DB2 11 for z/OS

What's New?

GC19-4068-00

DB2 11 for z/OS

What's New?

GC19-4068-00

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.

First edition (October 2013)

This edition applies to DB2 11 for z/OS (product number 5615-DB2), DB2 11 for z/OS Value Unit Edition (product number 5697-P43), 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.

Contents

About this information

This information provides an executive overview of new function in DB2® 11 for z/OS®. The topics in this information provide a framework for describing new function in DB2 for z/OS. New functions are categorized according to user benefits such as information on demand, availability, and performance.

In addition, this information summarizes changes that were introduced in this version for DB2 commands, DB2 utilities, SQL statements, the DB2 catalog, DB2 performance monitoring, and instrumentation facility component identifiers (IFCIDs).

This information assumes that your DB2 subsystem is running in Version 11 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.

Who should read this information

This information is written primarily for people who are evaluating and planning for DB2 for z/OS.

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.

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://pic.dhe.ibm.com/infocenter/dzichelp/v2r2/index.jsp.

Chapter 1. Introduction to Version 11

DB2 11 for z/OS (also referred to as Version 11 or DB2 11) delivers key innovations that reduce your total cost of ownership. Version 11 provides enhanced analytics and increased availability, reliability, and security for your business-critical information. In addition, DB2 11 offers improvements that make installation and migration simpler and faster.

Performance improvements

Performance improvements in Version 11 focus on optimizing query processing and reducing CPU processing time without causing significant administration or application changes. However, Version 11 also offers a balanced approach to performance improvements across all types of workloads, whether your workloads are for online transaction processing (OLTP), batch, or utilities.

For more information about performance improvements, see Performance.

Expanded RBA and LRSN log records

In Version 11, the relative byte address (RBA) and log record sequence number (LRSN) log records are expanded from basic 6-byte format to extended 10-byte format. You also can convert the RBA and LRSN to extended 10-byte format to avoid reaching the logging limits. Extending the RBA and LRSN to 10-byte format helps to avoid wrapping of the RBA and LRSN values, which can cause system problems. In addition, the increased precision of the 10-byte format for the LRSN provides performance improvements for data sharing environments.

For more information about this enhancement, see Expanded RBA and LRSN log records.

Availability enhancements

In Version 11 you can alter the limit keys for a partitioned table space without impacting the availability of the data. When you change the limit key values, the data remains available and applications can continue to access the data.

For more information about this and other availability improvements, see Improved availability when altering limit keys and Availability.

Archive transparency

In Version 11, DB2 can automatically insert rows that are deleted from one table into a separate table that is called an *archive table*. Archive tables provide the following benefits:

- DB2 can manage historical data for you. You do not have to manually move data to a separate table.
- Because rows that are infrequently accessed are stored in a separate table, you can potentially improve the performance of queries against the archive-enabled table.

- You can modify queries to include or exclude archive table data without having to change the SQL statement and prepare the application again. Instead, you can control the scope of the query with a global variable.
- You can store archive tables on a lower-cost device to reduce operating costs.

For more information about archive tables, see Support for archive tables.

For information about other ease of use enhancements in Version 11, see Ease of use.

Security and regulatory compliance

With key enhancements in Version 11, DB2 for z/OS and System z® continue to lead the industry in security and auditing.

For more information about security enhancements, see Security and regulatory compliance.

New application features

In Version 11, expanded support for SQL, XML, and temporal tables results in improved application performance.

For more information about expanded application features, see New application features.

Simpler, faster migration

In Version 11, simpler, faster migration results in a faster return on your investment. This version of DB2 for z/OS provides enhancements to the DB2 installation CLIST, ISPF panels, and jobs, and provides new installation verification procedures (IVPs). Also, a new feature helps to streamline the migration process by allowing an application with incompatible SQL or XML to continue running on DB2 11 without requiring code changes. You no longer need to wait for application changes to be planned and delivered for your business to realize the benefits of Version 11.

For more information about migration enhancements, see Migration.

Chapter 2. Performance

In Version 11 of DB2 for z/OS, performance improvements focus on optimizing query processing and reducing CPU processing time without causing significant administration or application changes.

This release of DB2 for z/OS runs on only IBM System z10® and subsequent 64-bit z/Architecture® processors, using z/OS Version 1 Release 13 or later operating systems. This configuration reduces CPU consumption from the start.

Support for indexes that exclude NULL keys

In Version 11, you can improve the performance of an index by excluding NULL keys from an index to reduce its size.

NULL key columns add to the size of an index and can reduce the performance of index scans. If you exclude NULL key columns from an index, DB2 creates index entries only for key columns that are not null. You can specify that an index excludes NULL keys when you create an index by using the CREATE INDEX statement.

Related concepts:

Indexes that exclude NULL keys (Introduction to DB2 for z/OS)

Improved performance of stored procedure calls from ODBC applications

In Version 11, DB2 improves the processing of stored procedure calls from a local ODBC application.

DB2 optimizes the communication between the DB2 ODBC driver and DB2 to execute CALL statements and return result sets more efficiently. In addition, DB2 improves the processing of the result sets that are returned by using limited block fetch and progressive streaming.

Related concepts:

ODBC limited block fetch (DB2 Programming for ODBC)

"Improved performance of ODBC applications that retrieve LOB or XML objects"

Improved performance of ODBC applications that retrieve LOB or XML objects

For ODBC applications, DB2 11 uses limited block fetch by default to retrieve small inline and large LOBs or XML objects. This new behavior can improve performance.

When limited block fetch is enabled, you can control whether LOB and XML objects are returned inline by using the new STREAMBUFFERSIZE initialization keyword.

To disable limited block fetch, set the LIMITEDBLOCKFETCH initialization keyword to 0.

Related concepts:

ODBC limited block fetch (DB2 Programming for ODBC)

"Improved performance of stored procedure calls from ODBC applications" on page 3

Related reference:

DB2 ODBC initialization keywords (DB2 Programming for ODBC)

Buffer pool enhancements

In Version 11, several buffer pool enhancements help to improve performance by controlling the size of a buffer pool.

Now, the DB2 command ALTER BUFFERPOOL provides the keywords VPSIZEMIN and VPSIZEMAX. You can use these keywords to specify the minimum and maximum number of buffer pools to allocate to the active buffer pool when AUTOSIZE(YES) is in effect.

In addition, you can specify the frame size for a buffer pool by using the new keyword FRAMESIZE.

Related reference:

-ALTER BUFFERPOOL (DB2) (DB2 Commands)

Performance improvements for the LOAD utility

In Version 11, the elapsed time of some LOAD utility jobs is reduced because LOAD can use parallelism in more situations than in earlier versions of DB2. LOAD can use parallelism for partitioned table spaces with only one input data set. Also, LOAD can use parallelism for non-partitioned table spaces.

In earlier versions of DB2, LOAD uses parallelism for partitioned table spaces only if you provide a separate input data set for each partition. Providing and maintaining such data sets is difficult, especially if you altered the partition boundaries and then had to redistribute the data across the input data sets. Also, LOAD cannot use parallelism for simple or segmented table spaces.

In Version 11, these restrictions are removed. You no longer need to provide separate input data sets for each partition so that LOAD can use parallelism. Also, LOAD can use parallelism for non-partitioned table spaces. Specifically, LOAD can use parallelism with one input data set for all of the following table space types:

- Simple table spaces
- Segmented (non-universal) table spaces
- Partitioned (non-universal) table spaces
- Range-partitioned universal table spaces

As a result, the elapsed time for the load operation can be reduced.

To specify that you want LOAD to use parallelism if possible, use the new PARALLEL keyword in the LOAD utility control statement.

Related concepts:

Types of DB2 table spaces (Introduction to DB2 for z/OS)

Related tasks:

Improving LOAD performance (DB2 Utilities)

Related reference:

Syntax and options of the LOAD control statement (DB2 Utilities)

Performance improvements for the REORG TABLESPACE utility

Version 11 delivers improved performance for the REORG TABLESPACE utility with the PART option.

In Version 11, when subsystem parameter REORG_PART_SORT_NPSI is enabled, REORG TABLESPACE PART sorts all keys of the nonpartitioned secondary indexes and builds the shadow index from the sorted keys. This improvement potentially reduces CPU time.

In addition, this version delivers potential performance improvements as the following items increase:

- The number of nonpartitioned secondary indexes on the table space
- The amount of data in the logical part of the nonpartitioned secondary indexes
- The total amount of data in the nonpartitioned secondary indexes

Improved performance for BIND and DDL options

Version 11 offers improved performance for the BIND and DDL options on long-running persistent threads.

The new PKGREL_COMMIT subsystem parameter specifies whether, at COMMIT or ROLLBACK, a persistent DB2 thread will release a package that is active on that thread if certain DB2 operations are waiting for exclusive access to that package. If you specify `YES`, for packages that are bound with the RELEASE(DEALLOCATE) option, the following operations are permitted at COMMIT or ROLLBACK while the package is active and allocated by DB2 for a persistent DB2 thread:

- BIND REPLACE PACKAGE and REBIND PACKAGE requests, including auto rebind online schema changes for tables and indexes that are statically referenced by the package
- Online REORG operations that materialize pending definition changes for objects that are statically referenced by the package

With this improvement, you no longer must identify and stop persistent DB2 threads that are running packages that are bound with the RELEASE(DEALLOCATE) bind option before you attempt BIND REPLACE PACKAGE and REBIND PACKAGE requests, or apply schema changes that are associated with those packages.

Related reference:

RELEASE bind option (DB2 Commands)

Faster recovery of catalog and directory objects

In Version 11, the recovery time of certain catalog and directory objects can be faster because unnecessary log scans are reduced.

The following directory objects and any associated indexes are now recorded in DSNDB01.SYSLGRNX:

- DSNDB01.SCT02
- DSNDB01.SPT01
- DSNDB01.SYSSPUXA
- DSNDB01.SYSSPUXB

When one of these objects is open for updates, the SYSLGRNX table stores the log range for the update. When these objects must be recovered, DB2 can use the SYSLGRNX records to determine the appropriate log records to read.

Before Version 11, when these directory objects were not recorded in SYSLGRNX, DB2 could not determine which log records to read and potentially had to scan unnecessary log data. This extra scanning could significantly increase the recovery time of these objects. This increased recovery time also delayed the recovery of the other objects that were included in the same recovery step. (As part of the procedure to recover the catalog and directory, these directory objects are grouped with other objects in the same RECOVER statement.)

This enhancement can improve the recovery time of not only these four directory objects that are now recorded in SYSLGRNX, but of all catalog and directory objects that are recovered in the same recovery step.

The recording of these objects in SYSLGRNX is supported in Version 11 conversion mode. DB2 can use these SYSLGRNX records for recovery in Version 11 new-function mode.

Related concepts:

Point-in-time recovery of the catalog, directory, and all user objects (DB2 Utilities)

Related tasks:

Recovering catalog and directory objects (DB2 Utilities)

Reduction in CPU usage for large numbers of partitions

Version 11 offers performance improvements by reducing CPU usage when table spaces that have a large number of partitions are accessed.

In earlier versions of DB2 for z/OS, you might experience performance degradation after greatly increasing the number of partitions in a table space. Version 11 optimizes partition processing so that CPU usage does not increase dramatically as the number of partitions increase.

CPU reduction for distributed applications

Version 11 reduces CPU usage for distributed applications that select many columns. This enhanced performance is a result of the improved way that DB2 processes the columns.

Improved performance for ROLLBACK TO SAVEPOINT

Version 11 provides improved performance for the ROLLBACK TO SAVEPOINT statement.

