 446
MAX USERS field of panel DSNTIPE 334
MAX_CONCURRENT_PKG_OPS subsystem parameter 253
MAXARCH subsystem parameter 438
MAXBFRU option of LINE statement 912
MAXCSA parameter 47
MAXDATA option of VTAM, considerations for NCP connections 914
MAXIMUM LE TOKENS field of panel DSNTIPX 455
MAXIMUM TAPE UNITS field of panel DSNTIP6 399
MAXKEEPD subsystem parameter 337
MAXPVT option of APPL statement 892
MAXRBLK subsystem parameter 495
MEMBER IDENTIFIER field of panel DSNTIPJ 395
MEMBER NAME field of panel DSNTIPK 287
member names
changing 741
DB2 158
MEMBER parameter of ssnmMSTR startup procedure 679
merging
DB2 data 704
DB2 subsystems 703
merging catalogs 703
message by identifier
DSNJ246I 737
migrating data sharing members
CLIST 712
DSNTIJUZ 717
migration 507, 570
checklists
Version 8 15
Version 9.1 25
CLIST 2
running 219
coexistence of DB2 releases 202
considerations
Version 8 65
Version 9.1 114
conversion mode 571
data sharing group 711
first member 711
new-function mode 723
subsequent members 712
enabling-new-function mode
from conversion mode* 667
modes 5, 16
new-function mode 649
from conversion mode* 668
from enabling-new-function mode 661
from enabling-new-function mode* 668
implications 650
overview 1
Version 8 5
Version 9.1 16
preparing 27
release incompatibilities 575
sample objects, required availability 578
tailoring session 3
tools 2
type 1 indexes 572
verifying views 631
migration jobs
description 2
DSNTIJEX 598
DSNTIJIC 583
DSNTIJIN 597
DSNTIJMV 594, 718
DSNTIJNF 663
DSNTIJPM 572
DSNTIJRI 604
DSNTIJTC 531, 602
DSNTIJUZ 591
DSNTIJVC 585
migration steps
conversion mode
Version 8 7, 18
conversion mode* 12, 23, 665, 666
enabling-new-function mode
from conversion mode* 13, 23, 667
Version 8 10
Version 9.1 21
enabling-new-function mode* 12, 23, 665, 666
new-function mode
from conversion mode* 13, 24, 668
from enabling-new-function mode* 14, 24, 668
Version 8 11
Version 9.1 22
MINIMUM DIVIDE SCALE field of panel DSNTIP4 373
MINUTES/CHECKPOINT field of panel DSNTIPL1 430
mixed CCSIDs (coded character set identifiers) 945

MIXED DATA field of panel DSNTIPF 367
MNOTE macro error 875
MNSPAS parameter 543
mode of VTAM
  adding new modes 894
  VTAM sessions
    associating with sessions 903
    default 893
    IBMDB2LM 893
    IBMRDB 893
    SNASVCMG 893
mode table
  default 889, 893
  entering modes 893
MODEENT macro 893
MODENAME column
  LUMODES table 904
  MODESELECT table 905
modes, migration 5, 16
MODESELECT column
  LUNAMES catalog table 897
MODETAB option of APPL statement 889
modified jobs 227
MODIFY IRLMPROC SET,LTE= 189
modifying OBIDs 709
MON subsystem parameter 348
MONITOR SIZE field of panel DSNTIPN 348
MONITOR TRACE field of panel DSNTIPN 348
monitoring
  VTAM buffer pools
    DISPLAY NET, BFRUSE 901
    MODIFY command of MVS 901
multi-site updates
  APPL options 891
  MODEENT options 893
multiple subsystems
  connecting IMS attachment facility 566
  connecting TSO attachment facility 565
  implications 561
  installing 564
  loading libraries 562
  overview 561
  preparing 566
  tailoring installation jobs 563
  verifying installation 567
MVS PARMLIB updates 422

N

naming
  coupling facility structures 161
  data sharing recommendations 162
  example 164
  group names 153
  IRLM 160
  member names 153, 158
national language
  character set identifiers 945
  double-byte character set identifiers 945
NCP-connected DB2 subsystems
  considerations for MAXDATA option 914
  sample definitions 914
NETID 885
NetView
  installation 560
new-function mode 5, 16, 65, 114
  implications 650
new-function mode *(continued)*
  migrating to 661
  returning to 667
NEWAUTHID
  column of USERNAMES catalog table 929
NFM 5, 16
NOCACHE option of CREATE SEQUENCE statement 197
non-failure-isolated coupling facility 169
non-volatile coupling facility 175
NON-VSAM DATA field of panel DSNTIPA3 284
Notice and acceptance of OTC license panel 275
NPGTHRSH subsystem parameter 253
NSYSLX parameter of IEASYSxx parmlib member 152
NUMBER OF COPIES field of panel DSNTIPH 292, 293
NUMBER OF LOCK ENTRIES field of panel DSNTIPJ 396
NUMBER OF LOGS field of panel DSNTIPL 426
NUMBER OF TCBS field of panel DSNTIPX 454
NUMLKTS subsystem parameter 393
NUMLKUS subsystem parameter 394

