

DB2 10 for z/OS

*Data Sharing: Planning and  
Administration*

**IBM**



DB2 10 for z/OS

*Data Sharing: Planning and  
Administration*

**IBM**

**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.

**Fourth edition (May 2011)**

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

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

© Copyright IBM Corporation 1994, 2011.

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

---

# Contents

<b>About this information . . . . .</b>	<b>vii</b>
Who should read this information. . . . .	vii
DB2 Utilities Suite . . . . .	vii
Terminology and citations . . . . .	viii
Accessibility features for DB2 10 for z/OS. . . . .	viii
How to send your comments . . . . .	ix
<b>Chapter 1. Introduction to DB2 data sharing . . . . .</b>	<b>1</b>
Advantages of DB2 data sharing. . . . .	1
Improved data availability. . . . .	2
Extended processing capacity. . . . .	2
Configuration flexibility . . . . .	4
Higher transaction rates . . . . .	8
How DB2 protects data consistency. . . . .	8
How an update happens . . . . .	9
How DB2 writes changed data to disk . . . . .	13
Implications of enabling DB2 data sharing . . . . .	13
Connecting to a data sharing group . . . . .	14
Database administration for data sharing . . . . .	14
Options that affect data sharing performance . . . . .	15
Commands for data sharing . . . . .	15
Data recovery in data sharing environments . . . . .	16
Maintenance of data sharing groups . . . . .	17
<b>Chapter 2. Planning for DB2 data sharing. . . . .</b>	<b>19</b>
<b>Chapter 3. Installing, migrating, and enabling DB2 data sharing . . . . .</b>	<b>21</b>
<b>Chapter 4. Consolidating data sharing members. . . . .</b>	<b>23</b>
Potential configuration changes when you consolidate data sharing members . . . . .	25
<b>Chapter 5. Removing members from the data sharing group . . . . .</b>	<b>29</b>
What data sets to keep when you quiesce a data sharing member . . . . .	29
Quiescing a data sharing member . . . . .	29
Deleting data sharing members. . . . .	30
Deactivating data sharing members . . . . .	30
Destroying data sharing members . . . . .	31
Restoring deactivated data sharing members . . . . .	32
<b>Chapter 6. Communicating with data sharing groups. . . . .</b>	<b>33</b>
Ways to access data sharing groups . . . . .	33
TCP/IP access methods . . . . .	34
Group access . . . . .	35
DNS network addressing. . . . .	41
Member-specific access . . . . .	42
Single-member access . . . . .	46
Setting up DB2 for z/OS as a requester . . . . .	47
Configuring data sharing groups as TCP/IP servers . . . . .	49
Connecting distributed partners in a TCP/IP network . . . . .	54
SNA access methods . . . . .	59
Member-routing access . . . . .	60
Group-generic access . . . . .	61
Single-member access . . . . .	61
Setting up DB2 for z/OS as a requester . . . . .	62

Configuring data sharing groups as servers . . . . .	65
Connecting distributed partners in an SNA network . . . . .	69
Preventing a member from processing requests . . . . .	73
Update the BSDS with the DSNJU003 utility . . . . .	74

**Chapter 7. Operating with data sharing . . . . . 77**

Commands for data sharing environments . . . . .	77
Command routing . . . . .	77
Command scope . . . . .	77
Commands issued from application programs . . . . .	78
Command authorization . . . . .	78
Where messages are received . . . . .	78
Effect of data sharing on sequence number caching . . . . .	78
Starting a data sharing member . . . . .	79
Stopping a data sharing member . . . . .	79
States of connections and structures after stopping DB2 . . . . .	79
Normal shutdown . . . . .	80
Abnormal shutdown . . . . .	80
Submitting work to be processed . . . . .	80
Group attachment names and subgroup attachment names . . . . .	80
CICS and IMS applications with DB2 data sharing . . . . .	81
Online utility jobs in data sharing environments . . . . .	82
Stand-alone utility jobs . . . . .	83
Monitoring the group . . . . .	83
Obtaining information about the group . . . . .	83
Obtaining information about structures . . . . .	84
Obtaining information about group buffer pools . . . . .	86
Database monitoring options . . . . .	87
Determining the data sharing member on which SQL statements run . . . . .	90
Controlling connections to remote systems in a data sharing environment . . . . .	90
Starting and stopping DDF . . . . .	90
Monitoring connections to remote systems . . . . .	91
Resetting generic LU information . . . . .	91
Logging environment for data sharing . . . . .	92
The impact of archiving logs in a data sharing group . . . . .	92
How to avoid using the archive log . . . . .	93
Recovering data . . . . .	94
How recovery works in a data sharing group . . . . .	94
Improving recovery performance . . . . .	96
Recovery options for data sharing environments . . . . .	97
System-level point-in-time recovery . . . . .	98
Recovering a data sharing group in case of a disaster . . . . .	98
Recovery of pages on the logical page list . . . . .	102
Recovery from coupling facility failures . . . . .	102
Coupling facility recovery scenarios . . . . .	108
Resolution of transaction manager indoubt units of recovery . . . . .	117
Restarting DB2 after termination in a data sharing environment . . . . .	119
Normal restart for a data sharing member . . . . .	120
Restart light . . . . .	122
Group restart phases . . . . .	123
Protection of retained locks: failed-persistent connections . . . . .	127
Handling coupling facility connections that hang . . . . .	128
Postponed backout in a data sharing environment . . . . .	130
Restarting a member with conditions . . . . .	132
Deferring recovery during restart . . . . .	133
Starting duplexing for a structure . . . . .	133
Stopping duplexing for a structure . . . . .	134
Shutting down the coupling facility . . . . .	135

**Chapter 8. Performance monitoring and tuning . . . . . 137**

Monitoring tools . . . . .	137
Resource Measurement Facility reports . . . . .	137
DB2 trace. . . . .	137
DB2 Performance Expert . . . . .	138
Improving the performance of data sharing applications . . . . .	138
DB2 address spaces involved in distributed data processing. . . . .	139
Migration of batch applications . . . . .	140
Resource limit facility implications for data sharing . . . . .	141
Removal of group buffer pool dependency . . . . .	141
Physical open of a page set of partition. . . . .	141
Improving the response time for read-only queries . . . . .	142
Planning for Sysplex query parallelism . . . . .	142
Enabling parallel processing within an application . . . . .	148
Enabling parallel processing within a data sharing group. . . . .	148
Monitoring and tuning parallel queries. . . . .	149
Disabling Sysplex query parallelism . . . . .	156
Improving concurrency . . . . .	158
Global transaction locking . . . . .	158
Tuning your use of locks . . . . .	161
Deadlock detection and resource timeouts in data sharing environments . . . . .	167
Ways to monitor DB2 locking activity . . . . .	170
Changing the size of the lock structure . . . . .	176
Tuning group buffer pools . . . . .	179
Assigning page sets to group buffer pools. . . . .	179
Inter-DB2 interest and GBP-dependency . . . . .	181
Physical locks in data sharing . . . . .	186
Read operations . . . . .	190
Write operations . . . . .	193
Group buffer pool thresholds . . . . .	200
Ways to monitor group buffer pools. . . . .	204
Determining the correct size and ratio of group buffer pools . . . . .	209
Changing group buffer pools . . . . .	220
Access path selection in a data sharing group . . . . .	224
Effect of member configuration on access path selection . . . . .	224
How EXPLAIN works in a data sharing group . . . . .	224
How DB2 maintains in-memory statistics in data sharing. . . . .	224
<b>Information resources for DB2 for z/OS and related products . . . . .</b>	<b>227</b>
<b>How to obtain DB2 information. . . . .</b>	<b>233</b>
<b>How to use the DB2 library . . . . .</b>	<b>235</b>
<b>Notices . . . . .</b>	<b>239</b>
Programming Interface Information . . . . .	240
Trademarks . . . . .	241
<b>Glossary . . . . .</b>	<b>243</b>
<b>Index . . . . .</b>	<b>289</b>



---

## About this information

This information is the main source of information about using DB2<sup>®</sup> data sharing. You can use the information to learn about DB2 data sharing and to do many of the tasks that are associated with DB2 data sharing.

However, there are many tasks that are associated with DB2 data sharing, especially those of setting up the hardware and software environment for the Parallel Sysplex<sup>®</sup>, that require the use of other product libraries, such as z/OS<sup>®</sup>. If you are installing DB2 and plan to use data sharing capabilities, use the *DB2 for z/OS Installation and Migration Guide* to do initial planning and develop your installation strategy. You can find detailed installation procedures in the *DB2 for z/OS Installation and Migration Guide*.

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

---

## Who should read this information

This information is primarily intended for system and database administrators who are responsible for planning and implementing DB2 data sharing. Many of the task descriptions in this information assume that the user is already familiar with administering DB2 in a non-DB2 data sharing environment. See *DB2 Administration Guide* for any concepts not explained in this information.

---

## DB2 Utilities Suite

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

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

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

These informational APARs are periodically updated.

### Related information

DB2 utilities packaging (Utility Guide)

---

## Terminology and citations

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

The following terms are used as indicated:

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

**OMEGAMON®**

Refers to any of the following products:

- IBM® Tivoli® OMEGAMON XE for DB2 Performance Expert on z/OS
- IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS
- IBM DB2 Performance Expert for Multiplatforms and Workgroups
- IBM DB2 Buffer Pool Analyzer for z/OS

**C, C++, and C language**

Represent the C or C++ programming language.

**CICS®** Represents CICS Transaction Server for z/OS.

**IMS™** Represents the IMS Database Manager or IMS Transaction Manager.

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

**RACF®**

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

---

## Accessibility features for DB2 10 for z/OS

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

### Accessibility features

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

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

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

### Keyboard navigation

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

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

## Related accessibility information

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

## IBM and accessibility

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

---

## How to send your comments

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

- Send your comments by email to [db2zinfo@us.ibm.com](mailto: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 send comments from the web. Visit the DB2 for z/OS - Technical Resources website at:

<http://www.ibm.com/support/docview.wss?rs=64&uid=swg27011656>

This website has an online reader comment form that you can use to send comments.

- You can also send comments by using the **Feedback** link at the footer of each page in the Information Management Software for z/OS Solutions Information Center at <http://publib.boulder.ibm.com/infocenter/imzic>.



---

## Chapter 1. Introduction to DB2 data sharing

The data sharing function of DB2 for z/OS enables multiple applications to read from, and write to, the same DB2 data concurrently.

The applications can run on different DB2 subsystems residing on multiple central processor complexes (CPCs) in a Parallel Sysplex. A *Sysplex* is a group of z/OS systems that communicate and cooperate with one another using specialized hardware and software. They are connected and synchronized through a Sysplex Timer<sup>®</sup> and enterprise systems connection channels. A *Parallel Sysplex* is a Sysplex that uses one or more coupling facilities, which provide high-speed caching, list processing, and lock processing for any applications on the Sysplex.

A collection of one or more DB2 subsystems that share DB2 data is called a *data sharing group*. DB2 subsystems that access shared DB2 data must belong to a data sharing group.

A DB2 subsystem that belongs to a data sharing group is a *member* of that group. Each member can belong to one, and only one, data sharing group. All members of a data sharing group share the same DB2 catalog and directory, and all members must reside in the same Parallel Sysplex. Currently, the maximum number of members in a data sharing group is 32.

All members of a data sharing group can read and update the same DB2 data simultaneously. Therefore, all data that different members of the group can access must reside on shared disks.

Some capabilities described in this information can be used in both data sharing and non-data sharing environments. This information uses the term *data sharing environment* to describe a situation in which a data sharing group has been defined with at least one member. In a non-data sharing environment, no group is defined.

---

### Advantages of DB2 data sharing

DB2 data sharing improves the availability of DB2 data, extends the processing capacity of your system, provides more flexible ways to configure your environment, and increases transaction rates.

You do not need to change the SQL in your applications to use data sharing, although you might need to do some tuning for optimal performance. Because DB2 data sharing does not affect the application interface, it does not create additional work for application programmers and end users. However, system programmers, operators, and database administrators must perform additional tasks in a data sharing environment.

#### Leaves application interface unchanged

Your investment in people and skills is protected because existing SQL interfaces and attachments remain intact when sharing data.

You can bind a package or plan on one member of a data sharing group and run that package or plan on any other member of the group.

## Improved data availability

More users demand access to DB2 data every hour, every day. Data sharing helps you meet your service objectives by improving data availability during both planned and unplanned outages.

Because data sharing provides multiple paths to data, a member can be down, and applications can still access the data through other members of the data sharing group. As the following figure illustrates, when an outage occurs and one member is down, transaction managers are informed that the member is unavailable, and they can direct new application requests to another member of the group.

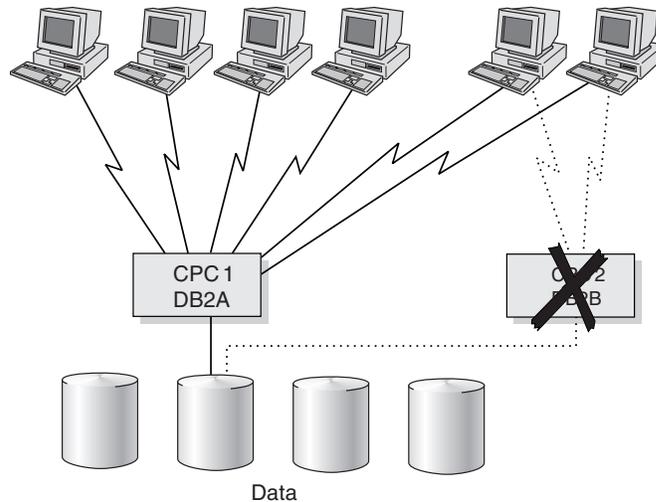


Figure 1. Data sharing improves data availability during outages. If one member or an entire central processor complex (CPC) is down, work can be routed to another member.

While increasing data availability has some performance cost, the overhead for interprocessor communication and caching changed data is minimal. DB2 provides efficient locking and caching mechanisms and uses coupling facility hardware. A *coupling facility* is a special logical partition that runs the coupling facility control program. It provides high-speed caching, list processing, and locking functions in a Parallel Sysplex. The DB2 structures in the coupling facility benefit from high availability.

## Extended processing capacity

As you move more data processing onto DB2, your processing needs can exceed the capacity of a single system. Using data sharing can help meet your ever-increasing capacity needs.

### Without DB2 data sharing

Without DB2 data sharing, you have the following options for addressing increased capacity needs:

- Copy the data, or split the data between separate DB2 subsystems.  
This approach requires that you maintain separate copies of the data. There is no communication between or among DB2 subsystems, and no shared DB2 catalog or directory.
- Install another DB2 subsystem and rewrite applications to access the data as distributed data.