In earlier versions of DB2, the performance of this statement progressively deteriorates with repeated rollbacks to the same external savepoint. This performance degradation is caused by the increasing number of log records that are being scanned each time that the ROLLBACK TO SAVEPOINT statement is issued.

In Version 11, the ROLLBACK TO SAVEPOINT statement recognizes the point in the log where the previous savepoint finished. This enhancement improves performance by ensuring that subsequent executions of the ROLLBACK TO SAVEPOINT statement scan only the log records that apply to the amount of work to be undone.

DB2 DPSI performance enhancements

In Version 11, you can specify the maximum degree of parallelism for a parallel group in which a data partitioned secondary index (DPSI) is used to drive parallelism.

A DPSI is a nonpartitioning index that is physically partitioned according to the partitioning scheme of the table. When you specify a value of greater than 0 for the new PARAMDEG_DPSI subsystem parameter, you limit the degree of parallelism for DPSIs. When you limit the degree of parallelism, DB2 does not create extraneous parallel tasks that use virtual storage.

Reductions in synchronous log writes

Version 11 improves performance and stability by reducing the occurrence of synchronous log writes during index structure modification operations in data sharing environments.

Previously, performance problems might result during insert and delete processing for group buffer pool dependent (GBP-dependent) indexes in data sharing environments. Synchronous log writes for the same operation might cause performance degradation.

Now, the number of synchronous log writes for both insert and delete processing is reduced, which improves transaction and batch response times.

Relief from storage shortages in group buffer pools

In Version 11, data sharing members can automatically respond to storage shortages in group buffer pools with minimal impact to your applications. This new behavior is called the *group-buffer-pool write-around protocol*.

Before Version 11, when a group buffer pool sustained a large amount of page-write activity, the coupling facility for the group buffer pool could quickly fill up with changed pages. As a result, applications could be slower, and in severe cases, pages could be placed in the LPL. These LPL pages could result in DB2 data outages.

In Version 11, DB2 automatically detects large amounts of page-write activity and switches to the group-buffer-pool write-around protocol for those objects that have the most write activity. (Typically, the objects that have the most write activity are utility and batch jobs.) Under this new protocol, pages that are changed by write operations are written to the group buffer pool only if they are already cached there. Otherwise, DB2 writes these pages to disk. The other objects (objects that are using the group buffer pool but do not have the most write activity) are not affected by this new protocol. They are able to access the group buffer pool as usual. When the storage shortage is relieved, DB2 switches back to normal write activity for the group buffer pool.

DB2 automatically implements the group-buffer-pool write-around protocol when necessary. However, the following prerequisite software is required:

- z/OS Version 1 Release 12 or Release 13 with the PTFS for OA37550 or z/OS Version 2 Release 1
- Coupling facility control code (CFCC) at CFLEVEL=17

To determine how many pages are written to disk by the group-buffer-pool write-around protocol, issue the DISPLAY GROUPBUFFERPOOL command with the MDETAIL option. In the output from this command, look at the WRITE-AROUND PAGES field in message DSNB777I.

Related reference:

-DISPLAY GROUPBUFFERPOOL (DB2) (DB2 Commands)

Automatic cleanup of pseudo-empty index pages

In Version 11, DB2 automatically cleans up pseudo-deleted index entries and pseudo-empty index pages. This enhancement can reduce the size of some indexes, which can improve SQL performance and reduce the need to run the REORG INDEX utility.

When rows are deleted, index entries are not physically deleted unless the delete operation has exclusive control over the index page set. These index entries are called *pseudo-deleted index entries*. Subsequent searches continue to access these pseudo-deleted entries, which can gradually degrade performance as more rows are deleted. These pseudo-deleted index entries can also result in timeouts and deadlocks for applications that insert data into tables with unique indexes.

Before Version 11, DB2 attempted to clean up pseudo-empty index pages as part of SQL DELETE processing. *Pseudo-empty index pages* are pages that contain only pseudo-deleted index entries. However, if some of the pseudo-deleted entries in the page were not committed during the SQL DELETE processing, cleanup could not be performed. Therefore, some pseudo-empty pages were likely not cleaned up.

In Version 11, in addition to the cleanup that was previously done, DB2 automatically deletes pseudo-empty index pages independently of the SQL DELETE transaction. DB2 also automatically deletes individual pseudo-deleted index entries.

When the system is zIIP-enabled, this automated clean-up function runs under enclave service request blocks (SRBs) that are zIIP-enabled. However, the system does not need to be zIIP-enabled to use this new function.

This new function is designed to run with minimal or no disruption to other concurrent DB2 work in the system.

Related concepts:

IBM System z Integrated Information Processors (DB2 Performance)

Improved duplicate removal from SQL results

In Version 11, DB2 improves the processing of removing duplicates from SQL results.

DB2 might optimize the processing of certain types of SQL statements that remove duplicates during processing. Some improvements require an index. Examples of SQL statements that are affected include:

- SELECT DISTINCT
- Single SET FUNCTION DISTINCT
- GROUP BY without set functions
- GROUP BY with single SET FUNCTION DISTINCT
- Single MAX or MIN set function
- GROUP BY with single MAX or MIN set function
- SELECT with a subquery

Performance improvements for distributed applications

Version 11 provides improved performance for distributed applications that return large result sets.

Version 11 introduces package-based continuous block fetch, which can improve performance for retrieval of large, read-only result sets from a remote DB2 for z/OS server.

Like SQL-based continuous block fetch, package-based continuous block fetch causes fewer messages to be transmitted from the requester to retrieve the entire result set. However, package-based continuous block fetch is easier to configure. It requires only that you bind your applications with the new DBPROTOCOL(DRDACBF) option. You do not need to modify your applications or set subsystem parameters to indicate the maximum number of blocks to be returned for a remote request.

In addition, package-based continuous block fetch is more efficient than SQL-based continuous block fetch. With package-based continuous block fetch, the requester opens a secondary connection to the DB2 server for each read-only cursor. The DB2 server returns extra query blocks until all of the rows for the cursor are retrieved. When the cursor is closed, the secondary connection is implicitly closed.

Package-based continuous block fetch provides a performance advantage for a DB2 for z/OS application that has the following characteristics:

- The application queries only remote sites.
- The application does not contain INSERT, UPDATE, DELETE, or MERGE statements.

- No statement in the application creates a unit of recovery on the remote site. This situation results in an SQL error when the application package is bound for package-based continuous block fetch.

Expanded RBA and LRSN log records

In Version 11, the RBA and LRSN log records are expanded from basic 6-byte format to extended 10-byte format.

In earlier versions, if you reach the end of the RBA log range, you must manually reset the RBA back to zero.

Beginning in Version 11, you can convert the RBA and LRSN to extended 10-byte format to avoid reaching the logging limits. Also, new warning messages alert you when the LRSN is approaching the end of its range.

Related concepts:

When RBA or LRSN limits are reached (DB2 Administration Guide)

Related tasks:

Converting the RBA to extended 10-byte format for non-data-sharing environments (DB2 Administration Guide)

Converting the RBA and LRSN to extended 10-byte format for data sharing environments (DB2 Administration Guide)

Improved access path reuse with warnings

The new APREUSE(WARN) bind option improves the ability to reuse existing access paths for as many statements as possible.

In earlier versions, access path stabilization through the APREUSE option of the BIND PACKAGE and REBIND PACKAGE commands is possible only when the access paths for all statements in a package can be reused successfully. If reuse fails for any statement in a package, reuse cannot be applied to any statements in that package.

In Version 11, the APREUSE(WARN) option enables a package to be successfully rebound when the access paths for some statements cannot be reused. When an access path cannot be reused, a new access path is chosen for the statement.

Related tasks:

Reusing and comparing access paths at bind and rebind (DB2 Performance)

Related reference:

APREUSE bind option (DB2 Commands)

Free space management enhancements

In Version 11, you can reserve free space in table spaces for use only by UPDATE operations.

By reserving the space for UPDATE operations, you can improve performance. The free space enables your data to remain clustered longer and reduces the use of overflow records and indirect references, which degrade performance.

You can specify the PCTFREE FOR UPDATE option when you create or alter a table space. When you specify that option, DB2 reserves space for future UPDATE operations. The reserved space is not used by INSERT operations or utilities that add data to the table space. The FOR UPDATE value specifies the percentage of each page that is reserved to be used only by future UPDATE operations. You can specify `FOR UPDATE -1` to specify that DB2 uses real-time statistics to automatically calculate how much free space to reserve for updates.

The PCTFREE_UPD subsystem parameter specifies the default value of the PCTFREE FOR UPDATE option.

Related tasks:

Reserving free space for table spaces (DB2 Performance)

Related reference:

ALTER TABLESPACE (DB2 SQL)

CREATE TABLESPACE (DB2 SQL)

PERCENT FREE FOR UPDATE field (PCTFREE_UPD subsystem parameter) (DB2 Installation and Migration)

Improved predicate transformations

Version 11 applies more techniques to transform query predicates for processing at earlier stages.

DB2 applies the following types of query transformations to make predicates eligible to be processed at earlier stages:

- Addition of generated predicates
- Removal of unneeded pre-evaluated predicates
- Removal of unneeded table references in certain join predicates

Generally, you do not have to do anything to take advantage of the new query transformations.

Related concepts:

Predicate manipulation (DB2 Performance)

Query transformations (DB2 Performance)

Predicates that DB2 generates (DB2 Performance)

Stage 1 and stage 2 predicates (DB2 Performance)

Indexable and non-indexable predicates (DB2 Performance)

Related reference:

Summary of predicate processing (DB2 Performance)

Predicate selectivity overrides

You can provide information to DB2 about predicates that are difficult for DB2 to estimate. DB2 uses the information that you provide during access path selection.

DB2 cannot estimate filter factors for certain types of predicates. It must rely on inaccurate estimates or even default filter factors to select access paths for statements that use such predicates. Examples include predicates that involve the following items, among others:

- Host variables
- Parameter markers
- Expressions
- Table self-joins
- Subqueries

However, you can override these default filter factors for certain predicates by creating selectivity overrides. Each predicate selectivity override describes the selectivity of a particular predicate in a particular SQL statement. When a statement contains more than one predicate, you can create separate selectivity overrides for each predicate in the statement. To create selectivity overrides, you populate a set of input EXPLAIN tables and issue a BIND QUERY command.

The selectivity overrides are added to certain catalog tables, and DB2 uses the information during access path selection.

Related tasks:

Overriding predicate selectivities at the statement level (DB2 Performance)

Managing query access paths (DB2 Performance)

Related reference:

BIND QUERY (DSN) (DB2 Commands)

DSN_PREDICAT_TABLE (DB2 Performance)

DSN_PREDICATE_SELECTIVITY table (DB2 Performance)

DSN_USERQUERY_TABLE (DB2 Performance)

Catalog tables that BIND QUERY commands populate (DB2 Performance)

Chapter 3. Availability

Version 11 continues to offer improvements in availability, such as support for altering the limit keys for a partitioned table space without impacting the availability of the data, and support for online REORG and work file database enhancements.

Automatic recovery of indexes from GRECP or LPL status

Version 11 improves system and application availability by extending support for the automatic recovery of indexes that are in group buffer pool recovery pending (GRECP) or logical page list (LPL) status during GRECP or LPL recovery.

In earlier versions of DB2, if an index page split or index page delete operation is unfinished when an index is put into GRECP or LPL status, the GRECP or LPL recovery might fail, and the index can be left in REBUILD-pending status. In these cases, a database administrator must rebuild the index, which is a time-consuming process.

Now, DB2 automatically uses second pass log apply to initiate recovery of an index that is in GRECP or LPL status. This enhancement makes the indexes immediately available after GRECP or LPL recovery.