O

OBIDs
  modifying 709
OBJT REGISTRATION TABLE field of panel DSNTIPZ 488
ODBA stored procedure
  sample 792
OJPERFEH subsystem parameter 253
online 961
online books 961
online help 107
OPERCNOS option of APPL statement 892
OPT1ROWBLOCKSORT subsystem parameter 253
OPTHINTS subsystem parameter 380
OPTIMIZATION HINTS field of panel DSNTIP8 380
OPTIMIZE EXTENT SIZING field of panel DSNTIP7 321
OPTION, option of DSNMAPN macro 874
OPTIOWGT subsystem parameter 253
order-dependent transactions 200
organization application
  format command 772
  panels 803
  with IMS 772
OTC license usage panel 274
OTC_LICENSE subsystem parameter 253
OUTPUT BUFFER field of panel DSNTIPL 427
OUTPUT MEMBER NAME field of panel DSNTIPA1 274
overriding built-in conversion 950

P

pacing
  controlling 902
    SRCVPAC option of MODEENT macro 894, 903
  VPACING option of VTAM APPL statement 889
PACKAGE AUTH CACHE field of panel DSNTIPP 411
PACKAGE LISTS field of panel DSNTIPD 316
PACKAGE STATEMENTS field of panel DSNTIPD 316
packages
  invalid after migration 581
packages bound on DB2 Version 6 and before 66, 114
PACKAGES field of panel DSNTIPD 316
PACKAGES option of DSNTP2DP tool 880
PAD NUL-TERMINATED field of panel DSNTIP4 378
PADIX subsystem parameter 338
PAGE PROTECT field of panel DSNTIPJ 392

panels
directory 228
field names directory 229
ISPF 3, 219
organization 803
projects 803
PARA_EFF subsystem parameter 381
Parallel Sysplex
planning 147
Parallel Sysplex requirements 147
PARALLELISM EFFICIENCY field of panel DSNTIP8 381
PARAMDEG subsystem parameter 381
PARAMETER MODULE field of panel DSNTIPO3 358
parameters
defined by DSNTIJUZ 253
PARMLIB update options 422
PARSESS option of APPL statement 891
password
VTAM 887
PASSWORD column of USERNAMES catalog table 929
PCLOSEN subsystem parameter 433
PCLOSET subsystem parameter 433
performance
planning 64
performance implications
installation libraries 211
performance trace
monitor structure rebuild 174
PERMANENT UNIT NAME field of panel DSNTIPA3 282
phone application
batch processing 816
CICS 777
data set format 816
format command 773
JCL 816
panels 815
program description 814
transaction code 777
PIU (path information unit)
description 908
relationship to MAXBFRU 912
PL/I COMPILER LIBRARY field of panel DSNTIPU 309
PL/I options 523
PLAN AUTH CACHE field of panel DSNTIPP 411
PLAN MANAGEMENT field of panel DSNTIP8 385
PLAN MANAGEMENT SCOPE field of panel DSNTIP8 385
PLAN option of DSNMAPN macro
description 875
installation format 874
PLAN STATEMENTS field of panel DSNTIPD 316
PLANNAME column
SYSIBM.MODESELECT catalog table 905
planning storage 28
planning XA
Global Transactions 178
xa 178
plans
remote bind name 905
plans bound on DB2 Version 6 and before 66, 114
PLANS field of panel DSNTIPD 315
PLANS option of DSNTP2DP tool 880
policy
automatic restart (ARM) 167
CFRM (coupling facility resource management) 149
DUPLEX parameter of CFRM policy 149
INITSIZE parameter of CFRM policy 149
PREFLIST parameter of CFRM policy 149
policy *(continued)*
SFM (Sysplex failure management) 173
SIZE parameter of CFRM policy 149
POOL THREAD TIMEOUT field of panel DSNTIP5 452
populating
CDB
connecting subsystems 895, 925
installation 556
PORT
column of LOCATIONS table 926
port numbers 922
ports
definition 918
ephemeral 918
server 918
well-known 918
PPINFOTABLE option of DSNTP2DP tool 880
PPT (z/OS program properties table) 508
PR/SM™ formulas 186
prefix
active log 293
archive log 294, 435
library 212
log 294
premigration 572
Version 8 checklist 14
Version 9.1 checklist 25
PRIMARY QUANTITY field of panel DSNTIPA 434
primary structures 175
PRIPROT option of MODEENT macro 894
PRIQTY subsystem parameter 434
PRIVATE_PROTOCOL subsystem parameter 253
PROC NAME field of panel DSNTIPI 388
product-sensitive programming information, described 969
profile tables
storage requirements 45
program libraries 208
program preparation 523
programming interface information, described 968, 969
PROGxx member of SYS1.PARMLIB 509, 595
project application
format command 772
panels 803
updating activities 808
viewing project details 808
with IMS 772
PROTECT subsystem parameter 410
provider
web service 866
PRTCT option of APPL statement 887, 889
PSERVIC option of MODEENT macro 894
PSPI symbols 969
PSTOP transaction type 875
PTASKROL subsystem parameter 253
PTFs 27