This approach might relieve the workload on the original DB2 subsystem, but it requires changes to your applications and has performance overhead of its own. Nevertheless, if DB2 subsystems are separated by great distance or DB2 needs to share data with another system, the distributed data facility is still your only option.

- Install a larger processor and move the data and applications to that machine. This option can be expensive. In addition, this approach demands that your system come down while you move to the new machine.

## With DB2 data sharing

With DB2 data sharing, you get the following benefits:

**Support for incremental growth:** A Parallel Sysplex can grow incrementally, allowing you to add processing power in granular units and in a non-disruptive manner. The coupling technology of Parallel Sysplex along with the additional CPU power results in more throughput for users' applications. You no longer need to manage multiple copies of data, and all members of the data sharing group share a single DB2 catalog and directory.

**Workload balancing:** DB2 data sharing provides workload balancing so that when the workload increases or you add a new member to the group, you do not need to distribute your data or rewrite your applications. DB2 data sharing is unlike the partitioned-data approach to parallelism (sometimes called *shared-nothing* architecture), in which a one-to-one relationship exists between a database management system (DBMS) and a segment of data. When you add a new DB2 subsystem onto another central processor complex (CPC) in a data sharing environment, applications can access the same data through the new member just as easily as through any of the existing members.

DB2 works closely with the Workload Manager (WLM) component of z/OS to ensure that incoming requests are optimally balanced across the members of a data sharing group. All members of the data sharing group have the same concurrent and direct read-write access to the data.

**Capacity when you need it:** A data sharing configuration can handle peak work loads (such as end-of-quarter processing) well. You can have data sharing members in reserve, bring them online to handle peak loads, and then stop them when the peak passes.

## More capacity to process complex queries

**Sysplex query parallelism** enables DB2 to use all the processing power of the data sharing group to process a single query. For complex data analysis or decision support, Sysplex query parallelism is a scalable solution. Because the data sharing group can grow, you can put more power behind queries even as those queries become increasingly complex and run on larger and larger sets of data.

The following figure shows that all members of a data sharing group can participate in processing a single query.

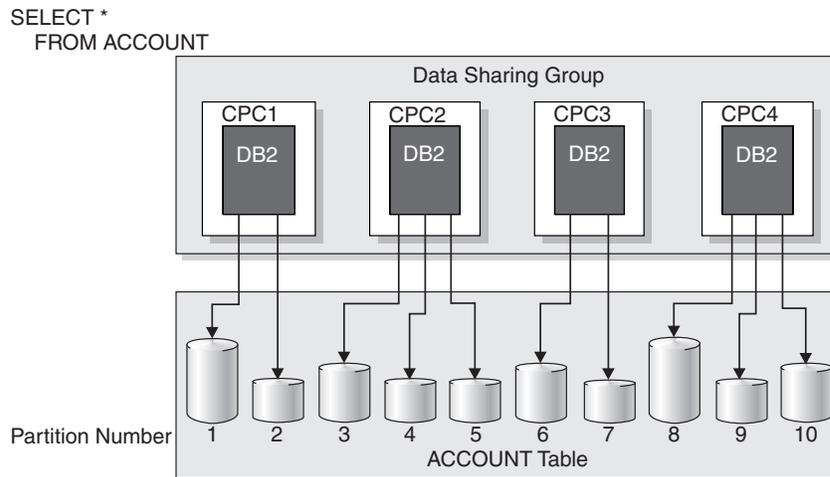


Figure 2. Query processed in parallel by members of a data sharing group. Different members process different partitions of the data.

The figure above is a simplification of the concept—several members can access the same physical partition. To take full advantage of parallelism, use partitioned table spaces.

## Configuration flexibility

Data sharing lets you configure your system environment much more flexibly with operational systems, decision support systems, and shared data management.

As the following figure shows, you can have more than one data sharing group on the same Parallel Sysplex. You might, for example, want one group for testing and another group for production data. This example also shows a single, non-data sharing DB2 subsystem.

## z/OS Parallel Sysplex

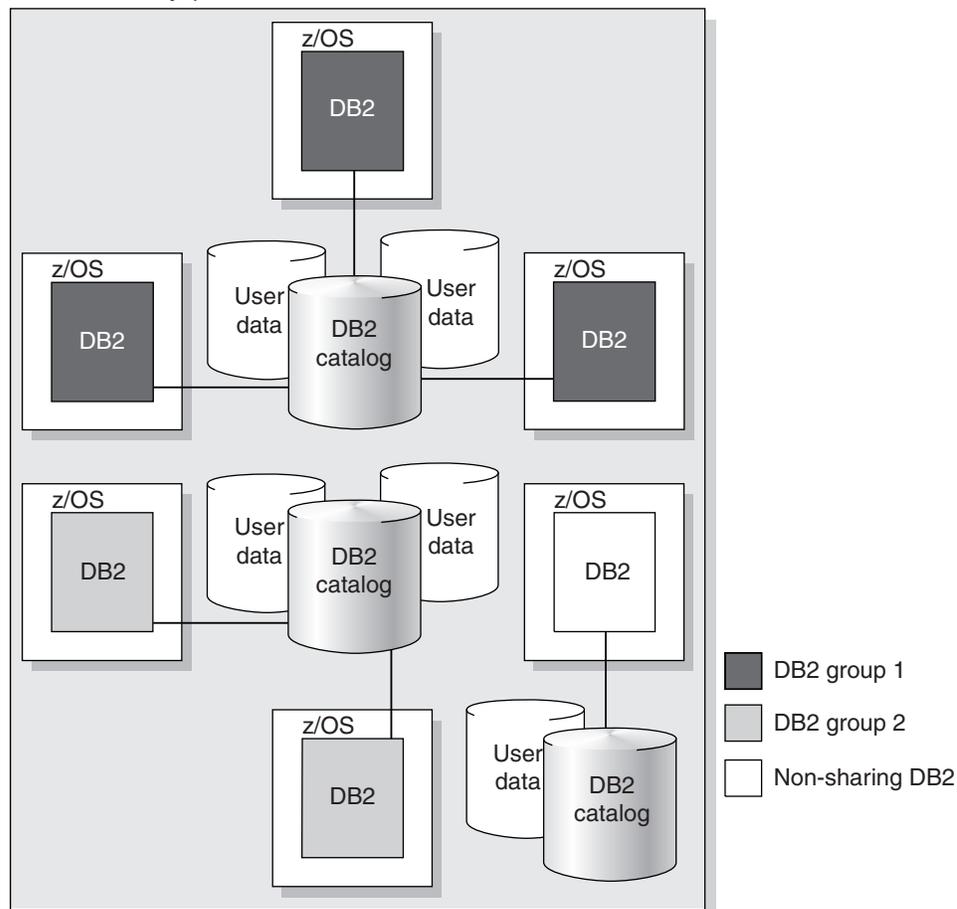


Figure 3. A possible configuration of data sharing groups. Although this example shows one DB2 subsystem per z/OS, a z/OS image can support multiple DB2 subsystems.

## Flexible operational systems

The following figure shows how, with data sharing, you can have query user groups and online transaction user groups on separate z/OS images. This configuration lets you tailor each system specifically for that user set, control storage contention, and provide predictable levels of service for that set of users. Previously, you might have had to manage separate copies of data to meet the needs of different user groups.

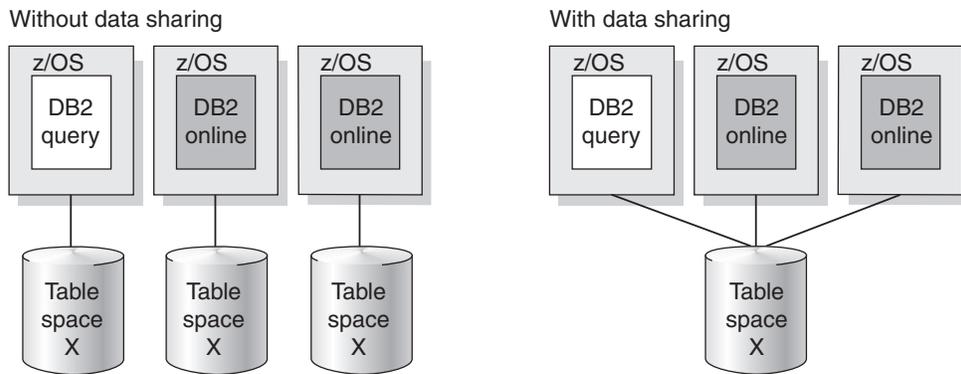


Figure 4. Flexible configurations with data sharing. Data sharing lets each set of users access the same data, which means that you no longer need to manage copies.

## Flexible decision support systems

The following figure shows two different decision support configurations. A *typical* configuration separates the operational data from the decision support data. Use this configuration when the operational system has environmental requirements that are different from those of the decision support system. For example, the decision support system might be in a different geographical area, or security requirements might be different for the two systems.

DB2 offers another option—a *combination* configuration. A combination configuration groups your operational and decision support systems into a single data sharing group and offers the following advantages:

- You can occasionally join decision support data and operational data using SQL.
- You can reconfigure the system dynamically to handle fluctuating workloads. (You can dedicate CPCs to decision support processing or operational processing at different times of the day or year.)
- You can reduce the cost of computing:
  - The infrastructure used for data management is already in place.
  - You can create a prototype of a decision support system in your existing system, and then add processing capacity as the system grows.

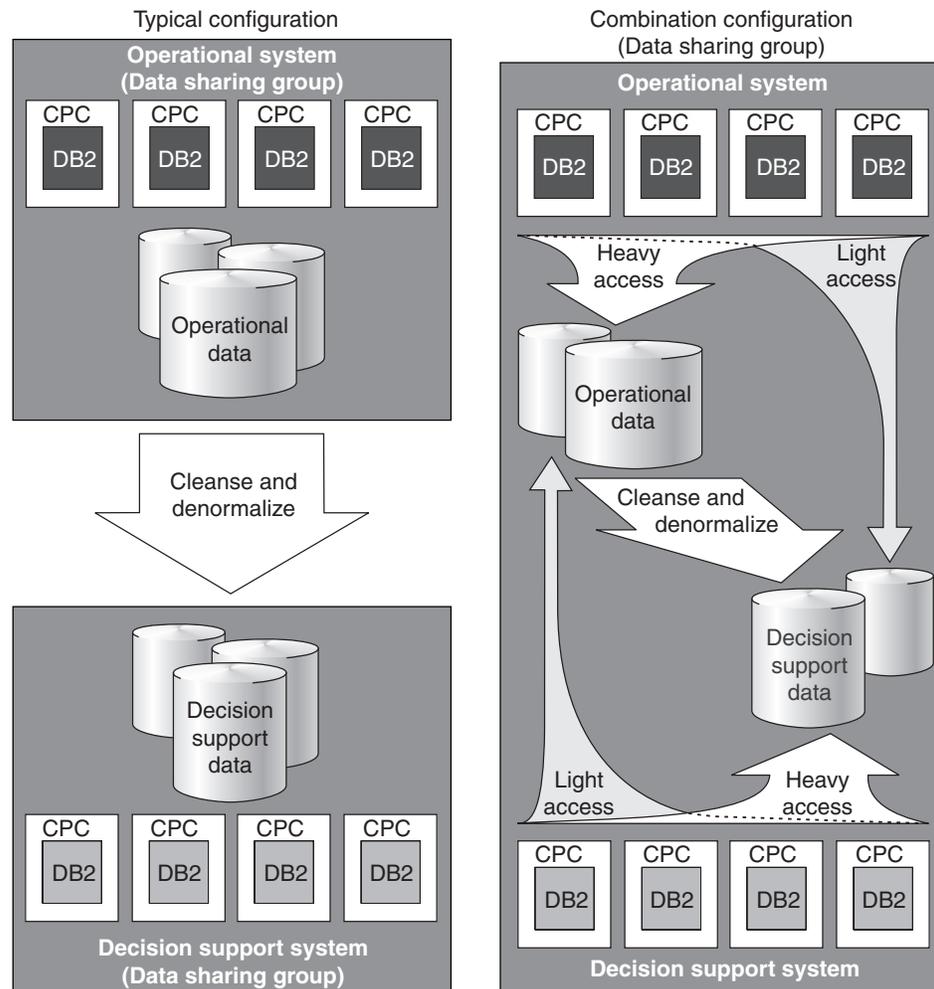


Figure 5. Flexible configurations for decision support. Data sharing lets you configure your systems in the way that works best in your environment.

If you want a combination system configuration, you must separate decision support data from operational data as much as possible. Buffer pools, disks, and control units that you use to decide on a support system should be separate from those that you use in your operational system. This separation greatly minimizes any negative performance impact on the operational system.

If you are unable to maintain the needed level of separation, or if you have separated your operational data for other reasons, such as security, using a separate decision support system is your best option.

## Flexible shared data management

DB2 data sharing can simplify the management of applications that must share data, such as a common customer table. Perhaps these applications were split in the past for capacity or availability reasons. But with the split architecture, the shared data must be synchronized across multiple systems (that is, by replicating data).

DB2 data sharing gives you the flexibility to configure these applications to access a single data sharing group. It also allows you to maintain a single copy of the shared data that can be read and updated by multiple systems with good

performance. This is an especially powerful option when data centers are consolidated.

## Higher transaction rates

Data sharing gives you opportunities to put more work through the system to produce higher transaction rates.

As the following figure illustrates, you can run the same application on more than one member to achieve transaction rates that are higher than possible on a single DB2 subsystem.

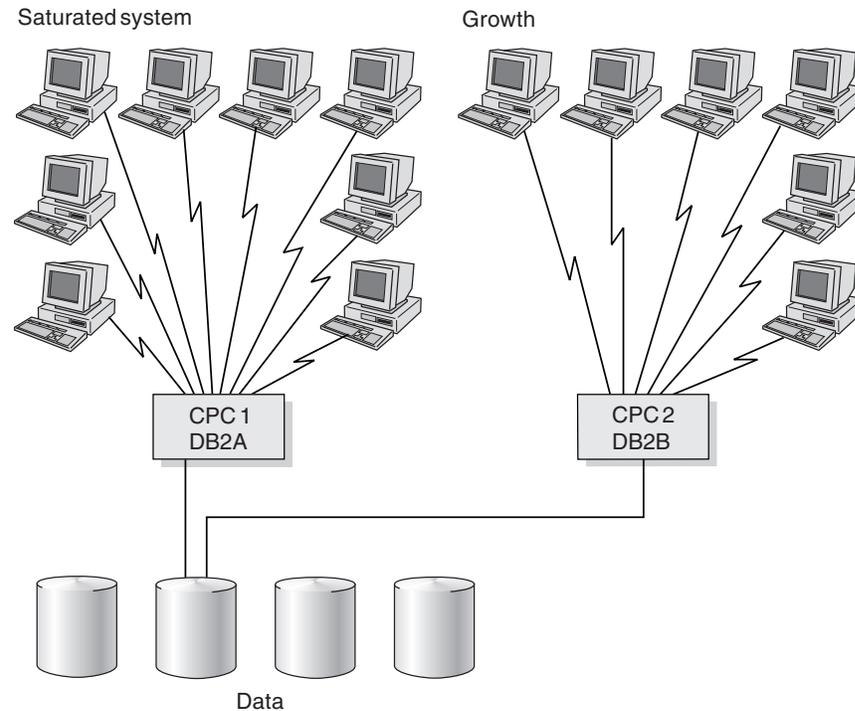


Figure 6. Data sharing enables growth. You can move some of your existing DB2 workload onto another central processor complex (CPC).