In some rare cases when you run the RECOVER utility or RESTORE SYSTEM utility, an index might be left in REBUILD-pending status. In these cases, you must rebuild the index by running the REBUILD INDEX utility.

Related concepts:

Recovery of pages on the logical page list (DB2 Data Sharing Planning and Administration)

Effects of running RECOVER (DB2 Utilities)

Effects of running RESTORE SYSTEM (DB2 Utilities)

Online schema enhancements

Version 11 improves system and application availability by allowing recovery to a point in time before materialization of pending definition changes.

DB2 Version 10 improved the availability of table spaces and indexes by allowing *pending definition changes*. Pending definition changes are changes that are made by ALTER statements, but are not yet materialized. Examples are changing segment size, data set size, buffer pool page size, and the MEMBER CLUSTER attribute. Users can defer materialization of the changes until a convenient time. Materialization is accomplished by running the REORG utility. However, in Version 10, after the REORG utility was run on the objects, you could not recover those objects to a point in time before the changes were materialized.

In Version 11 you can recover the following objects to a point in time before the pending definition changes were materialized:

- Range-partitioned table spaces
- LOB or XML table spaces

The pending definition changes must meet the following criteria:

- The changes were materialized in Version 11 new-function mode.
- The changes were not changes to the table space type or the hash organization.

Related concepts:

Point-in-time recovery (DB2 Utilities)

Improved availability when altering limit keys

In Version 11, you can alter the limit keys for a partitioned table space without impacting the availability of the data. This capability is called an *online alter limit key*.

In earlier versions of DB2, when you change the limit key values, all affected partitions are placed in REORG-pending (REORP) status. The data is unavailable until you run the REORG utility.

In Version 11, that restriction is removed. When you change the limit key values, the data remains available and applications can continue to access the data. However, in most cases, the limit key changes are not materialized until the next time that you run REORG. The affected partitions are placed in advisory REORG-pending (AREOR) status. This type of definition change, in which the changes are not immediately materialized, is called a *pending definition change*.

In a few cases, a change to a limit key value is immediately materialized, so you do not need to run REORG on the partitions. Immediate materialization occurs when DB2 determines that both of the following conditions are true:

- No data needs to be moved between partitions.
- No other alter limit key operation is pending on the partition.

This new online alter limit key capability is limited to the following types of partitioned table spaces:

- Range-partitioned universal table spaces
- Table spaces that are partitioned (non-universal) with table-controlled partitioning

In addition to improved data availability, online alter limit key provides other benefits. Because altering a limit key is now a pending definition change, you can alter a limit key even when the table space has other pending definition changes. Also, because the materialization of the limit key changes is deferred, you have a time period where you can drop the limit key changes without affecting the target table.

For partitioned table spaces (non-universal) with index-controlled partitioning, altering the limit key still causes the affected partitions to be placed in REORG-pending (REORP) status. Therefore, the data is unavailable until the affected range of partitions is reorganized. To prevent such outages, you can use the new subsystem parameter PREVENT_ALTER_LIMITKEY. Specifying a value of YES for this parameter restricts ALTER TABLE statements that alter limit keys for index-controlled partitioned table spaces.

Related concepts:

"Ability to alter limit keys for materialized query tables" on page 18

"Improvements to DROP PENDING CHANGES" on page 18

Types of DB2 table spaces (Introduction to DB2 for z/OS)

Related tasks:

Changing the boundary between partitions (DB2 Administration Guide)

Related reference:

REORG-pending status (DB2 Utilities)

PREVENT ALTER LIMITKEY field (PREVENT_ALTERTB_LIMITKEY subsystem parameter) (DB2 Installation and Migration)

Online REORG improvements

In Version 11, support for the MAPPINGDATABASE keyword for the REORG TABLESPACE utility and the new REORG_MAPPING_DATABASE subsystem parameter helps to improve data availability.

Now, when the REORG utility processes a REORG TABLESPACE SHRLEVEL CHANGE request, it can create its own mapping table and mapping index instead of relying on user's input. The MAPPINGDATABASE keyword specifies the database in which REORG implicitly creates the mapping table and index objects. The REORG_MAPPING_DATABASE subsystem parameter specifies the default database that REORG TABLESPACE SHRLEVEL CHANGE uses to implicitly create the mapping table.

The MAPPINGDATABASE keyword overrides the subsystem parameter value in the REORG_MAPPING_DATABASE subsystem parameter.

Related reference:

REORG MAPPING DB field (REORG_MAPPING_DATABASE subsystem parameter) (DB2 Installation and Migration)

Syntax and options of the REORG TABLESPACE control statement (DB2 Utilities)

Work file database enhancements

Version 11 provides improved monitoring of storage use in the work file database so that you can alleviate storage shortages before application failures occur.

In Version 11, DB2 provides the following improvements to help you monitor disk storage use in the work file database:

- Two subsystem parameters that provide threshold levels for alerts about work file storage shortages, and two messages that provide the alerts:

  WFSTGUSE_AGENT_THRESHOLD
  This subsystem parameter enables you to control when a message is issued to indicate that an agent is using too much disk space in the work file database. WFSTGUSE_AGENT_THRESHOLD determines the percentage of available space in the work file database on a DB2 subsystem or data sharing member that can be consumed by a single agent before message DSNI052I is issued.

WFSTGUSE_SYSTEM_THRESHOLD
This subsystem parameter enables you to control when a message is issued to indicate that the total disk space that is being used in the work file database by all agents is too great.
WFSTGUSE_SYSTEM_THRESHOLD determines the percentage of available space in the work file database on a DB2 subsystem or data sharing member that can be consumed by all agents before message DSNI053I is issued.

- New statistics class 1 trace records that track work file space usage.

Related reference:

"Changed IFCIDs in Version 11" on page 55

AGENT LEVEL THRESHOLD field (WFSTGUSE_AGENT_THRESHOLD subsystem parameter) (DB2 Installation and Migration)

SYSTEM LEVEL THRESHOLD field (WFSTGUSE_SYSTEM_THRESHOLD subsystem parameter) (DB2 Installation and Migration)

Governing of parallel processing of utilities

In Version 11, you can manage the amount of parallelism by using the new PARAMDEG_UTIL subsystem parameter and the PARALLEL(*num-subtasks*) utility option.

When utilities use parallel processing, they consume more processor resources. This use of processor resources can be an issue for some utilities, such as CHECK INDEX, LOAD, REBUILD INDEX, REORG TABLESPACE, and UNLOAD. The new PARAMDEG_UTIL subsystem parameter and the PARALLEL(*num-subtasks*) utility option enable you to control the amount of parallelism by specifying the maximum number of subtasks that a utility can generate.

Improved availability of previous compression dictionaries

Version 11 improves the availability of previous compression dictionaries by writing them to the log and making them available through the instrumentation facility interface (IFI) to data replication, log analysis, and similar tools and products.

With data compression, an initial compression dictionary is created during a LOAD, REORG, or SQL insert operation. In earlier versions of DB2, a REORG or LOAD REPLACE without KEEPDICTIONARY might occur and cause compression dictionaries older than two versions to be unavailable. In Version 11, DB2 can efficiently access older compression dictionaries from the log through the instrumentation facility interface (IFI).

Related concepts:

Contents of the log (DB2 Administration Guide)

Related reference:

Qualifying log records (DB2 Administration Guide)

Chapter 4. Ease of use

Version 11 provides several enhancements that improve the ease of use of DB2 for z/OS, such as support for archive tables and improvements to optimization statistics feedback and statistics collection.

Improved data validation after running DSN1COPY

To prevent abends, data corruption, and storage overlays after the DSN1COPY utility runs, DB2 Version 11 automatically validates the target data set. After a data set is populated by DSN1COPY, the first time that it is physically opened by an operation other than a utility, DB2 checks for certain data and catalog inconsistencies.

If DSN1COPY is not used correctly, subsequent attempts to access the data can result in abends, data corruption, and storage overlays. For example, the following scenarios might result in such errors:

- Incorrect DBID, PSID, or OBID values are specified when you run DSN1COPY with the OBIDXLAT option.
- DSN1COPY is used to copy data to a table space of a different type. For example, data is copied from a segmented table space to a partition-by-growth universal table space.
- DSN1COPY is used to copy data to a table space with a different version number or table schema.

In earlier versions of DB2, these data integrity errors might not be detected until a system failure occurs after the data is stored in the target table space. As a result, the data is unavailable.

Beginning in Version 11, DB2 automatically detects these inconsistencies the first time that the data set is physically opened after being populated by DSN1COPY. Specifically, DB2 checks for any data and catalog inconsistencies for the following items and reports them with a -904 SQL code:

- DBID, PSID, and OBID
- SEGSIZE and PAGESIZE
- Table space type
- Table schema (DB2 checks this item if the table space contains only one table, and an OBDREC is stored in the system page.)

Several exception situations exist. DB2 does not check for data and catalog inconsistencies during the following situations:

- The data set is physically opened by a utility, including the REPAIR utility.
- DB2 is restarting.
- The header page is not formatted yet.
- The REPAIR utility is operating on the header page. (The REPAIR utility closes the page set when it is finished. Therefore, validation can be done the next time that the data set is physically opened.)
- The LOGAPPLY phase of the RECOVER utility is processing.

By not checking for inconsistencies during these situations, DB2 limits any performance impact.

If any inconsistencies are reported, you can then correct them by using the REPAIR utility with the new CATALOG option. You can also use the REPAIR utility to proactively check for any inconsistencies after you run DSN1COPY instead of waiting for the data set to be physically opened. In this case, use the REPAIR utility with the CATALOG TEST option.

Restriction: None of these data validation enhancements apply to LOB and XML table spaces.

Related reference:

DSN1COPY (DB2 Utilities)

Syntax and options of the REPAIR control statement (DB2 Utilities)

Ability to alter limit keys for materialized query tables

In Version 11, you can alter limit keys for materialized query tables or tables that are referenced by materialized query tables.

In earlier versions of DB2, you cannot alter limit keys in materialized query tables or tables that are referenced by materialized query tables. You must unload the data, drop the table, re-create the table with the new partition limit key, and then reload the data.

In Version 11, you can change the limit key for materialized query tables or tables that are referenced by materialized query tables just as you would for other tables. The only restriction is that the table must be in a range-partitioned universal table space or in a table space that partitioned (non-universal) with table-controlled partitioning. If this requirement is met, you can use an ALTER TABLE statement to change the limit key. In most cases, changing the limit key is a pending definition change. Changing the limit key has no impact to data availability.

Related concepts:

"Improved availability when altering limit keys" on page 14

Creation of materialized query tables (Introduction to DB2 for z/OS)

Related reference:

ALTER TABLE (DB2 SQL)

Improvements to DROP PENDING CHANGES

You can remove pending table space definition changes that are not yet materialized by using the ALTER TABLESPACE statement with the DROP PENDING CHANGES clause. In Version 11, this statement now resets advisory REORG-pending (AREOR) status.

In earlier versions of DB2, this statement does not reset the AREOR status for the affected object, even though it removes the pending changes. Therefore, if you use automated tools, this remaining AREOR status might trigger unnecessary REORG utility jobs for objects that no longer have any pending changes.

In Version 11, an ALTER TABLESPACE statement with the DROP PENDING CHANGES clause resets any AREOR status. The only exception is hash tables that

are still in a transition state to become a hash table. (The table was altered from non-hash to hash without an online REORG.) In this case, AREOR status is not reset.