Q

QUERY_ACCEL_OPTIONS subsystem parameter 253
QUERY_ACCELERATION subsystem parameter 253
QUIESCE PERIOD field of panel DSNTIPA 440
QUIESCE subsystem parameter 440

R

RACF (Resource Access Control Facility)
  option to specify at installation or migration 410
RANDOM ATTACH field of DSNTIPK 289
RANDOMATT subsystem parameter 289
RANDOMIZE XML DOCID field of panel DSNTIP8 385
RCT (resource control table)
  installation 529
re-enabling DB2 data sharing 740
READ COPY2 ARCHIVE field of panel DSNTIPO 358
READ TAPE UNITS field of panel DSNTIPA 437
real storage 62
REAL TIME STATS field of panel DSNTIPO 358
REALSTORAGE_MANAGEMENT subsystem parameter 253
REALSTORAGE_MAX subsystem parameter 253
rebinding
  avoiding in a coexistence environment 726
  remote packages 952
rebinds
  disallowing all 727
  disallowing automatic remigration 728
REBUILD INDEX field of panel DSNTIP6 403
rebuild structures
  SETXCF START,REBUILD command 190
  specifying rebuild threshold 173
REBUILDPERCENT parameter of CFRM policy 173
REC_FASTREPLICATION subsystem parameter 400
RECALL DATABASE field of panel DSNTIPO 352
RECALL DELAY field of panel DSNTIPO 353
recording data to SMF
  adding data sharing members 696
  data sharing 685
RECORDING MAX field of panel DSNTIPA 438
RECORDS/CHECKPOINT field of panel DSNTIPL1 429
recovering data
  data sharing 740
recovery
  automatic recovery of group buffer pools 170
recovery job DSNTIJDE 514
REGISTRATION DATABASE field of panel DSNTIPZ 487
REGISTRATION OWNER field of panel DSNTIPZ 487
registration tables for DDL, recommendation 671
relational database name 885
release coexistence
  DB2 202
  IRLM service 206
release dependency marker 638
release incompatibilities 575
  Version 8 migration 65
    applications and SQL 66
    commands 94
    deprecated functions 105
    other 98
    security 95
    storage 96
    unsupported functions 107
    utilities 92
  Version 9.1 migration 114
    applications and SQL 114
    commands 132
    deprecated functions 139
    other 134
    security 132
    unsupported functions 141
    utilities 130
remigrating
  data sharing group 735
remigration 10, 21, 647
REMOTE LOCATION field of panel DSNTIPY 489
remote unit of work 877
removing
  employee information at remote locations 824
renaming member 741
REO (region error option)
  default OPTION of DSNMAPN macro 875
  installation 870
REORDERED ROW FORMAT field of panel DSNTIP7 322
REORG INDEX field of panel DSNTIP6 404
REORG TABLESPACE field of panel DSNTIP6 403
REORG_IGNORE_FREESPACE subsystem parameter 253
REORG_LIST_PROCESSING 253
REORG_PART_SORT_NPSI subsystem parameter 253
REQUIRE FULL NAMES field of panel DSNTIPZ 486
Resource Access Control Facility (RACF) 410
RESOURCE AUTHID field of panel DSNTIPP1 418
resource control table (RCT) 529
resource limit facility (governor)
  authorization ID 418
  creating database during installation 535
  specifying 353
  storage estimation 45
resource management policy. 172
Resource Recovery Services (RRS)
  log streams used (installation) 538
  log streams used (migration) 613
RESOURCE TIMEOUT field of panel DSNTIPI 387
resource translation table (RTT) 870
restart automatic 167
RESTART field of panel DSNTIPS 442
restart light 168
RESTORE/RECOVER field of panel DSNTIP6 398
RESTRICT_ALT_COL_FOR_DCC subsystem parameter 253
RESYNC INTERVAL field of panel DSNTIPR 445
RESYNC PORT field of panel DSNTIP5 450
RETAINED LOCK TIMEOUT 167
RETAINED LOCK TIMEOUT field of panel DSNTIPI 391
RETENTION PERIOD field of panel DSNTIPA 439
RETLWAIT subsystem parameter 391
RETVLCFK subsystem parameter 253
REVOKE DEP PRIV field of panel DSNTIPP1 420
REXX Language Support
  adding data sharing members 700
  enabling data sharing 688
  installing DB2 532
RGFESCP subsystem parameter 487
RID (record identifier) pool
  size 55
RID blocks 55
RID POOL SIZE field of panel DSNTIPC 495
RLF AUTO START field of panel DSNTIPO 353
RLF subsystem parameter 353
RLFERR subsystem parameter 353
RLFERRD subsystem parameter 445
RLST ACCESS ERROR field of panel DSNTIPO 353
RLST ACCESS ERROR field of panel DSNTIPR 445
RLST NAME SUFFIX field of panel DSNTIPO 353
RO SWITCH CHKPTS field of panel DSNTIPL1 433
RO SWITCH TIME field of panel DSNTIPL1 433
ROUTINE AUTH CACHE field of panel DSNTIPP 411
routines 536, 610
  administrative enablement 854
  administrative task scheduler 854
  DB2-supplied
    configuration control statements for 849