---

## How DB2 protects data consistency

Applications can access data from any member of a data sharing group, and many members can potentially read and write the same data. DB2 for z/OS uses special data sharing locking and caching mechanisms to ensure data consistency across the applications.

When multiple members of a data sharing group open the same table space, index space, or partition, and at least one of them opens it for writing, the data is said to be of inter-DB2 read/write interest to the members. (Sometimes called *inter-DB2 interest*.) To control access to data that is of inter-DB2 interest, whenever the data is changed, DB2 caches it in a storage area that is called a *group buffer pool*.

When there is inter-DB2 read/write interest in a particular table space, index, or partition, it is dependent on the group buffer pool, or *GBP-dependent* (group buffer pool-dependent).

You define group buffer pools by using coupling facility resource management (CFRM) policies.

As shown in the following figure, a mapping exists between a group buffer pool and the buffer pools of the group members. For example, each member has a buffer pool named BP0. For data sharing, you must define a group buffer pool (GBP0) in the coupling facility that maps to each member's buffer pool BP0. GBP0 is used for caching the DB2 catalog and directory table spaces and indexes, and any other table spaces, indexes, or partitions that use buffer pool 0.

Although a single group buffer pool cannot reside in more than one coupling facility (unless it is duplexed), you can put group buffer pools in more than one coupling facility.

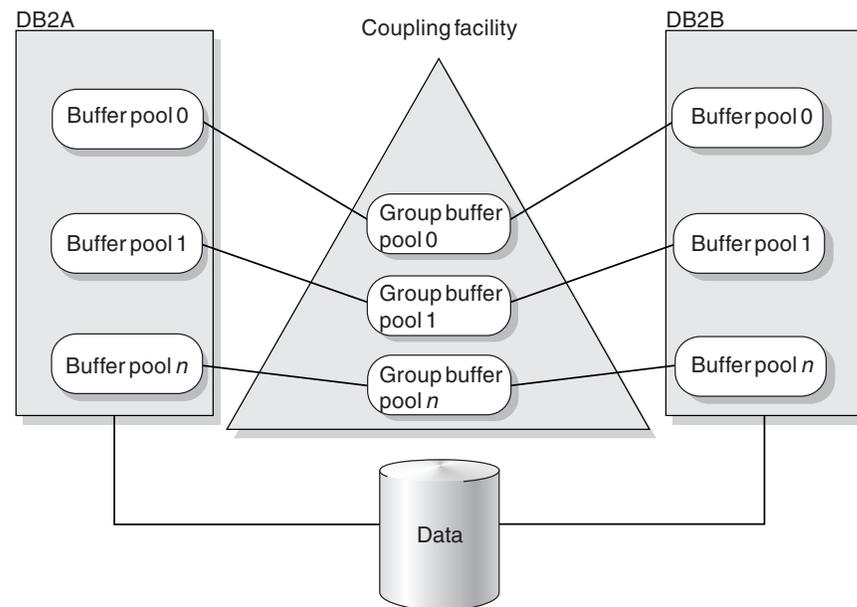


Figure 7. Relationship of member buffer pools to the group buffer pool. One group buffer pool supports all member buffer pools of the same name.

When you change a particular page of data, DB2 caches that page in the group buffer pool. The coupling facility invalidates any image of the page in the buffer pools associated with each member. Then, when a request for that same data is subsequently made by another member, that member looks for the data in the group buffer pool.

**Performance options to fit your application's needs:** By default, DB2 caches updated data, but you also have the options of caching all or none of your data. There is even an option especially for large object (LOB) data.

#### Related reference

[Updating a CFRM Policy\(z/OS MVS Setting Up a Sysplex\)](#)

---

## How an update happens

When you are deciding whether to use data sharing, it can help to understand how data is updated in a data sharing environment.

In the following illustrations, the most recent version of the data page is shaded. This scenario assumes that the group buffer pool is used for caching changed data that is duplexed for high availability. *Duplexing* is the ability to write data to two instances of a structure: in this case, a primary group buffer pool and a secondary group buffer pool.

Suppose, as shown in the following figure, that an application issues an UPDATE statement from DB2A. The data that is being updated does not already reside in either the member's own buffer pool or the group buffer pool; therefore, DB2A retrieves the data from disk and updates the data in its own buffer pool. Simultaneously, DB2A gets the appropriate locks to prevent another member from updating the same data at the same time. After the application commits the UPDATE, DB2A releases the corresponding locks. The changed data page remains in DB2A's buffer pool.

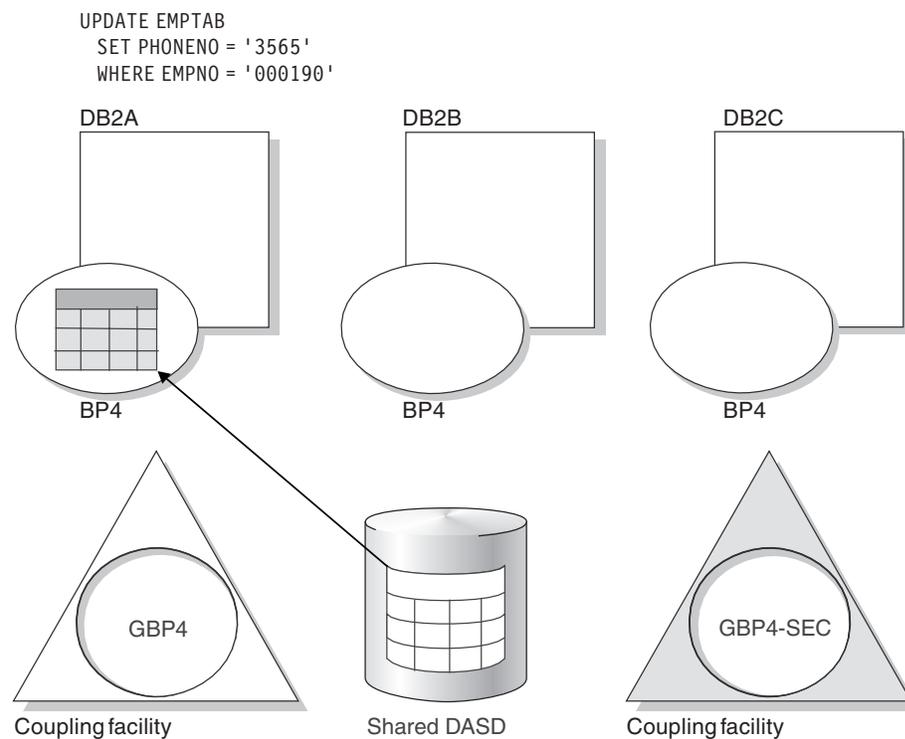
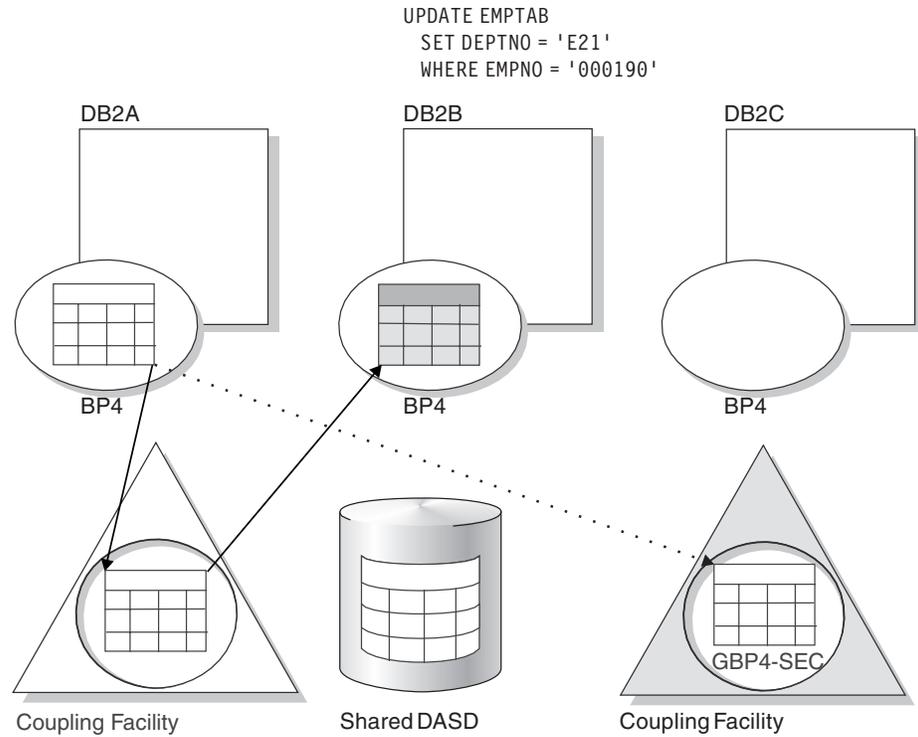


Figure 8. An application running on DB2A reads data from disk and updates it

Next, suppose another application, that runs on DB2B, needs to update the same data page as shown in the following figure. DB2 dynamically detects inter-DB2 interest in the page set, so DB2A writes the changed data page to the group buffer pools (both primary and secondary). DB2B then retrieves the data page from the primary group buffer pool.



*Figure 9. DB2B updates the same data page. When DB2B references the page set, it gets the most current version of the data from the primary group buffer pool.*

After the application that runs on DB2B commits the UPDATE, DB2B moves a copy of the data page into the group buffer pools. This invalidates the data page in DB2A's buffer pool as shown in the following figure. Cross-invalidation occurs from the primary group buffer pool.

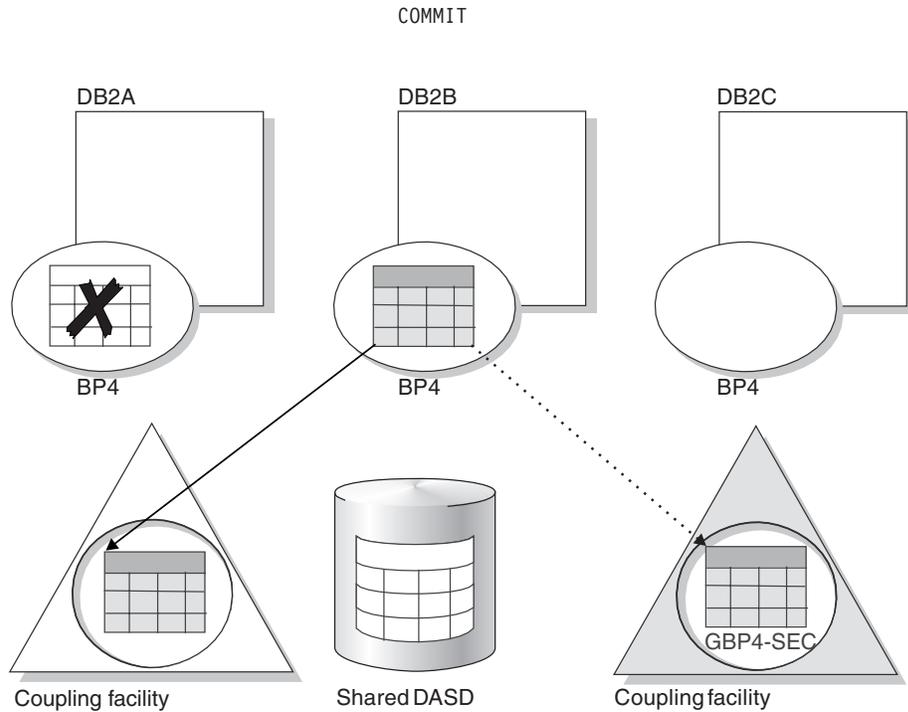


Figure 10. The updated data page is written to the group buffer pools and the data page in DB2A's buffer pool is invalidated.

Now, as shown in the following figure, when DB2A attempts to reread the data, it detects that the data page in its own buffer pool is invalid. Therefore, it reads the latest copy of the data from the primary group buffer pool.