Related tasks:

Altering tables to enable hash access (DB2 Administration Guide)

Related reference:

REORG-pending status (DB2 Utilities)

ALTER TABLESPACE (DB2 SQL)

Support for archive tables

In Version 11, DB2 can automatically insert rows that are deleted from one table into a separate table that is called an *archive table*. Archive tables are useful for managing historical data.

You can specify that you want a table to use an archive table by specifying the new ENABLE ARCHIVE clause on the ALTER TABLE statement. The original table is called an *archive-enabled table*. You then use a global variable (SYSIBMADM.MOVE_TO_ARCHIVE) to control whether DB2 inserts rows that are deleted from this table into the associated archive table. However, when you use this global variable to turn on archiving, you cannot update the archive-enabled table.

When you query an archive-enabled table, you can indicate whether the query considers the rows in the archive table. You also use a global variable (SYSIBMADM.GET_ARCHIVE) to indicate this preference. Therefore, you can easily change the query to include or exclude archive table data without having to update the SQL.

Archive tables provide the following benefits:

- DB2 can manage historical data for you. You do not have to manually move data to a separate table.
- Because rows that are infrequently accessed are stored in a separate table, you can potentially improve the performance of queries against the archive-enabled table.
- You can modify queries to include or exclude archive table data without having to change the SQL statement and prepare the application again. Instead, you can control the scope of the query with a global variable.
- You can store archive tables on a lower-cost device to reduce operating costs.

Related concepts:

Archive-enabled tables and archive tables (Introduction to DB2 for z/OS)

Related reference:

ALTER TABLE (DB2 SQL)

References to built-in global variables (DB2 SQL)

Optimization statistics feedback

In Version 11, DB2 identifies statistics values that are missing or conflicting when it selects access paths for SQL statements. DB2 externalizes data about the missing or conflicting statistics as optimization feedback in certain catalog and EXPLAIN tables.

DB2 externalizes data about the missing or conflicting statistics to the following tables:

- The DSN_STAT_FEEDBACK table, when you capture EXPLAIN information
- The SYSIBM.SYSSTATFEEDBACK catalog table, at the specified statistics interval

You can use the feedback data to identify missing or conflicting statistics to collect. Optimization tools can also use the optimization feedback data to recommend the collection of such statistics.

Related tasks:

Identifying missing or conflicting statistics (DB2 Performance)

Related reference:

SYSIBM.SYSSTATFEEDBACK table (DB2 SQL)

DSN_STAT_FEEDBACK (DB2 Performance)

Statistics collection enhancements

In Version 11, certain utilities and commands are enhanced to improve the ease of collecting and maintaining statistics for your database objects.

Version 11 includes the following enhancements that improve statistics collection:

- When you specify the USE PROFILE keyword in a RUNSTATS utility control statement, a default profile is applied when no profile exists for a target table.
- You can use the RUNSTATS utility to reset the access path statistics to default values for a target table space.
- You can use the ACCESS DATABASE command to immediately externalize in-memory real-time statistics to the catalog tables.
- You can collect distribution statistics when you collect inline statistics when you run the following utilities:
  - LOAD
  - REORG TABLESPACE
- You can collect histogram statistics when you collect inline statistics when you run the following utilities:
  - LOAD
  - REBUILD INDEX
  - REORG INDEX

  - REORG TABLESPACE
- You can use information that DB2 externalizes during access path selection to identify missing and conflicting statistics to collect.

Related concepts:

"Optimization statistics feedback" on page 20

Related tasks:

Using RUNSTATS profiles (DB2 Utilities)

Updating real-time statistics immediately (DB2 Performance)

Collecting histogram statistics (DB2 Performance)

Resetting access path statistics (DB2 Utilities)

Related reference:

LOAD (DB2 Utilities)

RUNSTATS (DB2 Utilities)

REBUILD INDEX (DB2 Utilities)

REORG INDEX (DB2 Utilities)

REORG TABLESPACE (DB2 Utilities)

-ACCESS DATABASE (DB2) (DB2 Commands)

Chapter 5. Security and regulatory compliance

Version 11 continues to offer improvements to security and regulatory compliance. With key enhancements to exit authorization checking and program authorization, DB2 for z/OS and System z continue to lead the industry in security and auditing.

DB2 enhancements for exit authorization checking

In Version 11, DB2 provides the access control environment element (ACEE) of the package owner for authorization checking when the access control authorization exit is active. DB2 also refreshes the cache entries of the package authorization, the routine authorization, the DDF user authorization, and the dynamic statement when a user profile or resource access is changed in RACF and the access control authorization exit is active.

If the AUTHEXIT_CHECK system parameter is set to DB2, DB2 provides the ACEE of the package owner to perform authorization checking when processing the autobind, BIND, and REBIND commands. DB2 provides the ACEE of the authorization ID as determined by the DYNAMICRULES option to perform dynamic SQL authorization checking. The access control authorization exit uses the ACEE for XAPLUCHK for authorization checking. The XAPLUCHK authorization ID can be a user or a group in RACF. To ensure successful authorization checks with the owner ACEE, the owner authorization ID in XAPLUCHK must be permitted access to the resources in RACF.

When the AUTHEXIT_CACHEREFRESH system parameter is set to ALL and the access control authorization exit is active, DB2 listens to type 62, type 71, and type 79 ENF signals from RACF for user profile or resource access changes. Then, DB2 refreshes the DB2 cache entries as needed. If you define RACF classes for DB2 objects and administrative authorities without using IBM-supplied RACF resource classes, you must enable the SIGNAL=YES option for these classes in the RACF Class Descriptor Table.

The RACF access control module (DSNXRXAC) is also enhanced to support the new global variable READ (READAUTH) and WRITE (WRITEAUTH) privileges. In addition, it returns the RACLISTED classes in the new XAPLCLST field at DB2 startup, and supports authorization checking that is associated with autobind requests for user-defined functions. 

Related reference:

AUTH EXIT CACHE REFR (AUTHEXIT_CACHEREFRESH subsystem parameter) (DB2 Installation and Migration)

AUTH EXIT CHECK (AUTHEXIT_CHECK subsystem parameter) (DB2 Installation and Migration)

Authorization IDs and ACEEs (Managing Security)

DB2 enhancements for program authorization

Starting in Version 11, DB2 provides the capability to check whether an application program is authorized to use a plan.

When program authorization is enabled, it is performed in addition to package authorization.

With package authorization, if you own a plan and do not know in advance which programs or packages use the plan, you must enable the necessary authorization by binding the plan with all collections or packages that are used by any programs that execute the plan. A disadvantage to this approach is that if a user has EXECUTE authority to run the plan, the user can accidentally invoke the wrong program, or can change the program to execute a different set of packages from the intended set.

Program authorization does not have this disadvantage, because it allows a program to run only the plans that you specify.

Program authorization is enabled for a program and its plan if the following conditions are true:

- The plan is bound with the new PROGAUTH(ENABLE) option for BIND PLAN and REBIND PLAN.
- Table SYSIBM.DSNPROGAUTH contains a row for the program and the plan.

Chapter 6. New application features

Version 11 provides expanded support for SQL, XML, and temporal tables, which results in improved application performance.

Autonomous procedures

Version 11 introduces autonomous procedures that commit independently from the calling application.

Autonomous procedures execute under their own units of work, separate from the calling program. They follow the rules of the COMMIT ON RETURN YES option when they finish, without committing the work of the calling program. The calling application program controls when its own updates are committed or rolled back.

You can define autonomous procedures by specifying the AUTONOMOUS keyword when you issue a CREATE PROCEDURE statement to create a native SQL procedure, or when you issue an ALTER PROCEDURE statement to modify an existing native SQL procedure.

Related concepts:

Autonomous procedures (DB2 Application programming and SQL)

Related tasks:

Controlling autonomous procedures (DB2 Administration Guide)

Related reference:

ALTER PROCEDURE (SQL - native) (DB2 SQL)

CREATE PROCEDURE (DB2 SQL)

Support for explicit period specification on views

Version 11 extends temporal support by enabling you to specify period specifications and period clauses for views. This enhancement makes implementing temporal tables easier and improves SQL consistency across the DB2 family of products.

Now, you can specify a period specification following the name of a view in the FROM clause of a query. When you query a view that references a system-period temporal table, an application-period temporal table, or a bitemporal table, you can specify a point in time or time range for a system period (SYSTEM_TIME) or an application period (BUSINESS_TIME). With this enhancement, you experience the same behavior whether you are querying a base table or a view.

Also, you can specify a period clause following the name of a target view in an UPDATE or DELETE statement. For example, you can specify a period of time for a BUSINESS_TIME period for an update or delete operation on a view that references an application-period temporal table or a bitemporal table. As a result, you experience the same behavior for the data change operation whether the operation is on a base table or a view.

Additional enhancements help to minimize the invalidation of packages for temporal tables, even if you alter a table that is referenced in a view definition and a statement that references the view is bound in a package.

Related tasks:

Querying views that reference temporal tables (DB2 Administration Guide)

Changing data by using views that reference temporal tables (DB2 Administration Guide)

Improvements to querying temporal tables

In Version 11, you can retrieve data from temporal tables for different points in time without modifying the SQL statement. Instead of changing the query, you can now use special registers to specify different points in time and run the same query.

The two new special registers are CURRENT TEMPORAL BUSINESS_TIME and CURRENT TEMPORAL SYSTEM_TIME. Both of these special registers set timestamp values to be used in queries against temporal tables. Use the CURRENT TEMPORAL BUSINESS_TIME for queries against application-period temporal tables. Use CURRENT TEMPORAL SYSTEM_TIME for queries against system-period temporal tables. When you set one of these special registers, DB2 returns rows as if the query included one of the following period specification clauses:

- FOR SYSTEM_TIME AS OF CURRENT TEMPORAL SYSTEM_TIME
- FOR BUSINESS_TIME AS OF CURRENT TEMPORAL BUSINESS TIME

In Version 10, if you wanted to retrieve data from a temporal table for a different point time, you had to modify the query. GUPI

For example, assume the following scenario: you have three system-period temporal tables, STT1, STT2 and STT3, and you have an application that contains the following SQL statement:

```
SELECT UDF_1(STT1.C1) FROM STT1,STT2
```

The user-defined function UDF_1 consists of the following SQL statement:

```
SELECT * FROM STT3
```

Today is 2012-03-26-17.44.49.000000, and the application wants to find the result from last week.

In Version 10, you had to take the following steps:

1. Modify the SQL statement as follows:

   ```
   SELECT UDF_(STT1.C1) FROM
   STT1 FOR SYSTEM_TIME AS OF  '2012-03-19-17.44.49',
   STT2 FOR SYSTEM_TIME AS OF  '2012-03-19-17.44.49'
   ```

2. Modify the SQL in UDF_1 as follows:

   ```
   SELECT * FROM
   STT3 FOR SYSTEM_TIME AS OF  '2012-03-19-17.44.49'
   ```

3. Precompile, compile, and bind both the application and UDF_1.

In Version 11, set the CURRENT TEMPORAL SYSTEM_TIME special register before you call the application:

```
SET CURRENT TEMPORAL SYSTEM_TIME = '2012-03-19-17.44.49'
```

You do not have to change the application or UDF_1. Alternatively, if you want the active data, set the special register to NULL, which is the default value for the special register. GUPI

Related tasks:

Querying temporal tables (DB2 Administration Guide)

Related reference:

CURRENT TEMPORAL BUSINESS_TIME (DB2 SQL)

CURRENT TEMPORAL SYSTEM_TIME (DB2 SQL)

XML enhancements

Version 11 provides enhanced support for XML data.