routines *(continued)*
DB2-supplied *(continued)*
core WLM environments for 840
installing 548, 623
sample statements for 849
setup 547, 622
ROUTINES CREATOR field of panel DSNTIPP1 421
RRS (Resource Recovery Services)
implementing for stored procedures (installation) 538
implementing for stored procedures (migration) 613
log streams used (installation) 538
log streams used (migration) 613
RTT (resource translation table)
description 875
installation 870
RUNLIB.LOAD library
DASD volume 283
device type 282
DSNTIJIN job 514
installing second DB2 subsystem 564, 566
naming considerations 212
RUNSTATS recommendation for migration 637
RUSIZES option of MODEENT macro 894

S

SAF (security authorization facility) class for coupling facility structures 152
sample application for data sharing 731
sample applications
DATA field value 806
description 803
distributed organization 818
displaying department information 819
displaying department structure 818
DRDA access 780
field procedure 830
function keys 807
ISPF/TSO environment 768
keys 807
LOB 826
starting in ISPF/TSO environment 803
table 829
ODBA stored procedure 792
organization 810
adding departments 811
adding operations 811
editing entries 812
transferring employees 813
output 300
phone 814
photo application 826
viewing photos 829
printing options 804
project 808
remote access
starting in ISPF/TSO environment 781
resume application 826
viewing resumes 827
scenarios 808
SQL procedure 794
SQL procedures processor 794
stored procedure with result set 785
stored procedure without result set 782
utilities stored procedure 787
valid entries 806
verifying installation 743
SAMPLE LIBRARY field of panel DSNTIPT 297
sample VTAM definitions
description 911
NCP-connected DB2 subsystems 914
samples
DSNTESR 849
stored procedures 782
SCA (shared communications area) 161
description 147
storage requirement 189
SCA structure 171
scrollable cursor
declared temporary table for 41
SDSNBASE library 216
SDSNC.H library 216
SDSNCLST library 216
SDSNDBRM library 216
SDSNENU library 216
SDSNEXIT library 216, 516
SDSNINDX library 216
SDSNINST library 216
SDSNIVPD library 216
SDSNLINK library
description 208, 216
suffix 425
SDSNLOAD library
description 216
link list options 208
SDSNLOD2 library 216
SDSNMACS library 216
SDSNSAMP library
description 216
output from panel session 225
SDSNSAMP library, naming recommendation 691
SDSNSHLF library 216
SDSNSPFM library 216
SDSNSPFP library 216
SDSNSPFPE library 216
SDSNSPFPK library 216
SDSNSPFS library 216
SDSNSPFT library 216
SDXRRESL library 216
SDXRSAMP library 216
SEC ADMIN 1 TYPE field of panel DSNTIPP1 416
SEC ADMIN 2 TYPE field of panel DSNTIPP1 417
SEC DEF CREATOR field of panel DSNTIPP1 421
SECACPT option of APPL statement 889
SECADM1 subsystem parameter 415
SECADM1_INPUT_STYLE subsystem parameter 253
SECADM1_TYPE subsystem parameter 416
SECADM2 subsystem parameter 416
SECADM2_INPUT_STYLE subsystem parameter 253
SECADM2_TYPE subsystem parameter 417
second DB2 subsystem
connecting IMS attachment facility 566
connecting TSO attachment facility 565
implications 561
installing 564
loading libraries 562
overview 561
preparing 566
tailoring installation jobs 563
verifying installation 567
SECONDARY QTY field of panel DSNTIPA 435
secondary structures 175
SECPROT option of MODEENT macro 894
SECURE PORT field of panel DSNTIP5 449