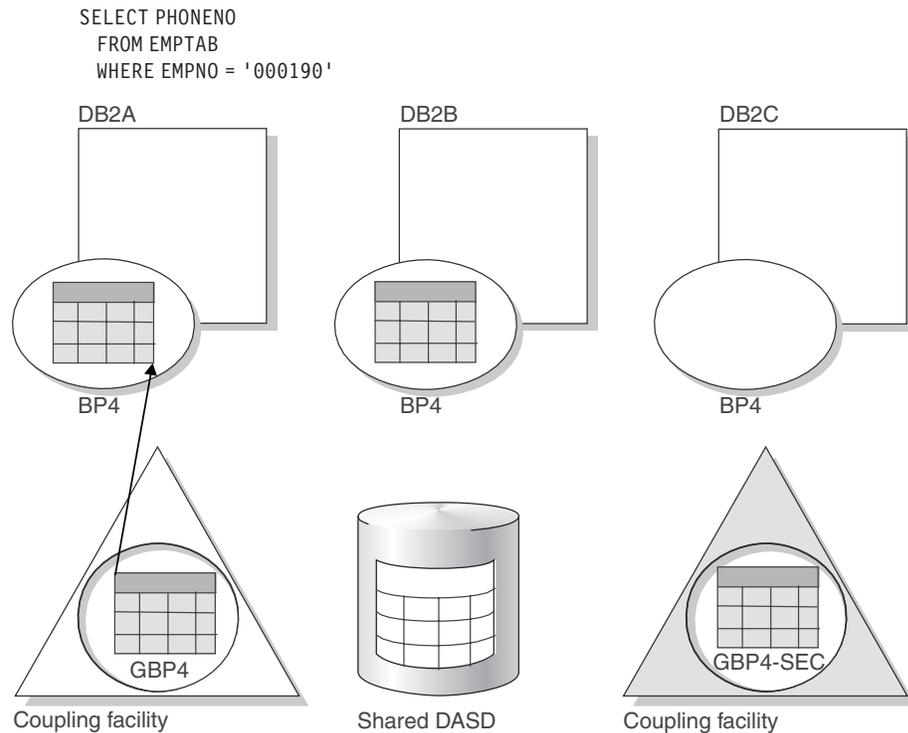
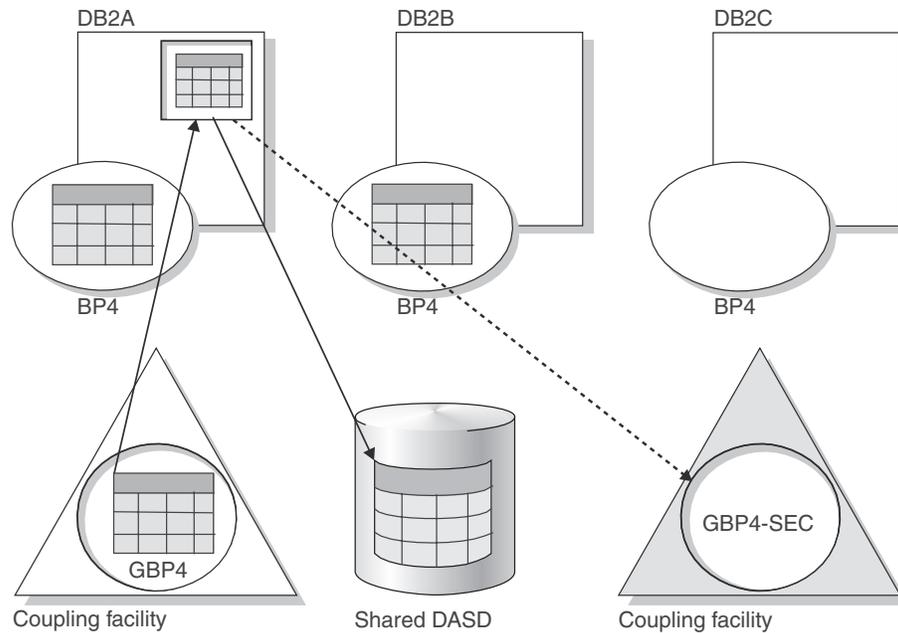


Figure 11. DB2A reads the data from the primary group buffer pool

## How DB2 writes changed data to disk

Periodically, DB2 must write changed pages from the primary group buffer pool to disk. This process is called *castout*.

The member that is responsible for casting out the changed data uses its own address space because no direct connection exists from a coupling facility to disk, as shown in the following figure. The data passes through a private buffer, not through the DB2 buffer pools.



*Figure 12. Writing data to disk.* No direct connection exists between the coupling facility and the disk. The data must pass through the address space of DB2A before being written to disk.

When a group buffer pool is duplexed, data is not cast out from the secondary group buffer pool to disk. After a set of pages is written to disk from the primary group buffer pool, DB2 deletes those pages from the secondary group buffer pool.

---

## Implications of enabling DB2 data sharing

You must plan a naming convention before enabling data sharing on the first member (the originating member) of the group.

Because names in the Parallel Sysplex and names in the data sharing group must be unique, you must have a naming convention before you create the group. Not only must shared data objects have unique names, but you must create a unique name for every group resource.

The originating member's DB2 catalog becomes the catalog for all the members of the data sharing group. You add additional members to the group as new installations, and those members use the originating member's DB2 catalog.

If you have data from existing DB2 subsystems that you want the group to share, you must merge the catalog definitions for that data into the group catalog. You

must also ensure that all members of the group can access the data. DB2 does not provide a way to merge members' catalogs automatically.

#### **Related concepts**

 [Data sharing naming conventions \(DB2 Installation and Migration\)](#)

---

## **Connecting to a data sharing group**

Applications can communicate with a data sharing group by using either Transmission Control Protocol/Internet Protocol (TCP/IP) or Systems Network Architecture (SNA) protocol.

Applications connect to a data sharing group by specifying a DB2 location name. The group provides a single-system image to requesting applications.

#### **Related concepts**

Chapter 6, “Communicating with data sharing groups,” on page 33

---

## **Database administration for data sharing**

Using data sharing has implications for planning exit routines, authorizing users, and loading and reorganizing data.

Because the DB2 catalog is shared by all members of a data sharing group, data definition, authorization, and control are the same as for non–data sharing environments. Be sure that every object has a unique name, and be sure that the shared data resides on shared disks.

### **Planning for exit routines**

If you use exit routines, such as a field or validation procedure or the access control authorization routine, ensure that all members of the group use the same routines.

**Recommendation:** Place all exit routines in a program library that is shared by all members of the group.

### **Authorizing users**

Use the same authorization mechanisms that are in place for non–data sharing DB2 subsystems to control access to shared DB2 data and to members. Because all members in the group share the same DB2 catalog, an authorization ID has the same granted privileges and authorities for every member of the group.

As suggested for non–data sharing DB2 subsystems, use a security system outside of DB2 (such as RACF<sup>®</sup> or its equivalent) to control which user IDs can access which members. RACF, for example, does not recognize a data sharing group as a single resource. Therefore, you must separately define DB2 resources to RACF for each member of the group, and connect all user IDs to a RACF group that permits access to all those resources. Or you can permit separate groups of user IDs to access different sets of resources. (In the latter case, however, you cannot move work freely among all members of the data sharing group.)

Each member of a data sharing group uses the same names for the connection and sign-on exit routines. As a good practice, all members of a group should share the same exit routines. Sharing avoids authorization anomalies such as:

- Primary authorization IDs that are treated differently by different members of the group
- Primary authorization IDs that are associated with different sets of secondary IDs by different members of the group

## Loading and reorganizing data

You can load or reorganize data from any member of a data sharing group.

### Related reference

 [LOAD \(DB2 Utilities\)](#)

 [REORG INDEX \(DB2 Utilities\)](#)

 [REORG TABLESPACE \(DB2 Utilities\)](#)

## Options that affect data sharing performance

The GBPCACHE, MEMBER CLUSTER, and TRACKMOD options all affect the performance of data sharing.

### GBPCACHE option

Use the GBPCACHE option when you create or alter table spaces or indexes to specify what data, if any, should be cached in the group buffer pool. Valid values for this option are NONE, SYSTEM, CHANGED, and ALL. The default is CHANGED.

### MEMBER CLUSTER option

Use the MEMBER CLUSTER option when you create table spaces to specify that DB2 locate data in the table space based on available space rather than clustering the data by the implicit or explicit clustering index.

This option can benefit applications when there are many inserts to the same table from multiple members.

### TRACKMOD option

Use the TRACKMOD option when you create or alter table spaces to specify whether you want DB2 to track modified pages in the space map pages of the table space or partition.

TRACKMOD NO can benefit applications when there are frequent updates from multiple members. Be aware that this option can degrade incremental image-copy performance; therefore, specify NO only if you never use incremental copies, or if you use DFSMS™ concurrent copies and LOGONLY recovery. In these cases, choosing TRACKMOD NO can improve transaction performance.

### Related concepts

“Options for reducing space map page contention” on page 189

---

## Commands for data sharing

Parallel Sysplex technology lets you manage a data sharing group from a console that is attached to a single z/OS system or from consoles that are attached to multiple z/OS systems.

The following figure shows how commands are issued from a single z/OS system.

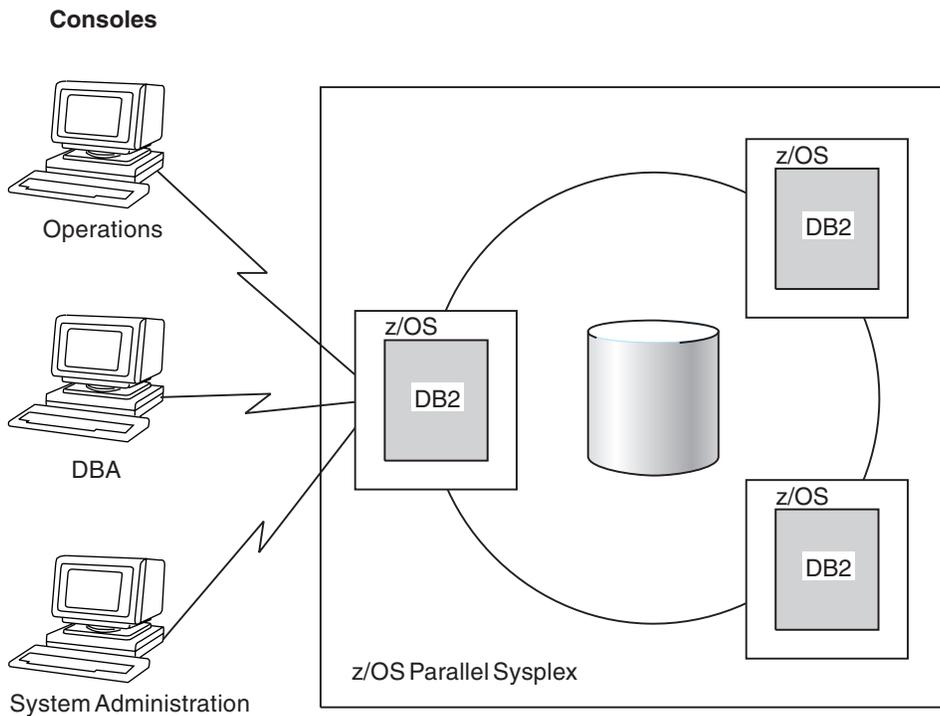


Figure 13. Issuing commands

Some commands manage group resources; others manage member resources.

#### Related information

 DB2 and related commands (DB2 Commands)

---

## Data recovery in data sharing environments

DB2 recovers data from information that is contained in both the logs and the bootstrap data sets (BSDSs) of members. However, because updates can be logged on several different members, DB2 coordinates recovery by using the SCA in a coupling facility.

The *shared communications area* (SCA) contains:

- Member names
- BSDS names
- Database exception status conditions about objects and members in the group
- Recovery information, such as log data set names and the list of indoubt XA transactions

The SCA is also used to coordinate startup.

You can stop and start an individual member of a data sharing group while the other members continue to run. The startup process for each member is similar to that of non-data sharing DB2 subsystems.

DB2 uses a process called *group restart* in the rare event that critical resources in a coupling facility are lost and cannot be rebuilt. When this happens, all members of the group terminate abnormally. Group restart rebuilds the lost information from

individual member logs. However, unlike data recovery, this information can be applied in any order. Because there is no need to merge log records, DB2 can perform many of the restart phases for individual members in parallel.

**Recommendation:** Use an automated procedure to restart failed members of the group.

### The RECOVER utility

You can run the RECOVER utility from any member of a data sharing group. The process for data recovery is basically the same for a data sharing group as it is for non-data sharing DB2 subsystems. However, updates to a single table space can be the work of several different members. Therefore, to recover an object, DB2 must merge log records from the appropriate members, using a *log record sequence number* (LRSN). The LRSN is a value derived from the store clock timestamp and synchronized across the members of a data sharing group by the Sysplex Timer.

**Recommendation:** Use more than one coupling facility to allow for structure duplexing and for automatic recovery in the event that a coupling facility fails.

### System-level point-in-time recovery

You can perform system-level point-in-time recovery by using the BACKUP SYSTEM and RESTORE SYSTEM online utilities.

#### BACKUP SYSTEM online utility

This utility provides fast, volume-level copies of DB2 databases and logs. It automatically keeps track of which volumes need to be copied. Using BACKUP SYSTEM is less disruptive than using SET LOG SUSPEND in copy procedures because the log write latch is not taken. An advantage for data sharing is that BACKUP SYSTEM has group-scope, whereas SET LOG SUSPEND has only member scope.

#### RESTORE SYSTEM online utility

This utility provides a way to recover a data sharing group to a specific point in time. RESTORE SYSTEM automatically handles any CREATE, DROP, and LOG NO events that might have occurred between the backup and the recovery point in time.

#### Related concepts

“Recovering data” on page 94

“Tuning group buffer pools” on page 179

 Coupling facility availability (DB2 Installation and Migration)

---

## Maintenance of data sharing groups

To apply maintenance, you can make most changes on one member at a time.

If you must take DB2, IRLM, or z/OS offline for a change to take place and the outage is unacceptable to users, you can move those users onto another member. Some sites find it useful to define an extra member that they bring up and down as needed to take on work while maintenance is being applied to another member.

The recommended way of testing maintenance is to apply that maintenance to a test data sharing group before moving it onto the production data sharing group.

Table 1. Actions required for planned maintenance changes

Type of change	Action required
Early code	Issue the -REFRESH DB2,EARLY command.
DB2 code	Bring down and restart each member independently.
IRLM code	Bring down and restart each IRLM member independently.
Attachment code	Apply the change and restart the transaction manager or application.
Subsystem parameters	For those that cannot be changed dynamically, update using the DB2 update process. Stop and restart the member to activate the updated parameter.

**Recommendation:** Consider specifying CASTOUT(NO) when you stop an individual member of a data sharing group for maintenance. This option speeds up shutdown because DB2 bypasses castout and associated cleanup processing in the group buffer pools.

Do not specify CASTOUT(NO) when you stop multiple members of a data sharing group and you need to maintain consistent data on disk. For example, if you stop all members to get a consistent copy of the databases on disk that you can send offsite, do not specify CASTOUT(NO) because some of the changed data could still reside in the group buffer pools after the members shut down. Your disk copy might not have all the most recent data.

**Tip:** Consider specifying a value of NODISCON for the IRLM procedure's SCOPE option to allow IRLM to continue doing work while you apply maintenance to DB2. (If you edit the IRLM procedure using the DB2 installation process, this is analogous to specifying NO for parameter DISCONNECT IRLM on installation panel DSNTIPJ.)

**Applying maintenance to IRLM:** Each member of a data sharing group has its own IRLM. As with DB2, you must stop and restart each IRLM in the group to roll a change throughout the group.

IRLM has a *function level* to control changes that affect the entire group. The function level for a particular IRLM member indicates the latest function that IRLM is capable of running. The *group function level* is the lowest function level that exists among the IRLMs in the group. The group function level is the function level at which all IRLMs in the group must run, even if some IRLM members are capable of running at higher function levels.

**Recommendation:** Keep all members of the IRLM group at the same function level. This ensures that all IRLMs are running with the same maintenance that affects the entire group. (Maintenance that does not affect the group does not increment the function level.)

To see the IRLM function levels, use the MODIFY *irlmproc,STATUS,ALLI* command of z/OS.

#### Related tasks

➡ Determining the function level of an IRLM group in coexistence (DB2 Installation and Migration)

#### Related reference

➡ DISCONNECT IRLM field (DB2 Installation and Migration)

---

## Chapter 2. Planning for DB2 data sharing

Information about installing, migrating, and enabling DB2 data sharing has been moved to the *DB2 for z/OS Installation and Migration Guide*.