DB2 implicitly adds a document node when it stores data with the following statements:

- INSERT
- UPDATE
- XMLDOCUMENT

DB2 issues fewer error SQLCODEs when it evaluates XPath predicate expressions with an explicit cast or an operation with an invalid value. If the XML data that is processed is filtered from the result, then processing the query continues. Examples of XPath expressions that have fewer errors include the following situations:

- When data is filtered from the result by the predicate before an invalid operation. For example, an operation calculates a division of a number by zero.
- When data is explicitly cast to an incompatible data type.

The LOAD utility performance is improved when it loads binary XML data that has been previously validated. When certain conditions are met, you might see improvements to CPU time when you are loading binary XML data that was created with the UNLOAD utility.

Related concepts:

Best practices for XML performance in DB2 (DB2 Performance)

XML support for the cross-loader function

Version 11 provides support for the XML data type for the cross-loader function.

This enhancement improves data portability by enabling you to use the LOAD utility with the INCURSOR option to move XML data from a local or remote table to a local table. In addition, for the cross-loader function, the limitation on the sum of LOB column lengths is removed.

Related tasks:

Loading data by using the cross-loader function (DB2 Utilities)

Support for Java stored procedures in a 64-bit JVM

DB2 Version 11 adds support for running Java™ stored procedures in a 64-bit JVM.

Earlier versions of DB2 can run Java stored procedures in 31-bit Java virtual machines (JVMs) only, and each JVM can run only one Java stored procedure at a time. DB2 Version 11 can concurrently run multiple Java stored procedures in 64-bit JVMs. Therefore, in Version 11, more Java stored procedures can run in a single stored procedure address space than in earlier DB2 versions.

Tip: For WLM environments that run 64-bit JVMs, set the NUMTCB parameter to 25. For WLM environments that run 31-bit JVMs, set the NUMTCB parameter to 5.

Related tasks:

Setting up the environment for Java routines (DB2 Application Programming for Java)

Support for variable-length timestamps in ODBC applications

In Version 11, ODBC applications can retrieve and update data TIMESTAMP data with up to 12 fractional digits of precision.

In earlier versions of DB2, ODBC applications can update and retrieve timestamp data with up to microseconds of precision (six decimal digits). In Version 11, ODBC applications can update and retrieve timestamp data with up to picoseconds of precision (12 decimal digits).

Related concepts:

Variable-length timestamps in ODBC applications (DB2 Programming for ODBC)

Support for arrays

Version 11 includes array support in SQL statements.

This support makes it easier to exchange long lists of values with the data server. Array support on DB2 for z/OS is compatible with the support that is offered on other members of the DB2 family of products.

With array support in Version 11 you can do the following tasks:

- Define arrays as parameters and variables for SQL routines
- Pass arrays from one procedure to another as arguments for input or output parameters
- Pass arrays to functions as parameters or from functions as return values
- Manipulate arrays, transform arrays to tables, and transform tables to arrays by using new built-in functions

Related concepts:

Array types (DB2 SQL)

Expanded support for not logging table spaces

In Version 11, you can specify the logging attributes LOGGED or NOT LOGGED when you define declared global temporary tables. The NOT LOGGED option suspends logging during insert, update, and delete activity for the table.

Because the existing logging facilities in DB2 are finely tuned, suspending logging does not necessarily improve the performance of your system. However, the ability to suspend logging improves scalability, particularly for operations that insert large volumes of data.

Related reference:

DECLARE GLOBAL TEMPORARY TABLE (DB2 SQL)

Enhancement to the LIKE predicate

In Version 11, enabling the new LIKE_BLANK_INSIGNIFICANT subsystem parameter enhances the LIKE predicate so that blanks at the end of fixed-length strings are ignored. Because of this new behavior, results for the LIKE predicate are more consistent regardless of whether the column data contains fixed-length strings or variable-length strings.

This behavior is called *LIKE blank insignificant* behavior. *LIKE blank significant* behavior, in which the blanks at the end of fixed-length strings are significant (not ignored), is the default behavior during installation or migration. For variable-length strings, blanks are significant.

Tip: After you enable the LIKE_BLANK_INSIGNIFICANT subsystem parameter, existing rows might not conform to table check constraints that contain a LIKE predicate. Consider running the CHECK DATA utility on all affected tables to find the records that do not conform to the table check constraint.

Related reference:

LIKE predicate (DB2 SQL)

LIKE BLANK INSIGNIFICANT field (LIKE_BLANK_INSIGNIFICANT subsystem parameter) (DB2 Installation and Migration)

Support for user-defined global variables

Version 11 introduces support for user-defined global variables, which you can use to share relational data between SQL statements without the need for additional application logic.

A *user-defined global variable* is a global variable that is defined by the user at the database manager level. User-defined global variables help to maximize the flexibility of a database management system by providing the following benefits:

- Faster and easier porting of applications from the databases of vendor software
- Reduced cost of ownership for DB2 for z/OS
- Easier implementation of DB2 for z/OS applications

Related reference:

CREATE VARIABLE (DB2 SQL)

Column masking improvements

In Version 11, column access control restrictions for the GROUP BY clause and the HAVING clause are removed. This improvement is beneficial when you implement row and column access control.

In certain contexts in Version 10, the semantics of the column mask can conflict with the semantics in the GROUP BY clause or the HAVING clause. When these

conflicts occur, the column mask cannot be applied for the statement and an error is returned at bind time. However, now when the GROUP BY or HAVING clauses are used, columns with column masks can be referenced in the result table without any restrictions.

Also, inconsistencies between the behavior of the SELECT DISTINCT statement and an aggregate function with the DISTINCT keyword are resolved. Now, column masking for an aggregate function with the DISTINCT keyword is applied after the removal of duplicate rows.

Related tasks:

Coding SQL statements to avoid unnecessary processing (DB2 Performance)

Eliminating redundant duplicate rows in the result table (DB2 Application programming and SQL)

Related reference:

select-clause (DB2 SQL)

Chapter 7. Migration

As with each version of DB2 for z/OS, Version 11 continues to provide substantial improvements that make migration easier and faster. Version 11 provides enhancements to the DB2 installation CLIST, ISPF panels, and jobs, and provides new installation verification procedures (IVPs).

Improvements to DB2 installation and samples

Version 11 continues to provide improvements to installing and migrating DB2 subsystems and data sharing groups. This version of DB2 for z/OS provides enhancements to the DB2 installation CLIST, ISPF panels, and jobs, and provides new installation verification procedures (IVPs).

The DB2 installation and migration jobs have the following enhancements:

- You can now specify functional IDs for running the installation, migration, and verification jobs. In installation panel DSNTIPG, you can specify the following functional IDs for use in the installation and IVP jobs:
  - A functional ID that is used as the current SQLID
  - A functional ID that is used as the OWNER value for the BIND commands
  - A functional ID to which privileges are granted on objects that are used by the installation jobs
- In installation panel DSNTIPG, you can specify an ID under which DB2-supplied routines are created and bound. You can also specify an ID that is used as the SECURITY DEFINER value for DB2-supplied routines.
- The installation CLIST now supports a maximum length of 44 bytes for DB2 target library names.
- BIND PLAN commands in the installation, migration, and IVP jobs now contain ACTION(REPLACE) RETAIN, so that EXECUTE privileges are preserved when the plans are bound again.

Related tasks:

Tailoring DB2 jobs to your environment using the installation CLIST (DB2 Installation and Migration)

Related reference:

Installation preferences panel: DSNTIPG (DB2 Installation and Migration)

Application compatibility by version

Version 11 adds the ability to set the SQL function and features available to an application package based on an application compatibility value.

In Version 11 new-function mode, you can run your application with either V10R1 or V11R1 features and functions. If you migrate your DB2 environment to Version 11 new-function mode, you can continue to run individual applications with Version 10 SQL features and functions. You can control which of your applications to run with V10R1 and which to run with V11R1. Leaving applications set to V10R1 gives you time to review differences in behavior between versions. The value of application compatibility is based on a bind option of your package, a special register, or a subsystem parameter value.

Related concepts:

Application compatibility of packages (DB2 Application programming and SQL)

Chapter 8. Command changes in Version 11

This version of DB2 for z/OS includes new and changed commands.

New commands in Version 11

This version of DB2 for z/OS includes no new commands.

Changed commands in Version 11

This version of DB2 for z/OS includes changes to some DB2 commands.

The following table shows the existing commands that have new and changed options. For information about an option with a changed default value, see Command release incompatibilities.

GUPI

Table 1. Changes to existing commands

Command	Description of enhancements and notes
-ACCESS DATABASE (DB2)	Additional option values: • MODE (STATS)
-ALTER BUFFERPOOL (DB2)	Additional options: • FRAMESIZE(4K | 1M | 2G) • VPSIZEMAX(* | *integer*) • VPSIZEMIN(* | *integer*)
-ALTER BUFFERPOOL (DB2)	Additional option values: • CLASST(*class-threshold1*,*class-threshold2*)
BIND PACKAGE and REBIND PACKAGE: APPLCOMPAT bind option (DB2 Commands) APREUSE bind option (DB2 Commands) ARCHIVESENSITIVE bind option (DB2 Commands) BUSTIMESENSITIVE bind option (DB2 Commands) DBPROTOCOL bind option (DB2 Commands) DESCSTAT bind option (DB2 Commands) SYSTIMESENSITIVE bind option (DB2 Commands)	Additional options: • APPLCOMPAT • APREUSE(WARN) • ARCHIVESENSITIVE(YES | NO) • BUSTIMESENSITIVE(YES | NO) • DBPROTOCOL(DRDACBF) • DESCSTAT(NO | YES) • SYSTIMESENSITIVE(YES | NO)
BIND PACKAGE: COPY bind option (DB2 Commands) DEPLOY bind option (DB2 Commands) LIBRARY bind option (DB2 Commands) MEMBER bind option (DB2 Commands) PACKAGE bind option (DB2 Commands)	The following option values can be delimited. If they are delimited, they can contain mixed-case characters. • *collection-id* • *dbrm-member-name* • *dbrm-library-name* • *package-id*
BIND PLAN and REBIND PLAN: PROGAUTH bind option (DB2 Commands)	Additional options: • PROGAUTH(DISABLE | ENABLE)

Table 1. Changes to existing commands (continued)

Command	Description of enhancements and notes
REBIND TRIGGER PACKAGE: APPLCOMPAT bind option (DB2 Commands) ARCHIVESENSITIVE bind option (DB2 Commands) BUSTIMESENSITIVE bind option (DB2 Commands) SYSTIMESENSITIVE bind option (DB2 Commands)	Additional options: • APPLCOMPAT • ARCHIVESENSITIVE(YES | NO) • BUSTIMESENSITIVE(YES | NO) • SYSTIMESENSITIVE(YES | NO)
-START DB2 (DB2)	Additional options: • LIGHT(CASTOUT)

GUPI

Chapter 9. Changes to utilities in Version 11

This version of DB2 for z/OS includes new and changed utilities.

New utilities in Version 11

This version of DB2 for z/OS includes no new utilities.

Utility option changes in Version 11

In this version of DB2 for z/OS, many existing utilities have new and changed options.

The following table lists and describes these new and changed options. For information about a new option, see the information for the utility. For information about an option with a changed default value, see Utility release incompatibilities.