security
incompatibilities
Version 8 95
Version 9.1 132
installation 520
adding data sharing members 697
migration 593
data sharing members 718
SECURITY ADMIN 1 field of panel DSNTIPP1 415
SECURITY ADMIN 2 field of panel DSNTIPP1 416
SECURITY_IN column of LUNAMES catalog table 897
SECURITY_OUT column
IPNAMES catalog table 928
LUNAMES catalog table 897
selection of data values 804
SEPARATE SECURITY field of panel DSNTIPP1 417
SEPARATE WORK FILES field of panel DSNTIP9 331
SEPARATE_SECURITY subsystem parameter 417
SEQCACH subsystem parameter 253
SEQPRES subsystem parameter 253
service 27
service names
TCP/IP 918
session limits
VTAM
calculating 907
modifying LUMODES 904
OPERCNOS option of APPL statement 892
specifying in APPL statement 889
session-level pacing 902
sessions
VTAM 883
SET CHECK PENDING field of panel DSNTIP61 409
SETSSI command of z/OS 691
SETXCF START,ALTER command of z/OS 190
SFM (Sysplex failure management) policy of z/OS 173
shared disk, requirement 152
shared memory
preparing for 49
shortcut keys
keyboard xix
sign-on exit routine
installation 518
migration 598
SINGLE VOLUME field of panel DSNTIPA 441
single-byte character set identifiers 945
SITE TYPE field of panel DSNTIPO 357
size calculations
common service area (CSA) storage 50
EDM pool space 56
package skeleton copies 57
prepared-statement cache 56
extended common service area (ECSA) storage 51
sort pool 54
buffer pool storage 55
local storage 54
SJTABLES subsystem parameter 253
skeleton cursor tables (SKCTs) 191
skeleton package tables (SKPTs) 191
SKIP UNCOMM INSERTS field of panel DSNTIP8 383
SMF (System Management Facility)
adding data sharing members 696
buffers 519
enabling data sharing 685
installation 519
SMF ACCOUNTING field of panel DSNTIPN 346
SMF STATISTICS field of panel DSNTIPN 346
SMF89 subsystem parameter 253
SMFACCT subsystem parameter 346
SMFSTAT subsystem parameter 346
SMP/E (System Modification Program/Extended) 207
data sets
for two releases 213
options 213
sharing with IMS 213
job listings 216
loading DB2 libraries 207
SMP/E jobs
ACCEPT 216
APPLY 216
cleanup 216
editing 207
RECEIVE 216
SMP/E steps
adding data sets 213
IRLM
adding new prefix 214
SMS environment
for installation 508
for migration 576
SMSDCFL subsystem parameter 253
SMSDCIX subsystem parameter 253
SNA sense code X'800A' 912, 914
SNASVCMG mode 893
softcopy publications 961
software requirements 27
SONSCIP option of APPL statement 891
sort
program
APF authorization of library 509
SORT LIBRARY field of panel DSNTIPW 310
SORT POOL SIZE field of panel DSNTIPC 494
SPT01 INLINE LENGTH field 281
SPT01_INLINE_LENGTH subsystem parameter 281
SPUFI
access to remote systems 568
binding
migration 606
remotely 568
different terminal CCSIDs 569
recommendations for a coexistence environment 731
testing 764
SQL (Structured Query Language)
processing conversations
description 904
specifying mode 905
SQL cursor
loading data 567
SQL STRING DELIMITER field of panel DSNTIPF 365
SQLJ
IBM Data Server Driver 869
SRBEXIT option of APPL statement 891
SRCLIB.DATA library
DASD volume 283
device type 282
DSNTIJIN job 514
installing second DB2 subsystem 564, 566
naming considerations 212
SRCVPAC option of MODEENT macro
for VTAM 894
used to control pacing 903
SRTPOOL subsystem parameter 494
SSID option of DSNTP2DP tool 880

SSM (subsystem member)
entry in IMS.PROCLIB 870
execution parameter 870
SSNDPAC option of MODEENT macro
for VTAM 894
recommended value 903
STAR JOIN QUERIES field of panel DSNTIP8 384
STARJOIN subsystem parameter 384
START IRLM CTRACE field of panel DSNTIPI 390
START NAMES field of panel DSNTIPS 442
start options
DB2
adding data sharing members 699
data sharing migration 720
enabling data sharing 687
installation 530
migration 600
VTAM 899
starting
DB2
adding data sharing members 699
after fallback 644
during data sharing migration 720
during installation 530
during migration 600
enabling data sharing 687
IRLM
adding data sharing members 699
after fallback 644
during data sharing migration 720
during installation 530
during migration 600
enabling data sharing 687
TSO
adding data sharing members 699
after fallback 644
during data sharing migration 720
during installation 530
during migration 600
enabling data sharing 687
startup procedures
creating for WLM application environment (installation) 543
creating for WLM application environment (migration) 618
STATCLUS subsystem parameter 253
statistics collection for migration 637
STATISTICS HISTORY field of panel DSNTIP61 405
STATISTICS ROLLUP field of panel DSNTIP61 406
STATISTICS SYNC field of panel DSNTIPN 347
STATISTICS TIME field of panel DSNTIPN 347
STD SQL LANGUAGE field of panel DSNTIP4 378
STEPLIB statement of DB2 program libraries 210
stopping
DB2
enabling data sharing 681
migrating data sharing 715
migration 588
storage
calculating
external storage 28
main 51, 61
predefined models 29
real 62
virtual constraints 61
VTAM IOBUF 908
WLM-established address spaces 48
storage *(continued)*
calculating *(continued)*
working 60
coupling facility
calculating 179
duplexing 175
group buffer pool 181
lock structure 187
shared communications area (SCA) 189
EDM pool
data sharing 193
IRLM 192
storage messages fields of panel DSNTIPC 495
storage planning 28
stored procedures
configuring DB2
installation 536
migration 610
DB2 external SQL procedures Processor 606
DB2-supplied 851
disabling validation 851
DSNAEXP EXPLAIN 857
for XML schema support 853
implementing RRS (installation) 538
implementing RRS (migration) 613
Java
Language Environment options 513
management by WLM 839
samples 782
callers 787
ODBA 792
SQL 794
with result set 785
without result set 782
setting up LE (installation) 537
setting up LE (migration) 612
setting up WLM application environments
WebSphere MQ user-defined functions 863
setting up WLM application environments (installation) 543
setting up WLM application environments (migration) 618
use of RRSAF (installation) 538
use of RRSAF (migration) 613
utilities 606, 787
Visual Explain 606
storm drain effect
avoiding with a CICS enhancement 200
avoiding with a CICS exit 199
description of 198
STRING DELIMITER field of panel DSNTIPF 365
structure sizes
changing automatically 191
changing dynamically 190
subgroup attachment name
after disabling data sharing 737
defining on IEFSSNxx parmlib member 195
SUBGRP ATTACH field of panel DSNTIPK 288
SUBSYSTEM MEMBER field of panel DSNTIPM 424
SUBSYSTEM NAME field of panel DSNTIPI 387
SUBSYSTEM NAME field of panel DSNTIPM 422
subsystem parameter module 265
data sharing members 671
generating 501
installation 516
migration 591, 717
option descriptions 292