See Preparing for DB2 data sharing (DB2 Installation and Migration).



---

## Chapter 3. Installing, migrating, and enabling DB2 data sharing

Information about installing, migrating, and enabling DB2 data sharing has been moved to the *DB2 for z/OS Installation and Migration Guide*.

See Installing, migrating, and enabling DB2 data sharing (DB2 Installation and Migration).



## Chapter 4. Consolidating data sharing members

Because DB2 Version 10 can provide significant virtual storage constraint relief, consider consolidating your data sharing groups so that they have fewer members. Especially consider consolidation if you originally added data sharing members to provide virtual storage constraint relief.

Starting with DB2 Version 10, nearly all thread-related storage is above the bar in the DBM1 address space, which can provide significant virtual storage constraint relief. If you want to get this virtual storage constraint relief for static SQL, the general recommendation is to rebind your packages after migrating to DB2 Version 10. However, some of this storage for static SQL is moved above the bar immediately after migrating to DB2 Version 10, before you rebind any packages. For dynamic SQL, the virtual storage constraint relief happens automatically when the statements are prepared.

Because of this location of the thread storage, you can potentially run more concurrent active threads for each DB2 subsystem or member. For data sharing configurations, this change might provide the opportunity to consolidate to fewer members.

For example, suppose that you have eight data sharing members in Version 9, each of which is running 500 active threads, as shown in the following figure.

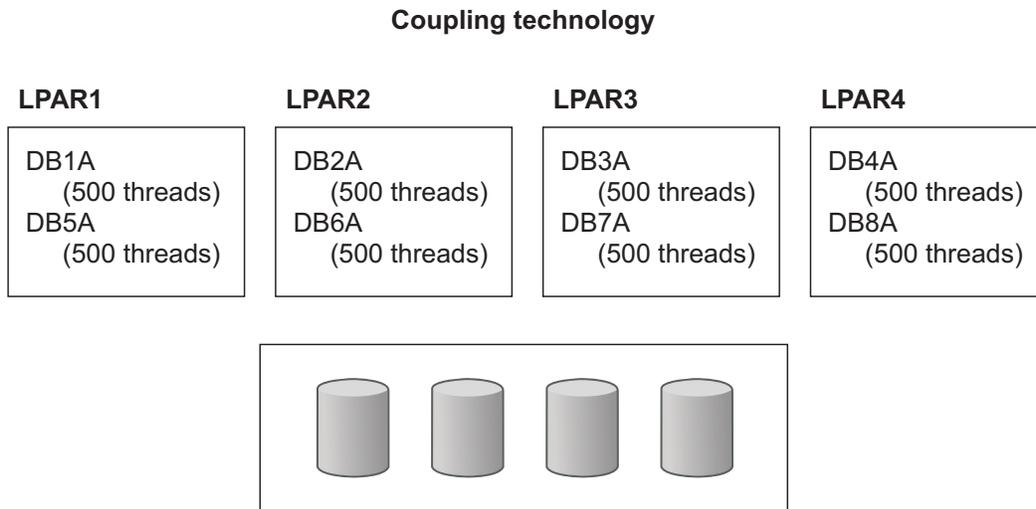


Figure 14. Data sharing configuration prior to consolidation

With the changes in Version 10, you might be able to consolidate to 4 members, each of which is running 2500 active threads, as shown in the following figure.

## Coupling technology

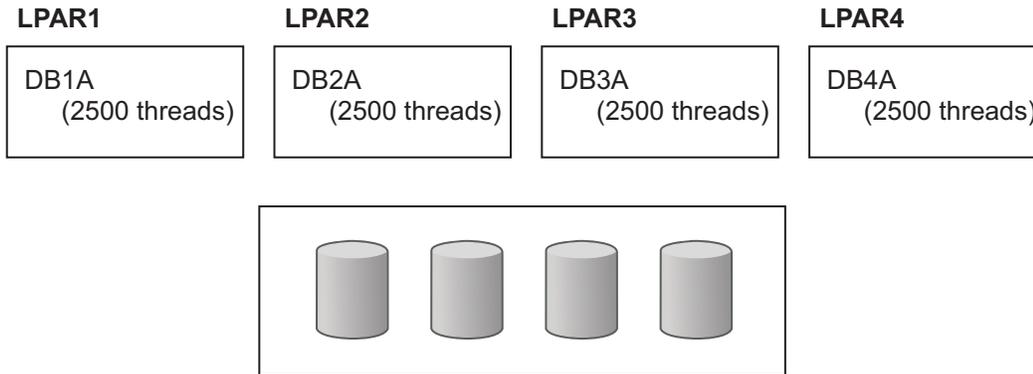


Figure 15. Data sharing configuration after consolidation

### Recommendations:

- In configuring for high availability, ensure that enough spare capacity (such as CPU and memory) is available to absorb the work in case of a member outage (planned or unplanned). To accomplish this goal, one common method is to configure a 4-way data sharing group with 2 central processor complexes (CPCs) and 2 LPARs per CPC.
- Keep the members spread across more than one LPAR.
- To help with workload balancing, consider removing the same number of members from each LPAR. For example, suppose that you have four LPARs, each with two members, and you want to consolidate to fewer members. Ideally in this situation, you would remove one member from each LPAR (instead of changing LPAR 2 and LPAR 4 to have one member and leaving the other LPARs with two members). Such a symmetrical reduction can help you with real storage and CPU planning. Otherwise, more time planning is probably required to achieve the correct workload balance.

To consolidate data sharing members:

1. Consider potential configuration changes that you might need to make when you consolidate data sharing members, and plan for these changes.
2. Redistribute the group workload:
  - a. Move the work from one of the members that you plan to remove to one or more of the members that you plan to keep.
  - b. After you move the work, monitor all members. If no issues arise, proceed to the next step. Otherwise, route any work back to the member that originally had that work, and handle any issues that arose during monitoring.
  - c. Repeat steps a and b until all work is moved from the members that plan to remove to the members that you plan to keep.

**Example:** Suppose that you have 8 members (member\_1 through member\_8) and you are consolidating to 4 members. In this scenario, assume that you plan to keep member\_1 through member\_4 and remove member\_5 through member\_8. First, move the work from member\_5 to member\_1 through member\_4. Monitor all members. If any issues arise, route the work back to member\_5 and address the issues. Otherwise, if no issues arise, move the work from member\_6 to member\_1 through member\_4 and monitor all members. Repeat these same steps for member\_7 and member\_8.

3. Stop the members that you no longer want to use by using the STOP DB2 command with MODE(QUIESCE).  
These members are now inactive.
4. Implement any configuration changes that you planned for in step 1.

---

## Potential configuration changes when you consolidate data sharing members

Although the process of consolidating members requires only a couple of steps, you should spend time planning for this change. You might want to change various settings and configurations. If you are also consolidating the number of z/OS LPARs, you need to do additional planning.

When you consolidate data sharing members, consider making changes in the following areas:

- **Subsystem parameters:** If you plan to run more concurrent active threads for each member after consolidation, you might need to adjust certain subsystem parameter settings to accommodate the larger number of threads. For example, you might need to increase certain in-memory cache sizes, certain limit settings, or both.

Consider adjusting the following subsystem parameters:

- DSMAX
- EDMPOOL
- MAXRBLK
- CTHREAD and MAXDBAT
- PARAMDEG
- MAXKEEPD
- MAXTEMPS
- SRTPOOL
- OUTBUFF
- LOBVALS
- CHKFREQ or CHKLOGR and CHKMINS
- XMLVALS

- **Active log data sets:** To accommodate the new consolidated member configuration, consider increasing the size and quantity of active log data sets. When more threads execute concurrently on each DB2 member, more DB2 logging is likely to occur on each member. Before consolidation, this logging was spread out over the active logs of multiple members.

Making sure that you have enough active log space helps performance. Reads from the active log are faster than those from the archive log, especially if archives are on tape.

You can dynamically add the active log data sets that you need while DB2 is operational. For instructions on how to add active log data sets, see Adding an active log data set to the active log inventory (DB2 Administration Guide).

- **Buffer pools:** If you plan to run more concurrent active threads for each member after consolidation, consider increasing the buffer pool sizes that are defined for each member. Also, consider adding new buffer pools.

For example, suppose that member\_1 had previously been running with a maximum of 400 concurrent active threads and a total buffer pool space allocation of 5 GB. Assume that after you consolidate your data sharing

members, you increase the maximum number of threads for member\_1 to 1200. In this case, the 5 GB of buffer pool space might no longer be sufficient, and threads might start to suffer I/O delays. In this case, you probably need to increase the 5 GB of buffer pool space to a larger size. The increase depends on whether the new threads predominantly access the same tables and indexes as the pre-existing threads. If the new threads predominantly access the same database objects, you probably do not need to increase the size of the buffer pools that much. If the new threads predominantly access different database objects, you probably need to increase the size of the buffer pools more.

After you consolidate to fewer members, the total amount of buffer pool space that is allocated for all members is likely to be less than the amount of space that was allocated before consolidation.

For more information about determining the appropriate buffer pool sizes, see Tuning database buffer pools (DB2 Performance).

- **Work file data sets:** If the number of concurrent active threads increases on a given subsystem after consolidation, consider allocating more temporary database storage. To create this extra space, create additional table spaces within the work file database.

To find temporary space usage statistics, look at data section 4 of statistics record IFCID 2. You can use these statistics to determine existing consumption and to help project future usage.

Also, consider the value of the MAXTEMPS\_RID subsystem parameter. This parameter determines the amount of temporary storage in the work file database that can be used for RID overflow. If this parameter is set to a value other than NONE, more work file space might be used for each thread and therefore, you might need to define more work file space.

You can use the MAXTEMPS subsystem parameter to regulate temporary space for each thread. You might need to increase the value of MAXTEMPS if MAXTEMPS\_RID is set to a value other than NONE.

**Recommendation:** If you need additional space, consider adding 32 KB work file data sets. Since Version 9 became available, DB2 exploits 32 KB work file data sets more aggressively than in earlier releases.

- **SQA and CSA:** If you plan to run more concurrent active threads for each member after consolidation, consider increasing the sizes of the SQA (system queue area) and CSA (common service area). After you consolidate members, carefully monitor the CSA subpools and SQA subpools to determine the amount of extra space, if any, that you need.

For example, a typical configuration might have a single DB2 subsystem on an LPAR that has 300 to 500 threads that are attached to DB2. Because of the advances in virtual storage constraint relief in Version 10, you can run 10 or more times the number of threads in a single LPAR. To handle this extra demand, depending on how many threads are running, you might need to increase the size of the SQA and CSA on the LPAR. This increase could be quite significant depending on several factors. For example, in the typical configuration described here, the size might need to increase to 50 MB. You need to consider your own configuration and monitor it carefully to determine an appropriate SQA and CSA size for your situation.

- **Real storage:** When DB2 members are consolidated, the demands on real storage can increase. For example, increases in buffer pools and the number of threads can increase the demands on real storage. Make sure that your consolidated system has enough real storage to prevent paging.

To estimate how much real storage you need, look at the IFCID 225 records for the existing members prior to consolidation. Subtract the buffer pool and EDM pool numbers from the real storage number in IFCID 225. (This calculation assumes no auxiliary storage.) The resulting value approximates the amount of real storage that the threads are using. You can then use this amount to calculate the new value for the LPAR that is to contain the consolidated members.

- **Number of LPARs and central processors (CPs):** After you consolidate data sharing members, consider consolidating the number of LPARs and releasing the CPs from the LPARs that are being deleted. Reconfigure the surviving LPARs to use the additional processors. When deciding whether and how to consolidate LPARs, consider availability and failover.
- **CICS attachment facility:** You can configure the CICS attachment facility to attach to a group or to attach to a specific member. If you have this facility configured to attach to a specific member, you might need to change many different CICS regions to point to the one of the members of the new consolidated data sharing group.

Group attachment has the advantage of easy movement of the CICS region around the sysplex. However, this method has the disadvantage of possibly allowing a DB2 member to be overloaded in Version 8 and Version 9 if a CICS region attaches to a member that does not have enough available storage. Attaching to specific member can guarantee improved availability.

- **DDF aliases:** When you consolidate data sharing members, you need to remove deleted members from any defined aliases.

A data sharing group can have multiple location names, or aliases. Each alias can identify a subset of the members for remote connection to the data sharing group.

Use the DSNJU004 utility to verify the DDF alias configuration. Then use the DSNJU003 utility to modify the DDF alias configuration to contain the remaining consolidated DB2 members that are needed for each alias.

- **TCP/IP addresses:** When members are removed from a data sharing group, review the TCP/IP configuration information for the remaining members to determine if you need to make any of the following changes:
  - Remove the IP address of any members that you are removing from the group.
  - Remove the resync port for any members that you are removing from the group.

You need to consider both ports that are reserved in the TCPPARMS data set and ports in the DVIPA configuration.

For instructions on how to customize the TCPPARMS data set, see Customizing the TCP/IP data sets or files (DB2 Installation and Migration). For instructions on how to customize the DVIPA configuration, see “Specify a DVIPA” on page 36.

- **Client CDB:** When you consolidate data sharing members, you might need to modify the configurations of remote clients that reference that data sharing group. If the client uses a DVIPA to access the data sharing group and its members, you do not need make changes. However, if the client uses IP addresses to access members of the data sharing group, you need to make changes. In this case, you need to remove from the client configuration the IP addresses of the members that are being deleted.
- **RBA rollover:** When you consolidate data sharing members, consider that you might need to reset the RBA on the consolidated members sooner than expected.



---

## Chapter 5. Removing members from the data sharing group

One of the features of DB2 data sharing is incremental growth, which entails being able to add members to an existing group. However, a situation might occur in which you want to remove members from the group, either temporarily or permanently.

For example, assume that your group does the job it needs to do for 11 months of the year. However, a surge of additional work every December requires you to expand your capacity. You can quiesce some members of the group for those 11 months. Those members are “dormant” until you restart them. A quiesced member can also remain dormant indefinitely.

A quiesced member (whether you intend for it to be quiesced indefinitely or only temporarily) still appears in displays and reports. It appears in DISPLAY GROUP output with a status of QUIESCED.

To permanently remove a member, you can delete it from the data sharing group by following the procedure in “Deleting data sharing members” on page 30.

---

### What data sets to keep when you quiesce a data sharing member

When you quiesce a member, you must keep the log data sets until they are no longer needed for recovery. Other members might need updates that are recorded on the quiesced member's log.