Table 2. New and changed utility options

Utility name	Description of enhancements and notes
CHECK INDEX	**New option:** PARALLEL
LOAD	**New options:** • COLGROUP • FREQVAL COUNT • HISTOGRAM • NUMQUANTILES • NUMCOLS • PARALLEL
REBUILD INDEX	**New options:** • HISTOGRAM • NUMQUANTILES • NUMCOLS • PARALLEL
REORG INDEX	**New options:** • HISTOGRAM • NUMQUANTILES • NUMCOLS **Options with changed defaults:** • DRAIN

Table 2. New and changed utility options (continued)

Utility name	Description of enhancements and notes
REORG TABLESPACE	**New options:** • LISTPARTS • MAPPINGDATABASE • RECLUSTER • COLGROUP • PARALLEL(*num-subtasks*) **Options with changed defaults:** • DRAIN
REPAIR	**New option:** CATALOG
RUNSTATS	**New option:** RESET ACCESSPATH **Changed option:** A default profile is used when USE PROFILE is specified and no profile exists for the target table.
UNLOAD	**New option:** PARALLEL

Other utility changes in Version 11

Other than new and changed utility options, this version of DB2 for z/OS includes additional changes to utilities.

DISPLAY UTILITY output

In Version 11, the output from the DISPLAY UTILITY command includes the job name and the date and time when the job was submitted.

Related information:

-DISPLAY UTILITY (DB2) (DB2 Commands)

DSNU100I (DB2 Messages)

Utility RBA and LRSN handling

Online utility output

In all migration modes, RBA and LRSN values are displayed in 10-byte format. This 10-byte display is unrelated to migration of the catalog or directory, conversion of individual objects to EXTENDED format, or BSDS conversion. For recovery purposes, this 10-byte format is the preferred input format for DB2. When 10-byte RBA or LRSN values are specified as input to DB2 conversion to 6-byte format is performed internally as needed.

Stand-alone utility output

In all migration modes, formatted RBA and LRSN values are displayed in 10-byte format. The 10-byte formatted display is unrelated to migration of the catalog or directory, conversion of individual objects to EXTENDED

format, or BSDS conversion. Dump format data, which might contain RBA or LRSN values, is not consistently shown in 10-byte format. The format of the display represents the actual data at the time of the utility run, and is shown without conversion.

SELECT OUTPUT FROM SYSIBM.SYSLGRNX

A SELECT from SYSIBM.SYSLGRNX displays the LGRSRBA, LGREPBA, LGRSLRSN, and LGRELRSN columns in either 6-byte or 10-byte format. Before CATENFM of SYSLGRNX the data and the display are in 6-byte format. After CATENFM of SYSLGRNX the data and the display are in 10-byte format.

SELECT OUTPUT FROM SYSIBM.SYSCOPY

A SELECT from SYSIBM.SYSCOPY displays the START_RBA and PIT_RBA columns in either 6-byte or 10-byte format. Before CATENFM of SYSCOPY, the data and the display are in 6-byte format but in all migration modes in utility-output, SYSCOPY columns are displayed in 10-byte format. After CATENFM of SYSCOPY the data and the display are in 10-byte format with non-zero digits in low order 3 bytes. Digits in the low order 3 bytes are unrelated to the conversion of the BSDS or conversion of individual objects to EXTENDED format.

Chapter 10. SQL statement changes in Version 11

This version of DB2 for z/OS provides new and changed SQL statements.

New SQL statements in Version 11

This version of DB2 for z/OS includes new SQL statements.

GUPI

Table 3. New SQL statements

SQL statement	Description
CREATE TYPE (array) (DB2 SQL)	The CREATE TYPE (array) SQL statement defines an array type at the current server.
CREATE VARIABLE (DB2 SQL)	The CREATE VARIABLE statement creates a global variable at the current server.
SET CURRENT APPLICATION COMPATIBILITY (DB2 SQL)	The SET CURRENT APPLICATION COMPATIBILITY statement changes the value of the CURRENT APPLICATION COMPATIBILITY special register.
SET CURRENT TEMPORAL BUSINESS_TIME (DB2 SQL)	The SET CURRENT TEMPORAL BUSINESS_TIME statement changes the value of the CURRENT TEMPORAL BUSINESS_TIME special register.
SET CURRENT TEMPORAL SYSTEM_TIME (DB2 SQL)	The SET CURRENT TEMPORAL SYSTEM_TIME statement changes the value of the CURRENT TEMPORAL SYSTEM_TIME special register.
SET assignment-statement (DB2 SQL)	The SET *assignment-statement* statement is a reclassification of the documentation of the SET *host-variable* and SET *transition-variable* statements into a single statement.

GUPI

Changed SQL statements in Version 11

In this version of DB2 for z/OS, many existing SQL statements have new and changed clauses.

The following table shows the changes to existing SQL statements. For information about a clause with a changed default value, see Application and SQL release incompatibilities.

GUPI

Table 4. Changes to existing SQL statements

SQL statement	Description of enhancements and notes
ALTER FUNCTION (SQL scalar)	**New clauses:** **BUSINESS_TIME SENSITIVE** **SYSTEM_TIME SENSITIVE** **ARCHIVE SENSITIVE** **APPLCOMPAT** **Changed clauses:** *data-type*, *data-type2* can include *array-type-name*.
ALTER PROCEDURE (SQL native)	**New clauses:** **BUSINESS_TIME SENSITIVE** **SYSTEM_TIME SENSITIVE** **ARCHIVE SENSITIVE** **APPLCOMPAT** **Changed clauses:** *data-type* can include *array-type-name*.
ALTER TABLE	**New clauses:** **DROP COLUMN** **ENABLE ARCHIVE** **DISABLE ARCHIVE**
ALTER TABLESPACE	**Changed clauses:** PCTFREE can now include FOR UPDATE *smallint*.
COMMENT	**Changed clauses:** *data-type* can include *array-type-name*.
CREATE FUNCTION (SQL scalar)	**New clauses:** **BUSINESS_TIME SENSITIVE** **SYSTEM_TIME SENSITIVE** **ARCHIVE SENSITIVE** **APPLCOMPAT** **Changed clauses:** *data-type* can include *array-type-name*.
CREATE INDEX	**New clauses:** **INCLUDE NULL KEYS** **EXCLUDE NULL KEYS**

Table 4. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
CREATE PROCEDURE (external)	**Changed clauses:** *data-type* can include *array-type-name*.
CREATE PROCEDURE (SQL native)	**New clauses:** **BUSINESS_TIME SENSITIVE** **SYSTEM_TIME SENSITIVE** **ARCHIVE SENSITIVE** **APPLCOMPAT** **Changed clauses:** *data-type* can include *array-type-name*.
CREATE TABLESPACE	**Changed clauses:** PCTFREE can now include FOR UPDATE *smallint*.
DECLARE GLOBAL TEMPORARY TABLE	**New clauses:** **LOGGED** **NOT LOGGED**
DROP	**Changed clauses:** *data-type* can include *array-type-name*.
EXECUTE	**Changed clauses:** The object of the USING clause can be an SQL variable, SQL parameter, global variable, or host variable.
FETCH	**Changed clauses:** The object of the INTO clause can be a host variable, a global variable, an SQL parameter, an SQL variable, a transition variable, or an array element.
GRANT (function or procedure privileges)	**Changed clauses:** *data-type* can include *array-type-name*.
GRANT (type or JAR privileges)	**Changed clauses:** The object of the TYPE clause can be a distinct type or an array type.

Table 4. Changes to existing SQL statements (continued)

SQL statement	Description of enhancements and notes
OPEN	**Changed clauses:** The object of the USING clause can be an SQL variable, SQL parameter, global variable, or host variable.
REVOKE (function or procedure privileges)	**Changed clauses:** *data-type* can include *array-type-name*.
REVOKE (type or JAR privileges)	**Changed clauses:** The object of the TYPE clause can be a distinct type or an array type.
SELECT INTO	**Changed clauses:** The object of the INTO clause can be a host variable, a global variable, an SQL parameter, an SQL variable, a transition variable, or an array element.
SET PATH	**Changed clauses:** The SYSTEM PATH now includes the schemas "SYSIBM", "SYSFUN", "SYSPROC", "SYSIBMADM".
SQL statement with subselect	**Changed clauses:** *collection-derived-table* is added to *table-reference* in the FROM clause of a *subselect*.
VALUES INTO	**Changed clauses:** The object of the INTO clause can be a host variable, a global variable, an SQL parameter, an SQL variable, a transition variable, or an array element.

GUPI

New functions in Version 11

This version of DB2 for z/OS includes new built-in functions that improve the power of the SQL language.

The following table shows the new built-in functions.

GUPI

Table 5. New functions

Function name	Description
ARRAY_AGG (DB2 SQL)	The ARRAY_AGG function returns an array in which each value of the input set is assigned to an element of the array.
ARRAY_DELETE (DB2 SQL)	The ARRAY_DELETE function deletes elements from an array.
ARRAY_FIRST (DB2 SQL)	The ARRAY_FIRST function returns the minimum array index value of an array.
ARRAY_LAST (DB2 SQL)	The ARRAY_LAST function returns the maximum array index value of an array.
ARRAY_NEXT (DB2 SQL)	The ARRAY_NEXT function returns the next larger array index value, relative to a specified array index value.
ARRAY_PRIOR (DB2 SQL)	The ARRAY_PRIOR function returns the next smaller array index value, relative to a specified array index value.
CARDINALITY (DB2 SQL)	The CARDINALITY function returns the number of elements in an array.
MAX_CARDINALITY (DB2 SQL)	The MAX_CARDINALITY function returns the maximum number of elements that an array can contain.
TRIM_ARRAY (DB2 SQL)	The TRIM_ARRAY function deletes elements from the end of an ordinary array.

GUPI

Reserved words

Certain words cannot be used as ordinary identifiers in some contexts because those words might be interpreted as SQL keywords. For example, ALL cannot be a column name in a SELECT statement. Each word, however, can be used as a delimited identifier in contexts where it otherwise cannot be used as an ordinary identifier. For example, if the quotation mark (") is the escape character that begins and ends delimited identifiers, "ALL" can appear as a column name in a SELECT statement.

New reserved words for this version of DB2 for z/OS are identified with notes in this topic. In addition, some topics in this information might indicate words that cannot be used in the specific context that is being described.

IBM SQL has additional reserved words that DB2 for z/OS does not enforce. Therefore, you should not use these additional reserved words as ordinary identifiers in names that have a continuing use. See *IBM DB2 SQL Reference for Cross-Platform Development* for a list of the words.