subsystem parameters
data sharing environment 671
data sharing recommendations 671
defined by DSNTIJUZ 253
DESCSTAT 374
list 242
scope 671
stored procedures (installation) 536
stored procedures (migration) 611
updating 505
DSNTIPB panel 502
updating for data sharing member 678
SUBSYSTEM SEQUENCE field of panel DSNTIPM 424
subsystems
multiple 209
installing 561
name
IEFSSNxx member of SYS1.PARMLIB 509
SUPPRESS SOFT ERRORS field of panel DSNTIPM 426
surviving member 736
symbols
From V8 xvii
From V9.1 xvii
SYNCLVL option of APPL statement 891
syntax diagram
how to read xix
SYS1.PARMLIB library 163
updated by DSNTIJMV 508, 594
SYS1.PROCLIB library 163, 508, 594
SYS1.VTAMLST library 899
SYSADM authority
establish authorization IDs 520
use 556
SYSADM2 subsystem parameter 413
SYSIBM.IPNAMES table of CDB
description 928
SYSIBM.LOCATIONS table of CDB
description 896, 926
example 896, 926
SYSIBM.LUMODES table of CDB
CONVLIMIT column 904
description 904
LUNAME column 904
MODENAME column 904
updating 904
SYSIBM.LUNAMES table of CDB
description 897
example 897
specifying modes 904
updating 904
SYSIBM.MODESELECT table of CDB
bind plan name 905
description 905
search order 905
SYSIBM.USERNAMES table of CDB
description 929
SYSLGRNX table space
increasing its size 196
SYSOPR authority
establish authorization IDs 520
SYSOPR1 subsystem parameter 414
SYSOPR2 subsystem parameter 414
Sysplex domain name, choosing for a group 153
Sysplex query parallelism
designing a database for 197
installation verification test 733
IRLM storage considerations 192
Sysplex requirements for data sharing 147
Sysplex Timer 152
SYSTEM ADMIN 1 field of panel DSNTIPP1 413
SYSTEM ADMIN 2 field of panel DSNTIPP1 413
system conversations
choosing default mode 897
specifying modes 904
SYSTEM LOB VAL STG field of panel DSNTIPD 318
SYSTEM MACLIB field of panel DSNTIPW 310
System Management Facility (SMF) 519
SYSTEM OPERATOR 1 field of panel DSNTIPP1 414
SYSTEM OPERATOR 2 field of panel DSNTIPP1 414
SYSTEM PROCEDURES field of panel DSNTIPW 310
system services address space 47
SYSTEM XML VALUE STG field of panel DSNTIPD 318
SYSTEM_LEVEL_BACKUPS subsystem parameter 398
SYSTEM-LEVEL BACKUPS field of panel DSNTIP6 398

T

TABLE SPACE ALLOCATION field of panel DSNTIP7 320
TABLE SPACES field of panel DSNTIPD 315
TABLES field of panel DSNTIPD 315
TABLES IN STMT field of panel DSNTIPD 317
target library 216
TBSBPLOB subsystem parameter 341
TBSBPOOL subsystem parameter 340
TBSBPXML subsystem parameter 341
TCP/IP
change log inventory 925
multiple stacks 934
PORT parameter 925
RESPORT parameter 925
single stack 934
terminology 918
TCP/IP ALREADY VERIFIED
field of panel DSNTIP5 932
TCP/IP ALREADY VERIFIED field of panel DSNTIP5 450
TCP/IP communications
enabling 919
TCP/IP connections
defining 922
outage protection 931
overview 917
security 935
tuning 931
TCP/IP KEEPALIVE field of panel DSNTIP5 452
TCP/IP stacks 934
IPv4
initializing 932
IPv6
initializing 932
VIPA
initializing 932
TCPALVER subsystem parameter 450
TCPKPALV subsystem parameter 452
TEMP CLIST LIBRARY field of panel DSNTIPT 296
TEMP data set 225
temporary
database storage estimation 35
TEMPORARY DS UNIT NAME field of panel DSNTIP61 405
temporary table space
storage estimation 41
TEMPORARY UNIT NAME field of panel DSNTIPA3 283
testing SPUFI 765
Text Search for DB2 for z/OS
enabling 936