You must keep the BSDS, too, because it contains information that points to those log data sets. The BSDS is also needed for group restart. However, if you are confident that logs for the quiesced member are no longer necessary, because that member has been quiesced for a long time or is permanently quiesced, it is possible to delete the BSDS data sets. If you delete the BSDS data sets, expect the following message during group restart:

```
DSNR020I -DB2A csect-name START MEMBER DB1A, OR REPLY 'NO' or QUIESCED'
```

When you respond with QUIESCED, DB2 issues the following message:

```
DSNR030I -DB2A csect-name WILL CONTINUE WITHOUT THE DB1A MEMBER'S  
LOG,  
REPLY 'YES' OR 'NO'
```

In summary, you must do one of the following things to ensure that you have group restart capability:

- Keep the quiesced member's BSDS data sets (thus avoiding the preceding WTOR messages).
- Update your group restart procedure to ensure that operators know how to respond to the DSNR020I and DSNR030I messages.

---

### Quiescing a data sharing member

Quiescing a member of a data sharing group temporarily or indefinitely removes the member from the group. A quiesced member can be restarted at any time.

**GUPI** To quiesce a member of a data sharing group:

1. Stop the member you are going to quiesce. This example assumes that you want to quiesce member DB3A:  
`-DB3A STOP DB2 MODE(QUIESCE)`
2. From another member, enter the following commands:  
`DISPLAY GROUP`  
`DISPLAY UTILITY (*) MEMBER(member-name)`  
`DISPLAY DATABASE(*) RESTRICT`  
 If there is no unresolved work, you can stop now. However, if you want to create an archive log, continue to the next step.
3. If there is unresolved work, or if you want to create a disaster recovery archive log (as in step 4), start the quiesced member with ACCESS(MAINT).  
`-DB3A START DB2 ACCESS(MAINT)`  
 If there is unresolved work, resolve any remaining activity for the member, such as resolving indoubt threads, finishing or stopping utility work, and so on.
4. Optional: To create an archive log that can be sent to a disaster recovery site, archive the log for the member by entering the following command:  
`-DB3A ARCHIVE LOG`
5. Stop the member again with MODE(QUIESCE).  
`-DB3A STOP DB2 MODE(QUIESCE)`



#### Related reference

- [-DISPLAY DATABASE \(DB2\) \(DB2 Commands\)](#)
- [-DISPLAY GROUP \(DB2\) \(DB2 Commands\)](#)
- [-DISPLAY UTILITY \(DB2\) \(DB2 Commands\)](#)
- [-START DB2 \(DB2\) \(DB2 Commands\)](#)
- [-STOP DB2 \(DB2\) \(DB2 Commands\)](#)

---

## Deleting data sharing members

You can permanently remove a member from a data sharing group by deleting the member. Delete a member only if you are sure that you will not need the member or its BSDS and log data sets again because a deleted member cannot be restarted.

Members can be deleted from a data sharing group only from Version 10 new-function mode. In other words, all surviving members must be in Version 10 and the data sharing group must be in new-function mode. Members do not need to be active in Version 10 new-function mode before being deleted.

Deleting data sharing members is a two-step process that requires deactivating the member and then destroying the member.

### Deactivating data sharing members

The first step in deleting a member from a data sharing group is to deactivate the member. Deactivating a member ensures that no new log data is created for that member.

Before deactivating a member, ensure that the data sharing group is in Version 10 new-function mode.

The member that is to be deleted must be quiesced before the surviving members. This requirement ensures that the BSDSs of the surviving members record the quiesced state of the member that is to be deleted.

To deactivate a data sharing member:

1. Ensure that the member that is to be deactivated does not have outstanding units of work or active utilities.
  - a. Start the member with the ACCESS(MAINT) option.
  - b. Check messages DSNR004I through DSNR008I to identify any active units of recovery (URs).
  - c. Resolve any active URs.
  - d. Issue the DISPLAY UTILITY(\*) command to verify that the member has no active utilities.
  - e. Issue the TERM UTILITY command to terminate any active utilities. Alternatively, active utilities can be restarted and allowed to complete normally.
  - f. Issue the STOP DB2 MODE(QUIESCE) command to stop the subsystem.
2. Issue the DISPLAY GROUP command to verify that the member that is to be deactivated is in the quiesced state.
3. Stop all members of the group.
4. Make a backup copy of all the BSDSs in the group.
5. Use the DSNJU003 DELMBR DEACTIV,MEMBERID=x control statement against all BSDSs of all members of the group, where x is the member ID of the member that is to be deactivated. The DSNJU003 change log inventory utility deactivates the member.
6. Restart the surviving members. All surviving members should be restarted so that they can record the updated status of the deactivated member.

A deactivated member can be reactivated at any time with the DSNJU003 RSTMBR control statement.

#### Related reference

➡ -DISPLAY GROUP (DB2) (DB2 Commands)

➡ -DISPLAY UTILITY (DB2) (DB2 Commands)

➡ -START DB2 (DB2) (DB2 Commands)

➡ -STOP DB2 (DB2) (DB2 Commands)

➡ -TERM UTILITY (DB2) (DB2 Commands)

➡ Syntax and options of the DSNJU003 control statement (DB2 Utilities)

## Destroying data sharing members

Destroying a data sharing member permanently deletes it from the data sharing group. A destroyed member cannot be restored or reactivated.

Before you destroy a data sharing member, you must deactivate it. If you want to destroy a member that was deactivated and then later reactivated and restarted, you must deactivate it again.

To destroy a data sharing member:

1. Stop all members of the group.

2. Make a backup copy of all the BSDSs in the group.
3. Use the DSNJU003 DELMBR DESTROY,MEMBERID=x control statement against all BSDSs of all members of the group, where x is the member ID of the member that is to be destroyed.
4. Restart the surviving members. All surviving members should be restarted so that they can record the updated status of the destroyed member.

The member is deleted, and its BSDS and logs are no longer needed.

If you want to reuse the member name of a destroyed member, you must add a new member to the data sharing group. The new member can have the same name and member ID as the member that you destroyed.

**Related reference**

 [-START DB2 \(DB2\) \(DB2 Commands\)](#)

 [-STOP DB2 \(DB2\) \(DB2 Commands\)](#)

 [Syntax and options of the DSNJU003 control statement \(DB2 Utilities\)](#)

---

## Restoring deactivated data sharing members

A member that has been deactivated (not destroyed) can be reactivated and restarted at any time.

No other members of the data sharing group must be stopped before restoring a deactivated member.

To restore a deactivated member:

1. Use the DSNJU003 RSTMBR MEMBERID=x control statement against the BSDS of the member that is to be reactivated, where x is the member ID of the member.
2. Restart the member.
3. Restart all surviving members so that they can record the updated status of the reactivated member.

**Related reference**

 [-START DB2 \(DB2\) \(DB2 Commands\)](#)

 [Syntax and options of the DSNJU003 control statement \(DB2 Utilities\)](#)

---

## Chapter 6. Communicating with data sharing groups

A data sharing group can be a powerful server in your client/server environment. The group can be part of a TCP/IP network, part of an SNA network, or part of a network that uses both protocols.

The group has a single-system image to requesting applications, whether requests arrive through TCP/IP or SNA. Queries can originate from any system or application that issues Structured Query Language (SQL) statements as a requester in the formats that are described by Distributed Relational Database Architecture™ (DRDA®).

The distributed data facility (DDF) of DB2 uses TCP/IP and SNA to communicate with other DB2 subsystems. The DDF enables a DB2 subsystem to access data that is held by other database management systems. The DDF also enables the DB2 subsystem to make its own data accessible to other DB2 subsystems.

A data sharing group can support many more connections than a single member of the group can support. The DDF connections limit for a group is “ $n \times 150\,000$ ”, where  $n$  is the number of members in the group. Thus, a group with 16 members can support 2 400 000 DDF connections.

### Related tasks

 [Connecting distributed database systems \(DB2 Installation and Migration\)](#)

---

## Ways to access data sharing groups

A *requester* is a client that manages connections and data access for client applications. Requesters can access data sharing groups in TCP/IP and SNA networks. Any product that supports the application requester protocols that are defined by DRDA can be a requester.

In this topic, the focus is on the following requesters:

- IBM Data Server Driver for JDBC and SQLJ.
- IBM Data Server Driver for ODBC and CLI.
- DB2 for z/OS requester support with built-in workload balancing support.
- DB2 Connect™ Server with connection concentrator and workload balancing support.

In a connection request, a client application specifies the DB2 location name of the data sharing group to which it wants to connect. The requester then maps this DB2 location name to an actual network element that identifies the member to which to connect. In the case of TCP/IP, this network element is either a domain name or an IP address. In the case of SNA, this network element is an LU name.

**Note:** For TCP/IP, use the group dynamic virtual IP address and the common port to access the DB2 group for remote clients.

### Mixed TCP/IP and SNA networks

The members of a data sharing group support both TCP/IP and SNA network protocols. However, when acting as requesters, members must be configured to

communicate with other data sharing groups by using either TCP/IP or SNA. The DDF uses the group's communication database (CDB) to map each remote DB2 location name to either an IP address (for TCP/IP) or an LU name (for SNA). You can configure a DB2 location name for either TCP/IP or SNA, but not both. If you configure the same remote DB2 location name for both TCP/IP and SNA, only TCP/IP is used to communicate with the remote location.

---

## TCP/IP access methods

Together, TCP and IP use the client/server model of communication to enable communication between computers and computer networks of the same or different types.

TCP/IP is a set of communication protocols that support peer-to-peer connectivity functions for both local and wide area networks. On the sending end, Transmission Control Protocol (TCP) manages the assembly of a message or file into smaller packets that are transmitted over a network. On the receiving end, TCP manages the re-assembly of those packets into the original message or file. Internet Protocol (IP) handles the routing of each packet, ensuring that packets reach the correct destination.

It is strongly recommended that when setting up the IP addresses to be used by a specific member of a data sharing group, that the IP address be a dynamic Virtual IP Address (DVIPA). By using such a capability, the specific DB2 member of the data sharing group is "assigned" its own IP address no matter where in the Sysplex the DB2 subsystem is started. This process will then permit successful automatic handling of indoubt units-of-work during subsystem restart or recovery especially when that member of the data sharing group is being accessed by requesters operating under the control of XA transaction managers and two-phase commit transaction coordinators.

When considering the use of a data sharing group IP address, you can only use a DVIPA which is being managed by the Sysplex Distributor. If you attempt to use a DVIPA that is not managed by the Sysplex Distributor or any other address as the group IP address, only one member of the data sharing group can be reached via that address. This process defeats the purpose of a common IP address that can be used to access any member of the data sharing group.

A TCP/IP requester can use one of several access methods to connect to a data sharing group:

### Group access

A requester uses the group's dynamic virtual IP address (DVIPA) to make an initial connection to the DB2 location. Accessing the DB2 group using the group IP address is always successful if at least one member is started. If the Sysplex distributor is configured for the group IP address, the initial connection is based on the routing algorithm used by the Sysplex Distributor.

Alternatively, a requester can use a domain name, which is set up to return a list of member IP addresses. The requester will attempt to open a connection using every IP address in the list until the connection is successful.

If the requester is enabled for Sysplex workload balancing, the initial connection returns a list of members that are currently active and the

capacity of each member to perform work. The requester uses this information to balance subsequent connections.

If the requester is enabled for connection concentrator support, connections return an updated list members that are currently active and the capacity of each member to perform work. The requester uses this information to route subsequent transactions to the member with the highest capacity. Transactions are processed across the group to provide the best utilization of work across the group.

#### **Member-specific access**

A requester uses location alias names, which specifies one or more members of the group, to make an initial connection to one of the members that is represented by the alias names. The member that receives the request returns a list of members that are currently active and able to perform work. The requester uses this information to send subsequent connection and query requests to available members that are represented by the location alias names. A DB2 for z/OS requester can also use the SYSIBM.IPLIST table to isolate requests to a subset of members using IPLIST or location alias names support to access remote data sharing groups.

#### **Single-member access**

A requester uses a member's DVIPA, actual IP address, or domain name to connect and send queries to a single member of the group. All connection and query requests from that requester are directed to the same member.

## **Group access**

One method of connecting to a data sharing group is to use group access and dynamic virtual IP address (DVIPA) network addressing.

If you plan to use DVIPA network addressing, you must see your network administrator to determine if DVIPA can be configured for the Parallel Sysplex. DVIPA network addressing is required if you are using clients that support Sysplex workload balancing. You also need to use VIPA network addressing to support remote XA connections.

### **DVIPA network addressing**

Dynamic virtual IP addressing gives you the ability to assign a specific VIPA to a data sharing group and to each member of the group.

This address is independent of any specific TCP/IP stack within the Parallel Sysplex. Even if a member is moved to another z/OS system, as in the case of a failure or maintenance, the member remains accessible and retains the same VIPA.

The TCP/IP Sysplex Distributor can play a role in DVIPA network addressing. If the DB2's group DVIPA is configured to distribute connections across all members listening to the DB2 DRDA PORT, the Sysplex Distributor can factor real-time information, such as member status and Quality of Service (QoS) data. This data, along with information obtained from the Workload Manager (WLM), ensures that the best member is chosen to serve each client connection request.

If the group DVIPA is configured to distribute connections using the Sysplex Distributor, when the application specifies the DB2 location that maps to a group DVIPA, the Sysplex Distributor dispatches the requester's initial connection request to the active member with the most capacity. The requester ensures that workload balancing is invoked as indicated below:

- Requests are always routed to an active member, if at least one member is active.
- Requests are dynamically directed to those members with the most capacity.

Sysplex workload balancing functionality on DB2 for z/OS servers provides high availability for client applications that connect directly to a data sharing group. Sysplex workload balancing functionality provides workload balancing and automatic client reroute capability. This support is available for applications that use Java clients (JDBC, SQLJ, or pureQuery), or non-Java clients (ODBC, CLI, .NET, OLE DB, PHP, Ruby, or embedded SQL). For more information, see Java client support for high availability for connections to DB2 for z/OS servers (Application Programming for Java) and Non-Java client support for high availability for connections to DB2 for z/OS servers.

**Recommendation:** For the highest level of availability and workload balancing, use DVIPA network addressing. Enable the Sysplex Distributor for load balancing of initial connections to the best performing active member. After member availability and weight information is returned on the initial connection, you can use the weighted list to choose the appropriate member for subsequent connections.

#### Related concepts

 Java client support for high availability for connections to DB2 for z/OS servers (Application Programming for Java)

 Customization of IBM Data Server Driver for JDBC and SQLJ configuration properties (Application Programming for Java)

 Sysplex distributor (z/OS Communications Server: IP Configuration Guide)

#### Related information

 Non-Java client support for high availability for connections to DB2 for z/OS servers

#### Specify a DVIPA:

You can specify a DVIPA by using one of two methods: The BINDSPECIFIC method or the INADDR\_ANY method.