GUPI

ADD	AND	ASUTIME
AFTER	ANY	AT
ALL	AS	AUDIT
ALLOCATE	ARRAY[1]	AUX
ALLOW	ARRAY_EXISTS[1]	AUXILIARY
ALTER	ASENSITIVE	
	ASSOCIATE	

BEFORE		
BEGIN		
BETWEEN		
BUFFERPOOL		
BY		
CALL	CLUSTER	CONTENT
CAPTURE	COLLECTION	CONTINUE
CASCADED	COLLID	CREATE
CASE	COLUMN	CURRENT
CAST	COMMENT	CURRENT_DATE
CCSID	COMMIT	CURRENT_LC_CTYPE
CHAR	CONCAT	CURRENT_PATH
CHARACTER	CONDITION	CURRENT_SCHEMA
CHECK	CONNECT	CURRENT_TIME
CLONE	CONNECTION	CURRENT_TIMESTAMP
CLOSE	CONSTRAINT	CURRVAL
	CONTAINS	CURSOR
DATA	DELETE	DO
DATABASE	DESCRIPTOR	DOCUMENT
DAY	DETERMINISTIC	DOUBLE
DAYS	DISABLE	DROP
DBINFO	DISALLOW	DSSIZE
DECLARE	DISTINCT	DYNAMIC
DEFAULT		
EDITPROC	ENDING	EXECUTE
ELSE	END-EXEC[2]	EXISTS
ELSEIF	ERASE	EXIT
ENCODING	ESCAPE	EXPLAIN
ENCRYPTION	EXCEPT	EXTERNAL
END	EXCEPTION	
FENCED	FOR	
FETCH	FREE	
FIELDPROC	FROM	
FINAL	FULL	
FIRST	FUNCTION	
GENERATED	GRANT	
GET	GROUP	
GLOBAL		
GO		
GOTO		
HANDLER		
HAVING		
HOLD		
HOUR		
HOURS		

IF	INHERIT	INSERT
IMMEDIATE	INNER	INTERSECT
IN	INOUT	INTO
INCLUSIVE	INSENSITIVE	IS
INDEX		ISOBID
		ITERATE
JAR		
JOIN		
KEEP		
KEY		
LABEL	LIKE	LOCK
LANGUAGE	LOCAL	LOCKMAX
LAST	LOCALE	LOCKSIZE
LC_CTYPE	LOCATOR	LONG
LEAVE	LOCATORS	LOOP
LEFT		
MAINTAINED	MINUTES	
MATERIALIZED	MODIFIES	
MICROSECOND	MONTH	
MICROSECONDS	MONTHS	
MINUTE		
NEXT	NOT	
NEXTVAL	NULL	
NO	NULLS	
NONE	NUMPARTS	
OBID	OPTIMIZE	
OF	OR	
OLD	ORDER	
ON	ORGANIZATION	
OPEN	OUT	
OPTIMIZATION	OUTER	
PACKAGE	PATH	PRIOR
PARAMETER	PIECESIZE	PRIQTY
PART	PERIOD	PRIVILEGES
PADDED	PLAN	PROCEDURE
PARTITION	PRECISION	PROGRAM
PARTITIONED	PREPARE	PSID
PARTITIONING	PREVVAL	PUBLIC
QUERY		
QUERYNO		

READS	RESULT_SET_LOCATOR	ROUND_DOWN
REFERENCES	RETURN	ROUND_FLOOR
REFRESH	RETURNS	ROUND_HALF_DOWN
RESIGNAL	REVOKE	ROUND_HALF_EVEN
RELEASE	RIGHT	ROUND_HALF_UP
RENAME	ROLE	ROUND_UP
REPEAT	ROLLBACK	ROW
RESTRICT	ROUND_CEILING	ROWSET
RESULT		RUN

SAVEPOINT	SET	STOGROUP
SCHEMA	SIGNAL	STORES
SCRATCHPAD	SIMPLE	STYLE
SECOND		SUMMARY
SECONDS	SOME	SYNONYM
SECQTY	SOURCE	SYSDATESYSTEM
SECURITY	SPECIFIC	SYSTIMESTAMP
SEQUENCE	STANDARD	
SELECT	STATIC	
SENSITIVE	STATEMENT	
SESSION_USER	STAY	

TABLE	TRUNCATE
TABLESPACE	TYPE
THEN	
TO	
TRIGGER	

UNDO	USER
UNION	USING
UNIQUE	
UNTIL	
UPDATE	

VALIDPROC	VCAT
VALUE	VERSIONING[1]
VALUES	VIEW
VARIABLE	VOLATILE
VARIANT	VOLUMES

WHEN
WHENEVER
WHERE
WHILE
WITH
WLM

XMLEXISTS
XMLNAMESPACES
XMLCAST

YEAR
YEARS

ZONE

Note:

1. New reserved word for Version 11.
2. COBOL only

GUPI

Chapter 11. Catalog changes in Version 11

This version of DB2 for z/OS includes changed catalog tables, new catalog tables, and new indexes on catalog tables.

New catalog tables in Version 11

This version of DB2 for z/OS includes new catalog tables.

GUPI

Table 6. New catalog tables

Catalog table name	Description
SYSIBM.SYSINDEXCLEANUP table (DB2 SQL)	The rows in the SYSIBM.SYSINDEXCLEANUP table specify time windows to control index cleanup processing. Each row specifies a time window to enable or disable the cleanup of pseudo-deleted index entries for specific database objects.
SYSIBM.SYSQUERYPREDICATE table (DB2 SQL)	The SYSIBM.SYSQUERYPREDICATE table contains information about predicates for queries in the SYSIBM.SYSQUERY table that have been identified for extended optimization. It correlates to the SYSIBM.SYSQUERY table by the QUERYID column.
SYSIBM.SYSQUERYSEL table (DB2 SQL)	The SYSIBM.SYSQUERYSEL table contains information about the selectivity of predicates for queries in the SYSIBM.SYSQUERY table that have been identified for extended optimization. It correlates to the SYSIBM.SYSQUERY table by the QUERYID column.
SYSIBM.SYSSTATFEEDBACK table (DB2 SQL)	The SYSIBM.SYSTATFEEDBACK table contains information about missing or conflicting catalog statistics for SQL statements.
SYSIBM.SYSVARIABLEAUTH table (DB2 SQL)	The SYSIBM.SYSVARIABLEAUTH table contains one row for each privilege of each authorization ID that has privileges on a global variable.
SYSIBM.SYSVARIABLES_DESC table (DB2 SQL)	The SYSIBM.SYSVARIABLES_DESC table is an auxiliary table for the SYSIBM.SYSVARIABLES table.
SYSIBM.SYSVARIABLES table (DB2 SQL)	The SYSIBM.SYSVARIABLES table contains one row for each global variable that is created.
SYSIBM.SYSVARIABLES_TEXT table (DB2 SQL)	The SYSIBM.SYSVARIABLES_TEXT table is an auxiliary table for the DEFAULTTEXT column of the SYSIBM.SYSVARIABLES table.

GUPI

Changed catalog tables in Version 11

Many existing catalog tables are changed in this version of DB2 for z/OS.

The following table shows a list of the new columns and the existing columns that are revised. Revisions to columns include new column descriptions, new values for a column, changed data types, changed column lengths, or both changed data types and lengths.

GUPI

Table 7. Summary of new and revised catalog table columns

Catalog table name	New column	Revised column
SYSIBM.SYSCHECKS table (DB2 SQL)		RBA
SYSIBM.SYSCOPY table (DB2 SQL)	MODECREATED	• PIT_RBA • START_RBA • ICTYPE • STYPE • TTYPE
SYSIBM.SYSDATATYPES table (DB2 SQL)	• ARRAYLENGTH • ARRAYINDEXTYPEID • ARRAYINDEXTYPELEN • ARRAYINDEXSUBTYPE	• LENGTH • METATYPE • SCALE • SUBTYPE
SYSIBM.SYSDEPENDENCIES table (DB2 SQL)	DVERSION	• BTYPE • DTYPE
SYSIBM.SYSINDEXES table (DB2 SQL)		• CLUSTERED • CLUSTERRATIO • COPYLRSN • SPARSE
SYSIBM.SYSINDEXES_HIST table (DB2 SQL)		• AVGKEYLEN • CLUSTERRATIOF • DATAREPEATFACTORF
SYSIBM.SYSINDEXPART table (DB2 SQL)	RBA_FORMAT	• AVGKEYLEN • CARDF • FAROFFPOSF • NEAROFFPOSF
SYSIBM.SYSINDEXPART_HIST table (DB2 SQL)		• AVGKEYLEN • CARDF • FAROFFPOSF • NEAROFFPOSF
SYSIBM.SYSINDEXSPACESTATS table (DB2 SQL)		COPYLRSN
SYSIBM.SYSINDEXSTATS table (DB2 SQL)		• CLUSTERRATIO • CLUSTERRATIOF • DATAREPEATFACTORF • KEYCOUNT • KEYCOUNTF
SYSIBM.SYSINDEXSTATS_HIST table (DB2 SQL)		• CLUSTERRATIOF • DATAREPEATFACTORF • KEYCOUNTF
SYSIBM.SYSKEYTARGETS table (DB2 SQL)	DESCRIPTOR	

Table 7. Summary of new and revised catalog table columns (continued)

Catalog table name	New column	Revised column
SYSIBM.SYSPACKAGE table (DB2 SQL)	• APPLCOMPAT • ARCHIVESENSITIVE • BUSTIMESENSITIVE • DESCSTAT • EXTSEQNO	• APREUSE • DBPROTOCOL • SYSTIMESENSITIVE
SYSIBM.SYSPACKCOPY table (DB2 SQL)	• APPLCOMPAT • ARCHIVESENSITIVE • BUSTIMESENSITIVE • DESCSTAT • EXTSEQNO • SYSTIMESENSITIVE	• DBPROTOCOL • RECORDTEMPORALHIST • SYSTIMESENSITIVE
SYSIBM.SYSPACKSTMT table (DB2 SQL)	EXPANSION_REASON	
SYSIBM.SYSPARMS table (DB2 SQL)		• CCSID • ENCODING_SCHEME • LENGTH • SUBTYPE
SYSIBM.SYSPENDINGDDL table (DB2 SQL)	• COLNAME • COLUMN_KEYWORD • PARTITION • PARTITION_KEYWORD	• CREATEDTS • OBJTYPE • OPTION_KEYWORD • OPTION_VALUE • STATEMENT_TYPE
SYSIBM.SYSPLAN table (DB2 SQL)	PROGAUTH	
SYSIBM.SYSQUERY table (DB2 SQL)	• ACCESS_PATH_HINT • OPTION_OVERRIDE • SELECTIVITY_VALID • SELECTVTY_OVERRIDE	
SYSIBM.SYSQUERYPLAN table (DB2 SQL)	EXPANSION_REASON	
SYSIBM.SYSRESAUTH table (DB2 SQL)		QUALIFIER
SYSIBM.SYSROUTINES table (DB2 SQL)		COMMIT_ON_RETURN
SYSIBM.SYSSEQUENCES table (DB2 SQL)	• SEQNAME • SEQSCHEMA	
SYSIBM.SYSTABLEPART table (DB2 SQL)	• PCTFREE_UDP • PCTRFREE_UPD_CALC • RBA_FORMAT	
SYSIBM.SYSTABLES table (DB2 SQL)	• ARCHIVING_SCHEMA • ARCHIVING_TABLE • STATS_FEEDBACK	• RBA1 • RBA2 • VERSION

Table 7. Summary of new and revised catalog table columns (continued)

Catalog table name	New column	Revised column
SYSIBM.SYSTABLESPACESTATS table (DB2 SQL)	• UPDATESIZE • LASTDATACHANGE	COPYUPDATELRSN

GUPI

New and changed indexes in Version 11

This version of DB2 for z/OS includes new indexes and changes to existing indexes.

Refer to the table in Table spaces and indexes for information about the new and changed indexes for catalog tables, as well as information about table space changes.

Chapter 12. EXPLAIN table changes in Version 11

Before you can use EXPLAIN, you must create an EXPLAIN table to hold the captured information. Certain optimization tools also create and use their own instances of the EXPLAIN tables.

PSPI

For the current formats, and sample CREATE TABLE statements for PLAN_TABLE and the other EXPLAIN tables, see member DSNTESC of the SDSNSAMP library. For the complete set of column descriptions for each table, see EXPLAIN tables (DB2 Performance) and Input tables (DB2 Performance).

The following table lists only new and changed columns for the following EXPLAIN tables:

Table 8. List of new and changed EXPLAIN tables

EXPLAIN table	Changes
PLAN_TABLE (DB2 Performance)	New columns: • SCAN_DIRECTION • EXPANSION_REASON Changed columns: • MERGN (new values) • PRIMARY_ACCESSTYPE
DSN_PREDICAT_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON Changed columns: • ADDED_PRED (new values)
DSN_PREDICATE_SELECTIVITY table (DB2 Performance)	New table.
DSN_PTASK_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_QUERYINFO_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_QUERY_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_SORTKEY_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_SORT_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_STATEMNT_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_STRUCT_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON

Table 8. List of new and changed EXPLAIN tables (continued)

EXPLAIN table	Changes
DSN_VIEWREF_TABLE (DB2 Performance)	New columns: • EXPANSION_REASON
DSN_VIRTUAL_INDEXES (DB2 Performance)	New columns: • KEYTARGET_COUNT • UNIQUE_COUNT • IX_EXTENSION_TYPE • DATAREPEATFACTORF • SPARSE
DSN_VIRTUAL_KEYTARGETS (DB2 Performance)	New table.