thread
  space 60
TIME FORMAT field of panel DSNTIP4 376
TIME TO AUTOSTART field of panel DSNTIPI 389
TIMEOUT VALUE field of panel DSNTIPX 454
TIMESTAMP ARCHIVES field of panel DSNTIPH 294
TPN (transaction program name) 887
TPN column of LOCATIONS catalog table 896
TRACE AUTO START field of panel DSNTIPN 345
TRACE SIZE field of panel DSNTIPN 346
TRACKER SITE field of panel DSNTIPO 357
TRACSTR subsystem parameter 345
TRACTBL subsystem parameter 346
transaction program name (TPN)
  description 887
transmitting character data 939
TRKRSITE subsystem parameter 357
TSO
  adding data sharing members 699
  coexistence considerations 726
  enabling data sharing 687
  establishing user IDs 520
  installation procedures 521
    data sharing 686
  making load modules available 585
  migration procedures 584
  reconnecting 643
  starting 530, 687, 699
  testing 752
TSO attachment facility
  multiple subsystems 565
TSPROF option of MODEENT macro 894
TSQTY subsystem parameter 320
tuning
  VTAM
    buffer storage 901
    creating new modes 894
    description 900
    MODEENT macro 903
    pacing count 902
    SRCVPAC option of MODEENT macro 903
two-phase commit 878
  DRDA (Distributed Relational Database Architecture) 878
  TCP/IP 933
  VTAM 891
TYPE
  option of MODEENT macro 894
type 1 indexes 572
TYPE column
  USERNAMES catalog table 929

U

U LOCK FOR RR/RS field of panel DSNTIPI 389
UCS-2 940
Unicode
  case conversion 953
  character conversion 940
  conversion support
    customizing 941
  description 940
UNICODE CCSID field of panel DSNTIPF 369
UNICODE IFCIDS field of panel DSNTIPN 349
UNIT subsystem parameter 436
UNIX System Services 920, 923, 930
UNKNOWN AUTHID field of panel DSNTIPP1 418
UNREGISTERED DDL DEFAULT field of panel DSNTIPZ 487
unsupported functions
  Version 8 migration 107
  Version 9.1 migration 141
UPDATE RATE field of panel DSNTIPL 427
updating
  activity 809
  communications database while DDF is active 907
  LUMODES table of CDB 904
  LUNAMES table of CDB 904
  MODESELECT table of CDB 905
  session limits 904
  system parameters 502
UPPER function
  ASCII
    setup 953
  Unicode
    setup 953
UR CHECK FREQ field of panel DSNTIPL1 430
UR LOG WRITE CHECK field of panel DSNTIPL1 431
USE DATA COMPRESSION field of panel DSNTIP7 320
USE FOR DYNAMICRULES field of panel DSNTIP4 374
USE PROTECTION field of panel DSNTIPP 410
USER LOB VALUE STG field of panel DSNTIPD 317
USER XML VALUE STG field of panel DSNTIPD 318
user-defined characters 945
user-defined functions 536, 610
USERNAMES column
  IPNAMES catalog table 928
  LUNAMES catalog table 897
UT DB2 SORT USE field of panel DSNTIP61 409
UT SORT DATA SET ALLOCATION field of panel DSNTIP61 407
UTF-16 940
UTF-8 940
UTIL TEMP STORCLAS field of panel DSNTIP61 405
utilities
  loading 215
  migration considerations 579
  mixed-release data sharing environment, operating in 215
  packaging 214
  recommendations for a coexistence environment 728
  release incompatibilities
    from Version 8 92
    from Version 9.1 130
  SMP/E jobs 215
  suite
    installing 215
  types
    CATMAINT 531, 602
UTILITY TIMEOUT field of panel DSNTIP61 406
UTIMOUT subsystem parameter 406