Use only the BINDSPECIFIC method or the INADDR\_ANY method to specify a DVIPA.

With the BINDSPECIFIC method, you specify a DVIPA on the PORT statement of the TCP/IP profile by using the BIND keyword as follows:

```
PORT
    446 TCP DB2ADIST SHAREPORT BIND 9.30.113.198
    5001 TCP DB2ADIST BIND 1::1
    446 TCP DB2BDIST SHAREPORT BIND 9.30.113.198
    5002 TCP DB2BDIST BIND 2::2
    446 TCP DB2CDIST SHAREPORT BIND 9.30.113.198
    5003 TCP DB2CDIST BIND 3::3
```

If you use the BINDSPECIFIC approach to specify DVIPAs for DB2, any IP addresses specified in the BSDS are ignored.

With the INADDR\_ANY method, you specify a DVIPA in the BSDS using by using the DSNJU003 utility (recommended). First, reserve the SQL, Secure SQL (if any), and then resync the ports as follows:

```

PORT
      446 TCP DB2ADIST SHAREPORT      /* SQL PORT */
      448 TCP DB2ADIST SHAREPORT      /* Secure SQL PORT */
      5001 TCP DB2ADIST                /* Resync PORT */
      446 TCP DB2BDIST SHAREPORT
      448 TCP DB2BDIST SHAREPORT
      5002 TCP DB2BDIST
      446 TCP DB2CDIST SHAREPORT
      448 TCP DB2CDIST SHAREPORT
      5003 TCP DB2CDIST

```

Now, if you use the INADDR\_ANY method or approach, you can specify the DVIPAs in the BSDS:

```

//DB2A EXEC PGM=DSNJU003
//SYSIN DD *
DDF GRPIPv4=9.30.113.198,IPV4=9.30.113.115,
    GRPIPv6=2001:DB8::8:800:200C:417A,
    IPV6=2001:DB8::8:800:200C:417B

```

**Recommendation:** Use INADDR\_ANY. BINDSPECIFIC is not supported with Secure SQL port. Also, with BINDSPECIFIC, DB2 supports only one member DVIPA (IPv4 or IPv6, but not both), and one group DVIPA (IPv4 or IPv6).

#### Example of TCP/IP configuration statements:

When you are setting up your data sharing group to use group access, looking at example TCP/IP configuration statements can be helpful.

The figures below show the TCP/IP configuration statements that are required to set up a three-member data sharing group to route DVIPA requests and perform workload balancing across the members. In these figures, the following statements apply:

- Vx represents the group DVIPA, and V1, V2, and V3 represent the members' DVIPAs.
- The group DVIPA and the member-specific DVIPAs are defined on each TCP/IP stack.
- The SHAREPORT option is required only when multiple members are started on the same LPAR or TCP/IP stack; without this option, TCP/IP does not allow multiple listeners on the DRDA port.
- The BIND (SPECIFIC) option restricts the clients to use only DVIPA to connect to DB2.
- VIPADYNAMIC statements:
  - The group DVIPA must be defined with the VIPADefine and VIPADISTRIBUTE statements on the TCP/IP stacks that are associated with the systems on which the Sysplex Distributor executes.
  - The group DVIPA must be defined with the VIPABACKUP statement on the TCP/IP stacks for DVIPA takeover. Note that the VIPABACKUP statements are coded with the MOVEABLE IMMEDIATE keywords, and that the VIPADISTRIBUTE statements are also specified on the backup TCP/IP stacks. This allows for the group DVIPA to be activated on one of the backup stacks if it is not active anywhere else in the Sysplex. For example, if z/OS-1 has not been started when z/OS-2 or z/OS-3 start up, then group DVIPA is activated on one of the backup stacks.
  - To allow for failover, the member-specific DVIPAs are defined with the VIPARANGE statement on all TCP/IP stacks.

**z/OS - 1**

```

PORT
  446 TCP DB2ADIST SHAREPORT BIND Vx
  446 TCP DB2BDIST SHAREPORT BIND Vx
  446 TCP DB2CDIST SHAREPORT BIND Vx
5001 TCP DB2ADIST BIND V1
5002 TCP DB2BDIST BIND V2
5003 TCP DB2CDIST BIND V3
VIPADYNAMIC
  VIPARANGE DEFINE INTFV1 V1/128
  VIPARANGE DEFINE INTFV2 V2/128
  VIPARANGE DEFINE INTFV3 V3/128
  VIPADEFINE MOVE IMMED INTFVx Vx
  VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

```

**z/OS - 2**

```

PORT
  446 TCP DB2ADIST SHAREPORT BIND Vx
  446 TCP DB2BDIST SHAREPORT BIND Vx
  446 TCP DB2CDIST SHAREPORT BIND Vx
5001 TCP DB2ADIST BIND V1
5002 TCP DB2BDIST BIND V2
5003 TCP DB2CDIST BIND V3
VIPADYNAMIC
  VIPARANGE DEFINE INTFV1 V1/128
  VIPARANGE DEFINE INTFV2 V2/128
  VIPARANGE DEFINE INTFV3 V3/128
  VIPABACKUP 1 MOVE IMMED INTFVx Vx
  VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

```

**z/OS - 3**

```

PORT
  446 TCP DB2ADIST SHAREPORT BIND Vx
  446 TCP DB2BDIST SHAREPORT BIND Vx
  446 TCP DB2CDIST SHAREPORT BIND Vx
5001 TCP DB2ADIST BIND V1
5002 TCP DB2BDIST BIND V2
5003 TCP DB2CDIST BIND V3
VIPADYNAMIC
  VIPARANGE DEFINE INTFV1 V1/128
  VIPARANGE DEFINE INTFV2 V2/128
  VIPARANGE DEFINE INTFV3 V3/128
  VIPABACKUP 2 MOVE IMMED INTFVx Vx
  VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

```

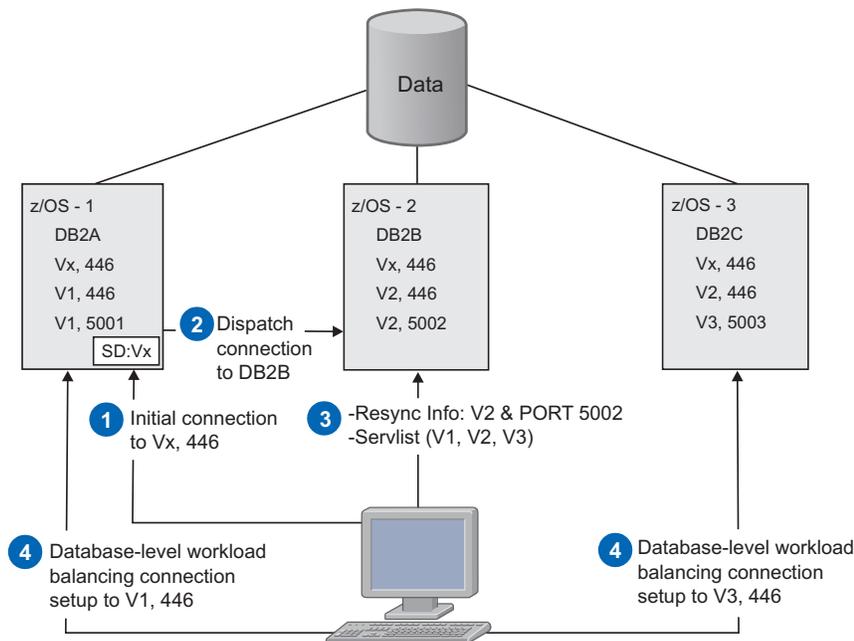


Figure 16. Example TCP/IP network configuration (BINDSPECIFIC IPv6 dynamic virtual IP addressing)

In this network configuration:

1. The initial connection uses the group dynamic Virtual IP Addresses (Vx). Port 446 is identified as the DRDA port in each member's PORT statements.
2. The Sysplex Distributor dispatches the initial connection request to the member with the lightest workload (DB2B).
3. Resynchronization information and a list of members in the data sharing group are returned to the requester.
4. Database-level workload balancing connections are established.

**z/OS - 1**

```

PORT
  446 TCP DB2ADIST SHAREPORT
  446 TCP DB2BDIST SHAREPORT
  446 TCP DB2CDIST SHAREPORT
5001 TCP DB2ADIST
5002 TCP DB2BDIST
5003 TCP DB2CDIST
VIPADYNAMIC
VIPARANGE DEFINE INTFV1 V1/128
VIPARANGE DEFINE INTFV2 V2/128
VIPARANGE DEFINE INTFV3 V3/128
VIPADefINE MOVE IMMED INTFVx Vx
VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

```

**z/OS - 2**

```

PORT
  446 TCP DB2ADIST SHAREPORT
  446 TCP DB2BDIST SHAREPORT
  446 TCP DB2CDIST SHAREPORT
5001 TCP DB2ADIST
5002 TCP DB2BDIST
5003 TCP DB2CDIST
VIPADYNAMIC
VIPARANGE DEFINE INTFV1 V1/128
VIPARANGE DEFINE INTFV2 V2/128
VIPARANGE DEFINE INTFV3 V3/128
VIPABACKUP 1 MOVE IMMED INTFVx Vx
VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

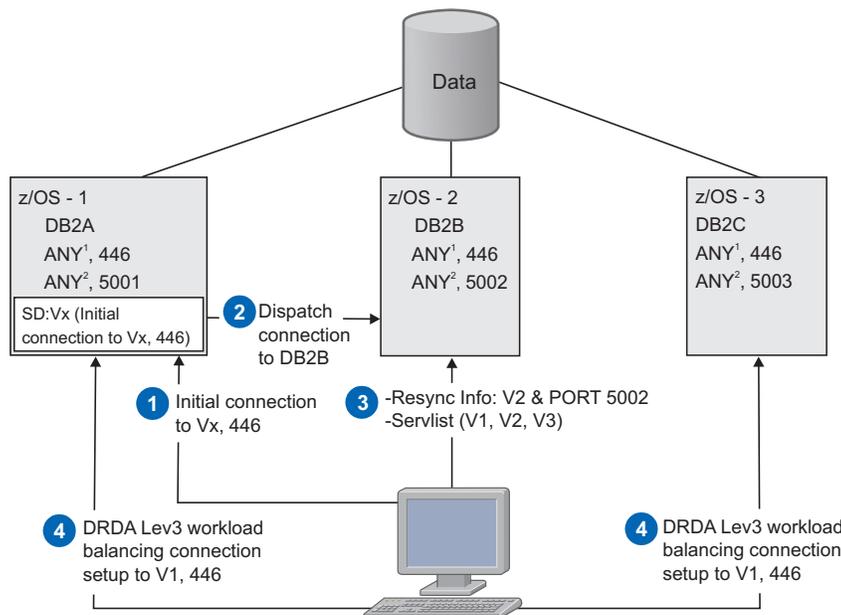
```

**z/OS - 3**

```

PORT
  446 TCP DB2ADIST SHAREPORT
  446 TCP DB2BDIST SHAREPORT
  446 TCP DB2CDIST SHAREPORT
5001 TCP DB2ADIST
5002 TCP DB2BDIST
5003 TCP DB2CDIST
VIPADYNAMIC
VIPARANGE DEFINE INTFV1 V1/128
VIPARANGE DEFINE INTFV2 V2/128
VIPARANGE DEFINE INTFV3 V3/128
VIPABACKUP 2 MOVE IMMED INTFVx Vx
VIPADISTRIBUTE DEFINE INTFVx
  PORT 446 DESTIP ALL
ENDVIPADYNAMIC

```



<sup>1</sup> Including Vx and V1 or V2 or V3

<sup>2</sup> Including V1 or V2 or V3

Figure 17. Example TCP/IP network configuration (INADDR\_ANY IPv6 dynamic virtual IP addressing)

In this network configuration:

1. The initial connection uses the group dynamic Virtual IP Addresses (Vx). Port 446 is identified as the DRDA port in each member's PORT statements.
2. The Sysplex Distributor dispatches the initial connection request to the member with the lightest workload (DB2B).
3. Resynchronization information and a list of members in the data sharing group are returned to the requester.
4. Database-level workload balancing connections are established.

#### Related concepts

 Using dynamic VIPAs (z/OS Communications Server: IP Configuration Guide)

#### Configuration requirements:

DVIPA network addressing requires you to define multiple DVIPAs, one for the data sharing group and one for each member of the group.

Coding example for the DDF BSDS statement processed by DSNJU003:

```
//DB2A EXEC PGM=DSNJU003
//SYSIN DD *
DDF GRPIPv4=9.30.113.198,IPV4=9.30.113.115,
  GRPIPv6=2001:DB8::8:800:200C:417A,
  IPV6=2001:DB8::8:800:200C:417B
```

- Group DVIPA

The group DVIPA is common to all members of the group, and is used to make the initial connection to a member of the group. It should be specified in one of the two places. Do not specify in both places.

Define the group DVIPA at the end of each DRDA port number entry in the PORT statement of each member's TCP/IP profile data set (PROFILE.TCPIP). For example:

```
PORT
      446 TCP DB2ADIST SHAREPORT BIND Vx
      446 TCP DB2BDIST SHAREPORT BIND Vx
      446 TCP DB2CDIST SHAREPORT BIND Vx
```

Specify the group DVIPA in the BSDS using the DSNJU003 utility as shown in **\*\*See coding example\*\***:

The group DVIPA should be owned and managed by the Sysplex Distributor, which provides workload balancing among members during new connection request processing.

- Member-specific DVIPAs

Every member of the group needs a unique DVIPA, which is used to directly connect to a particular member after the initial connection.

Configure DVIPAs for each member of the data sharing group that can be started. You can specify the member's DVIPA in one of the two places below. Do not specify in both places.

- Specify a RESYNC PORT statement in the member's TCP/IP profile data set (PROFILE.TCPIP). For example:

```
PORT
      5001 TCP DB2ADIST BIND V1
      5002 TCP DB2BDIST BIND V2
      5003 TCP DB2CDIST BIND V3
```

- Specify it in the BSDS using the DSNJU003 utility as shown in **\*\*coding example\*\***.

Also ensure that the following conditions exist before making a data sharing group available for group access that uses DVIPA network addressing:

- All members of the group are configured with DVIPAs before starting the DDF on any member of the group.
- All members of the group are configured with DVIPAs before remote connections are made to any member of the group.
- All members are running OS/390® Version 2, Release 10 or later.

Before moving to DVIPA network addressing from another access method, have all members check for the existence of indoubt threads. If any indoubt threads exist, resolve them before moving. To check for indoubt threads, use the DISPLAY THREAD command.