PSPI

Related concepts:

Investigating SQL performance by using EXPLAIN (DB2 Performance)

Input tables (DB2 Performance)

Related reference:

EXPLAIN (DB2 SQL)

EXPLAIN tables (DB2 Performance)

Chapter 13. New and changed IFCIDs in Version 11

This version of DB2 for z/OS contains a number of trace enhancements.

PSPI

This information briefly describes the new IFCIDs and the changes to the existing IFCIDs for each new function. For a detailed description of the fields in each IFCID record, refer to the mapping macros data set library DSNB10.SDSNMACS. PSPI

New IFCIDs in Version 11

This version of DB2 for z/OS includes new instrumentation facility component identifiers (IFCIDs).

PSPI

Table 9. New IFCIDs

IFCID	Trace	Class	Mapping macro	Description
0377	None	None	DSNDQW05	Records information about indexes for which pseudo-deleted index entries are automatically cleaned up.
0382	Accounting, Monitor	3, 8	DSNDQW05	Records suspend operations for parallel task synchronization.
0383	Accounting, Monitor	3, 8	DSNDQW05	Records resume operations after parallel task synchronization.

PSPI

Changed IFCIDs in Version 11

This version of DB2 for z/OS introduces changes to a number of trace records.

PSPI

Changes to selected trace records: The following table gives an overview of changes to specific IFCIDs. Changes to IFCID 0106, the system parameters record, are not included.

Table 10. Changed IFCIDs

Enhancement	IFCID	Description of changes
Array support	0002, 0225	Fields are added to record information about storage use by arrays.
Autonomous transactions	0003, 0239	Fields are added to record information about autonomous transactions, and on their effect on parallel groups.
Application compatibility	0366	Field values are added to record incompatibilities between Version 11 and previous DB2 versions.

Table 10. Changed IFCIDs (continued)

Enhancement	IFCID	Description of changes
Castout enhancements	0230, 0256	Fields are added to record class castout queue threshold values, which are based on the number of pages.
Larger RBA and LRSN support	0001, 0032, 0034, 0043, 0114, 0119, 0143, 0144, 0185, 0188, 0203, 0204, 0206, 0207, 0208, 0209, 0235, 0261, 0306, 0313, 0335	Fields that contain RBAs and LRSNs are expanded from 6 or 8 bytes to 10 bytes. Fields other than QW0204UR are moved to avoid changes in other offsets. Field QW0204UR is expanded but not moved. Therefore, the offsets of all fields in IFCID 0204 that follow it are changed.
Parallelism performance enhancements	0002, 0003, 0316, 0401	Fields are added to track the effect of changes to the degree of parallelism after parallel system negotiation that occurs because of resource constraints.
Temporal support	0053, 0058, 0060, 0061, 0065, 0316, 0401	Fields are added to indicate the impacts of the CURRENT TEMPORAL BUSINESS_TIME special register, the CURRENT TEMPORAL SYSTEM_TIME special register, and the SYSIBMADM.GET_ARCHIVE built-in global variable.
Miscellaneous changes	0002	A field is added to record the amount of storage that is allocated to shareable, static SQL statements.
	0002, 0003	Fields are added to record the number of pages that are written to disk through group-buffer-pool write-around protocol.
	0003	• Fields are added to record the wait time and number of waits for parallel queries to synchronize. • Existing fields that record accumulated wait time for write I/O that is done under another thread and the number of wait trace events that were processed for write I/O under another thread now include buffer manager force write class 3 accounting time.
	0003, 0148, 0239	Sysplex query parallelism is no longer supported. Fields that are related to Sysplex query parallelism are not populated.
	0127, 0128	Fields are added to record the time spent waiting for buffer manager force write I/O.
	0148	A field is added to record an additional DDF enclave.
	0201	Fields are added to record the virtual pool size, minimum pool size, maximum pool size, and frame size before and after the ALTER BUFFERPOOL command.
	0217, 0225	Fields are added to record IRLM private storage limits.
	0225	The following changes are made: • Fields are added to record the common storage that is used by log manager buffers and control structures. • Data is no longer recorded for the number of log manager write buffer frames in auxiliary storage. • Fields that record shareable storage use for SQL statements are moved and expanded to 8 bytes. The corresponding 4-byte fields are no longer used.
	0239	• Package detail information is rolled up and associated with a rollup QPAC record if an accounting class 10 trace or a monitor class 10 trace is enabled. • Fields are added to record the wait time and number of waits for parallel queries to synchronize.

Table 10. Changed IFCIDs (continued)

Enhancement	IFCID	Description of changes
	0313	New fields are added to record: • User IDs that are longer than 16 bytes • Transaction names that are longer than 32 bytes • Workstation names that are longer than 18 bytes

Chapter 14. New and changed subsystem parameters in Version 11

This version of DB2 for z/OS includes new and changed subsystem parameters.

New subsystem parameters in Version 11

This version of DB2 for z/OS includes several new subsystem parameters.

GUPI

Table 11. New subsystem parameters

Subsystem Parameter	Description
APPL COMPAT LEVEL field (APPLCOMPAT subsystem parameter) (DB2 Installation and Migration)	Determines the default value for application compatibility.
AUTH EXIT CHECK (AUTHEXIT_CHECK subsystem parameter) (DB2 Installation and Migration)	Specifies whether the owner or the primary authorization ID is to be used for authorization checks, when the access control authorization exit is active.
AUTH EXIT CACHE REFR (AUTHEXIT_CACHEREFRESH subsystem parameter) (DB2 Installation and Migration)	Specifies whether the package authorization cache, routine authorization cache, and dynamic statement cache entries are to be refreshed, when access control authorization exit is active and the user profile is changed in RACF.
INDEX CLEANUP THREADS field (INDEX_CLEANUP_THREADS subsystem parameter) (DB2 Installation and Migration)	Specifies the maximum number of threads that can be created to process the cleanup of pseudo-deleted index entries on this subsystem or data sharing member. Pseudo-deleted entries in an index are those that are logically deleted but still physically present in the index. This parameter works in conjunction with the SYSIBM.SYSINDEXCLEANUP catalog table, which controls cleanup processing of pseudo-deleted index entries.
LIKE BLANK INSIGNIFICANT field (LIKE_BLANK_INSIGNIFICANT subsystem parameter) (DB2 Installation and Migration)	Specifies whether the LIKE predicate should ignore trailing blanks within fixed length character strings.
MAX IN-MEMORY SORT SIZE field (MAXSORT_IN_MEMORY subsystem parameter) (DB2 Installation and Migration)	Specifies the maximum size in kilobytes of storage to allocate for sorting the results of each query that specifies the ORDER BY clause, the GROUP BY clause, or both. The storage is allocated only during the query processing. Increasing the value in this field can improve performance of such queries.

GUPI

Related reference:

"Changed subsystem parameters in Version 11"

Directory of subsystem parameters and application default values (DB2 Installation and Migration)

Directory of panel field names (DB2 Installation and Migration)

Changed subsystem parameters in Version 11

This version of DB2 for z/OS includes changes to some subsystem parameters.

The following table shows the existing subsystem parameters that have new and changed options. For information about an option with a changed default value, see Other release incompatibilities.

GUPI

Table 12. Changes to existing subsystem parameters

Subsystem parameter	Description of enhancements and notes
DSMAX field (DSMAX subsystem parameter) (DB2 Installation and Migration)	The upper limit for the maximum number of data sets that can be open at one time is changed from 100,000 to 200,000.
EDM DBD CACHE field (EDMDBDC subsystem parameter) (DB2 Installation and Migration)	The upper limit for the minimum size (in KB) of the DBD cache that is to be used by EDM is changed from 2,097,152 to 4,194,304.
EDM STATEMENT CACHE field (EDMSTMTC subsystem parameter) (DB2 Installation and Migration)	The upper limit for the size (in KB) of the statement cache that is to be used by the EDM is changed from 1,048,576 to 4,194,304.
MAX KEPT DYN STMTS field (MAXKEEPD subsystem parameter) (DB2 Installation and Migration)	The upper limit for the total number of prepared, dynamic SQL statements that can be saved past a commit point by all threads in the system using the KEEPDYNAMIC(EXECUTABLE) bind option is changed is changed from 65,535 to 204,800.
EDM SKELETON POOL SIZE field (EDM_SKELETON_POOL subsystem parameter) (DB2 Installation and Migration)	The upper limit of the minimum size of the EDM skeleton pool (in KB) is changed from 2,097,152 to 4,194,304.

Removed subsystem parameters in Version 11

The following tables lists subsystem parameters that are removed from this version of DB2 for z/OS. Refer to the information for the earlier version for detailed descriptions of the removed subsystem parameters.

Table 13. Changes to existing subsystem parameters

Subsystem parameter	Setting used in Version 11 behavior
ASSIST	NO
COORDNTR	NO
DSABSCL	NO
MSVGP and MSVGP2	These devices are not recognized in Version 11.
OJPERFEH	YES

Table 13. Changes to existing subsystem parameters (continued)

Subsystem parameter	Setting used in Version 11 behavior
OTPIOWGT	ENABLE
PTCDIO	OFF
RETVLCFK	NO
SMSDCFL	blank
SMSDCIZ	blank
STATCLUS	ENHANCED
SEQCACH	SEQ
SEQPRES	YES

GUPI

Related reference:

"New subsystem parameters in Version 11" on page 59

Directory of subsystem parameters and application default values (DB2 Installation and Migration)

Directory of panel field names (DB2 Installation and Migration)

Information resources for DB2 for z/OS and related products

Information about DB2 for z/OS and products that you might use in conjunction with DB2 for z/OS is available in online information centers or on library websites.

Obtaining DB2 for z/OS publications

The current DB2 for z/OS publications are available from the following website:

http://pic.dhe.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db2z11.doc/src/alltoc/db2z_lib.htm

Links to the information center version and the PDF version of each publication are provided.

DB2 for z/OS publications are also available for download from the IBM Publications Center (http://www.ibm.com/shop/publications/order).

In addition, books for DB2 for z/OS are available on a CD-ROM that is included with your product shipment:

- DB2 11 for z/OS Licensed Library Collection, LK5T-8882, in English. The CD-ROM contains the collection of books for DB2 11 for z/OS in PDF format. Periodically, IBM refreshes the books on subsequent editions of this CD-ROM.

Installable information center

You can download or order an installable version of the Information Management Software for z/OS Solutions Information Center, which includes information about DB2 for z/OS, QMF™, IMS, and many DB2 and IMS Tools products. You can install this information center on a local system or on an intranet server. For more information, see http://pic.dhe.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.dzic.doc/installabledzic.htm.

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.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, 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.

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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Programming interface information

This information is intended to help you to learn about and plan to use DB2 11 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 11 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 11 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.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at http://www.ibm.com/privacy and IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM

Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Glossary

The glossary is available in the Information Management Software for z/OS Solutions Information Center.

See the Glossary topic for definitions of DB2 for z/OS terms.

Index

Numerics

A

B

C

D

E

F

G

I

J

K

L

M

N

O

P

Q

Product Number: 5615-DB2
5697-P43

Printed in USA

GC19-4068-00

Spine information:

DB2 11 for z/OS

What's New?