V

VARY DS CONTROL INTERVAL field of panel DSNTIP7 321
VARY NET command of VTAM
  ACTIVE option 899
verification jobs
  DSNTEJ0 751
  DSNTEJ1 752
  DSNTEJ1L 752, 755
  DSNTEJ1P 752, 756
  DSNTEJ1U 757
  DSNTEJ2A 758
  DSNTEJ2C 759

verification jobs *(continued)*
DSNTEJ2D 759
DSNTEJ2E 760
DSNTEJ2F 760
DSNTEJ2H 761
DSNTEJ2P 761
DSNTEJ2U 762
DSNTEJ3C 766
DSNTEJ3M 768
DSNTEJ3P 766
DSNTEJ4C 769
DSNTEJ4P 769
DSNTEJ5A 774
DSNTEJ5C 774
DSNTEJ5P 774
DSNTEJ6 781
DSNTEJ61 792
DSNTEJ62 793
DSNTEJ63 794
DSNTEJ64 795
DSNTEJ65 796
DSNTEJ66 797
DSNTEJ6D 786
DSNTEJ6P 784
DSNTEJ6R 788
DSNTEJ6S 783
DSNTEJ6T 785
DSNTEJ6U 787
DSNTEJ6V 789
DSNTEJ6W 790
DSNTEJ6Z 791
DSNTEJ7 799
DSNTEJ71 799
DSNTEJ73 800
DSNTEJ75 801
DSNTEJ76 801
DSNTEJ77 802
DSNTEJ78 802
DSNTESA 765
DSNTESC 765
DSNTESE 765
phases 744
programs 744
testing SPUFI 765
verification testing
DDF 778
DRDA 766
dynamic SQL 766
LOB 798
migrating data sharing members 722
VERIFY
option of APPL statement 889
views
verifying 631
VIEWS field of panel DSNTIPD 315
VIPA (virtual IP address) 932
virtual sequential access method (VSAM) 514
virtual storage
calculating constraints 61
calculations 51
layout 46
Virtual Telecommunications Access Method (VTAM) 885
volatility of coupling facility 174
VPACING option 902
VPACING option of APPL statement 889
VSAM (virtual storage access method)
clusters 514
VSAM CATALOG, DEFAULT AND WORK FILE DATABASE field of panel DSNTIPA3 284
VTAM (Virtual Telecommunications Access Method)
APPL statement 888
buffer pools
calculating storage 908
increasing 901
monitoring 901
performance effect 901
commands
DISPLAY BFRUSE 901
VARY NET 899
defining DB2 subsystem 888
log mode table, default 893
MODEENT macro 893
modes, default for DB2 893
sample definitions 911
definitions 911
installing 560
libraries
SYS1.SAMPLIB 893
SYS1.VTAMLST 899
naming 885, 887
example 888
options
not applicable to DB2 893
parallel sessions 889
password 885
choosing 887
specifying in APPL statement 889
updating BSDS 916
planning considerations 885
session limit 907
calculating 907
mode default 889
start options 899
tracing 901
VTAMFRR option of APPL statement 891

W

web service consumer
enabling DB2 868
web service provider
enabling DB2 866
web service user-defined functions
enabling 863
enabling support for AT-TLS 865
enabling support for HTTPS 865
installing 863
supported web service calls 865
WebSphere MQ user-defined functions
AMI-based 860
enabling 859
MQI-based 860
new versions 860
queue manager 862
updating 860
verifying 863
WFDBSEP subsystem parameter 331
WLM application environments
configuring for WebSphere MQ user-defined functions 863
defining (installation) 543
defining (migration) 618
of DB2-supplied routines 840
security 543

WLM application environments *(continued)*
setting up for stored procedures (installation) 543
setting up for stored procedures (migration) 618
WLM ENVIRONMENT field of panel DSNTIPX 455
WLM environments
setup during installation 547
setup during migration 622
WLM panels
setting up application environment (installation) 543
setting up application environment (migration) 618
WLM PROC NAME field of panel DSNTIPX 453
work file
installation job DSNTIJTM 532
adding data sharing members 700
enabling data sharing 688
migration job DSNTIJTM 605
work file database
connectivity requirements 152
considerations when disabling data sharing 737
considerations when re-enabling DB2 data sharing 740
installation job DSNTIJTM 532
adding data sharing members 700
enabling data sharing 688
migration job DSNTIJTM 605
storage estimation 35
table space size
overview 42
WORK FILE DB field of panel DSNTIPK 288
working storage 60
Workload Manager (WLM)
coexistence considerations 723
description 839
management of stored procedures 839
service policy 839
WRITE TO OPER field of panel DSNTIPA 439
WTO ROUTE CODES field of panel DSNTIPO 352
WTOR ROUTE CODE field of panel DSNTIPA 439

X

X LOCK FOR SEARCHED U/D field of panel DSNTIPI 389
XCF (cross-system coupling facility) component of z/OS
DB2 group name 153
description 147
XML schema support
creating stored procedures for 853
XML_RANDOMIZE_DOCID subsystem parameter 385
XRMIOUT exit of CICS 199
XSR_COMPLETE stored procedure
fallback 647

Z

z/OS
defining DB2 to 508
IPL 529
adding data sharing members 698
enabling data sharing 686
z/OS commands
SETSSI 691
ZOSMETRICS subsystem parameter 253

Product Number: 5605-DB2
5697-P31

Printed in USA

GC19-2974-08

Spine information:

DB2 10 for z/OS

Installation and Migration Guide

IBM