## Related concepts

 Sysplex distributor (z/OS Communications Server: IP Configuration Guide)

## Related reference

 -DISPLAY THREAD (DB2) (DB2 Commands)

## Prepare for failure recovery with DVIPA:

Using DVIPA is the most flexible way to be prepared for DB2 or system failure. If at least one member of a data sharing group is currently active and can perform work, connection attempts of requesters do not fail.

If a member suffers a failure or the underlying z/OS system suffers an outage, automation software such as z/OS Automatic Restart Manager (ARM) can restart the member on a different z/OS system. When this happens, the member-specific DVIPA is also moved to the new system and automatic recovery of any indoubt threads is performed, thereby enabling requesters to access the member on the new z/OS system.

## DNS network addressing

A domain name server (DNS) provides standard name resolution services for IP applications in a Parallel Sysplex cluster, converting domain names into IP addresses.

If, in its connection request, an application specifies a DB2 location name that maps to a group domain name, the requester queries the DNS with a call to *gethostbyname*. Either the DNS resolves the name to a member's IP address, or it resolves the name to a list of member IP addresses. The DNS returns the IP address or list of IP addresses to the requester, which then uses it to make an initial connection to a member of the data sharing group.

If your DNS supports it, and the Sysplex Distributor is not used to manage the group's DVIPA, configure the DNS to return a list of member IP addresses (member DVIPAs) instead of returning a single IP address. The list includes member IP addresses and ports, and a weight for each active member that indicates the member's current capacity. The requester uses the returned information to connect to the member with the most capacity. If you use the Sysplex Distributor to manage the DVIPA of the group, configure the DNS to return the group DVIPA.

**Recommendation:** If your DNS supports it, configure the DNS to return a list of member IP addresses instead of returning a single IP address. Because a DNS does not know whether a member is active when it returns the member's IP address to the requester, the DNS might return the IP address of an inactive member. Using this IP address on a subsequent connection request causes the request to fail. In contrast, if you configure the DNS to return a list of member IP addresses, the connection request can be retried using another IP address on the list. When the DNS retries the connection request, it will run through the list once, searching for the active member.

## Example of group access configuration

Applications that use group access use the group's DB2 location name (DB2A) to direct connection requests to the group.

The configuration in the figure below is an example of group access that uses DNS network addressing

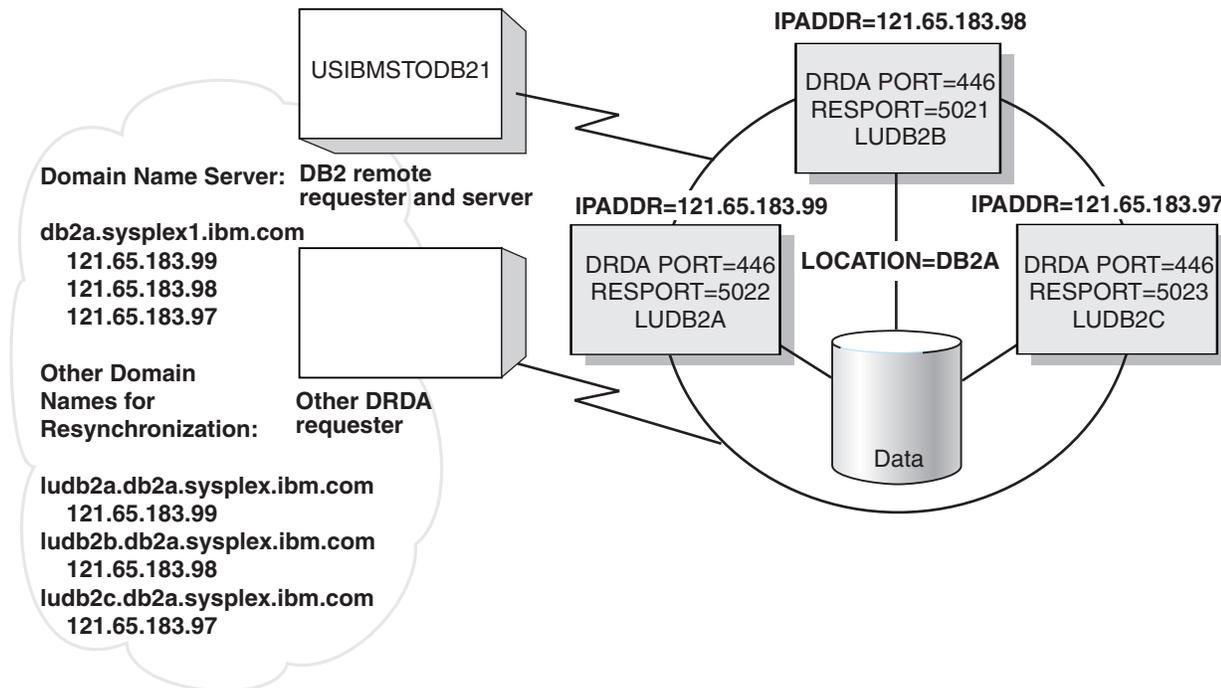


Figure 18. Example TCP/IP network configuration (DNS network addressing)

**Note:** When a DB2 for z/OS system is a member of a data sharing group, DB2 cannot be utilized as a DRDA XA server. As a result, the XA Transaction Manager cannot directly access the DB2 for z/OS DRDA server. Instead, you must configure the DRDA requester system in such a way that it insulates the DB2 for z/OS DRDA server from receiving XA calls from the XA Transaction Manager.

## Member-specific access

A DB2 location name represents all members of a data sharing group. In contrast, member-specific access uses location aliases that map to the domain names or IP addresses of only a subset of the members of the data sharing group.

Requesters can use location aliases to bypass default TCP/IP workload balancing and to establish connections with one or more members. Workload is balanced among members at the requester's discretion.

With member-specific access, a requester uses a location alias to make an initial connection to one of the members that is represented by the alias. The member that receives the connection request works in conjunction with Workload Manager (WLM) to return a list of members that are currently active and able to perform work. The list includes member IP addresses and ports, and a weight for each active member that indicates the member's current capacity. The IP addresses are the member-specific alias IP addresses, if they exist, or the member-specific IP addresses. The requester uses this information to connect to the member or members with the most capacity that are also associated with the location alias.

To enable RACF PassTickets you must define either a generic LU name for the data sharing group, an IPNAME for the data sharing group, or both. If all members of the data sharing group are to be configured to enable TCP/IP communications

only, then all members of the data sharing group must share the same IPNAME value. If you use RACF PassTickets, then all LOCATION aliases must eventually refer to a single common IPNAMES row whose LINKNAME value must match the remote group's generic LU name or IPNAME value. The IPADDR value in the IPNAMES row must eventually refer to the group distributing DVIPA of the data sharing group.

If some of the members of the data sharing group are to be configured to enable TCP/IP communications only and the others are to enable both, then the IPNAME value of the TCP/IP only members must match the generic LU name of the members that enable both TCP/IP and SNA/APPC communications. The generic LU name or IPNAME value is used as the RACF application name when validating a RACF PassTicket at any member of the data sharing group.

### Related tasks

[➦ Sending RACF PassTickets \(DB2 Administration Guide\)](#)

## Member-specific location aliases

Member-specific location aliases represent one, a subset, or all members of a data sharing group.

Location aliases are useful in several ways.

- Member-specific access requires that you identify to the server (remote data sharing group) each location alias that is defined by a requester. Doing so enables a server to recognize itself as the intended recipient of connection requests that specify location aliases instead of the server's location name.
- After adding a new member to a data sharing group, you can create an alias for the member's old (subsystem) DB2 location name that maps to its new (group) DB2 location name. Doing so enables applications that are coded to connect to the member's old DB2 location name to continue to work.
- Location aliases enable you to define subsets of data sharing group members. Subsetting gives you the ability to limit the members to which DRDA requesters can connect.

| Location aliases can be defined without an *alias-port* or *alias-secpport* value. Clients  
| that use a location alias without an *alias-port* or *alias-secpport* value receive  
| information only about the member that processes the connection request in the  
| list of servers. To distribute work requests across the members or a subset of a  
| group, clients must be configured to use the DB2 group location name or location  
| name that is defined for a subset of members.

## Dynamic location aliases

You can use the MODIFY DDF command with the ALIAS option to define and manage as many as 40 location aliases dynamically. You can start, stop, cancel, change, and delete dynamic location aliases without stopping either DDF or DB2. You can use the DISPLAY DDF command to find information about existing location aliases.

## Static location aliases

| You can use DSNJU003 (change log inventory) utility to define and modify as  
| many as 8 static location aliases. Changes to these aliases require you to stop both  
| DDF and DB2. The MODIFY DDF command cannot be used to manage this type  
| of alias. Any alias of this type that has the same name as an alias that was

previously defined by the MODIFY DDF command is not used. In that case, the dynamically defined alias takes precedence, and is used instead.

### Related concepts

“Update the BSDS with the DSNJU003 utility” on page 74

### Related tasks

“Defining dynamic location aliases”

“Managing dynamic location aliases” on page 45

### Related reference

 -DISPLAY LOCATION (DB2) (DB2 Commands)

 -MODIFY DDF (DB2) (DB2 Commands)

## Defining dynamic location aliases

You can define dynamic location aliases that enable you to manage subsets of members in a data sharing group, without stopping and restarting DDF or DB2.

Before you can define dynamic location aliases, DB2 must be started.

You can define as many as 40 dynamic location aliases. These aliases cannot be defined or managed by the DSNJU003 utility, and the DSNJU004 utility does not print any information about these aliases.

When a static and dynamic location alias are defined with the same name, DB2 uses only the dynamic location alias. The statically defined alias with the same name is ignored.

To define dynamic location aliases:

1. Issue the MODIFY DDF command and specify the ALIAS and ADD options. For example, you might issue the following command, where *alias-name* is the name of the alias:

```
-MODIFY DDF ALIAS(alias-name) ADD
```

The location alias is defined as a stopped location alias.

2. Issue the MODIFY DDF command again to configure the alias. You can specify only one additional option each time that you issue the MODIFY DDF command with the alias option. For example, to configure the ALIAS1 location alias with a port of 9000 and an IPv4 address of 1.1.1.1, you would issue the following sequence of commands:

```
a. -MODIFY DDF ALIAS(ALIAS1) PORT(9000)
```

```
b. -MODIFY DDF ALIAS(ALIAS1) IPV4(1.1.1.1)
```

3. Issue the MODIFY DDF command and specify the ALIAS and START options. For example, you might issue the following command, where *alias-name* is the name of the alias:

```
-MODIFY DDF ALIAS(alias-name) START
```

DDF accepts requests to the newly defined location alias whenever DDF is started.

You can manage location aliases that are created by the MODIFY DDF command dynamically by issuing the MODIFY DDF command to stop or cancel the alias, modify its configuration, and restart it, all without stopping DDF or DB2.

## Related concepts

“Member-specific location aliases” on page 43

## Related tasks

“Managing dynamic location aliases”

## Managing dynamic location aliases

You can stop, cancel, modify, and start dynamic location aliases without stopping and restarting DDF or DB2.

Before you can modify existing dynamic location aliases, DB2 must be started.

Unlike statically defined location aliases, you cannot define or modify dynamic aliases by using the DSNJU003 utility, and the DSNJU004 utility does not print any information for dynamic location aliases.

When a static and dynamic location aliases are defined with the same name, DB2 accepts requests only to the dynamic location alias.

To modify an existing dynamic location alias:

1. Issue the MODIFY DDF command to stop or cancel the alias.

Option	Description
<b>To stop the alias</b>	Issue the following command: <code>-MODIFY DDF ALIAS(<i>alias-name</i>) STOP</code>  DB2 stops accepting new connection requests to the specified alias and existing database access threads continue to process connections to the specified alias. Inactive connections to the alias are automatically closed.
<b>To cancel the alias</b>	Issue the following command: <code>-MODIFY DDF ALIAS(<i>alias-name</i>) CANCEL</code>  DB2 stops accepting new connection requests to the specified alias and existing database access threads that process connections to the specified alias are terminated. Inactive connections to the alias are automatically closed.

Stopped and canceled dynamic location aliases do not start automatically when you start DDF. In a data sharing group, DB2 deregisters the alias with WLM, which means that DB2 is no longer included in sysplex workload balancing information, related to the alias, that is returned to remote client systems. Those steps are not needed in non-data sharing environments, because DB2 never registers the alias with WLM.

2. Issue the MODIFY DDF command one or more times to modify the configuration of the alias. You can specify only one additional option with the MODIFY DDF command when you specify the ALIAS option. Therefore, you might need to issue the MODIFY DDF command more than once to make multiple changes to the configuration of the alias.

3. Issue the MODIFY DDF command to start the alias. For example, you might issue the following command, where *alias-name* is the name of the alias:

```
-MODIFY DDF ALIAS(alias-name) START
```

For example, assume that a location alias ALIAS1 was defined by the MODIFY DDF command, and has the following configuration:

- A port: 9000
- An IPv4 address: 1.1.1.1

However, you want to modify the location alias to use following configuration:

- The default port
- No IPv4 address
- An IPv6 address: 3::3

You might issue the following series of commands:

1. -MODIFY DDF ALIAS(ALIAS1) CANCEL  
DB2 stops accepting new connection requests to the specified alias and existing database access threads that process connections to the specified alias are cancelled. You can now modify the configuration of the alias.
2. -MODIFY DDF ALIAS(ALIAS1) NPORT  
The port value is removed from the alias. The group SQL port is used instead.
3. -MODIFY DDF ALIAS(ALIAS1) NIPV4  
The IPv4 address is removed from the alias. The member-specific IP address is returned in the server list instead.
4. -MODIFY DDF ALIAS(ALIAS1) IPV6(3::3)  
The IPv6 address is added to the alias.
5. -MODIFY DDF ALIAS(ALIAS1) START  
DDF starts accepting new connections to the alias using its new configuration.

#### **Related concepts**

“Member-specific location aliases” on page 43

#### **Related tasks**

“Defining dynamic location aliases” on page 44

## **Single-member access**

Single-member access using TCP/IP is the same method that is used to access DB2 in non-data sharing environments.

With single-member access, a requester uses the real or virtual IP address in its TCP/IP configuration to connect to a specific member of the group. (Alternatively, a requester's TCP/IP configuration can contain a domain name, which resolves to an IP address.) The requester sends all connection and query requests to this same member.

**DB2 Connect:** DB2 Connect Server is a connectivity server that concentrates and manages connections from multiple desktop clients and web applications to DB2 database servers. Configure DB2 Connect to use single-member access by disabling its Sysplex or concentrator support. When you disable Sysplex or concentrator support, no failover capability will be provided and if the single member fails, the application will not have any access. If you do not disable Sysplex support, DB2 Connect uses DRDA workload balancing by default to allocate requests among the members of the group. For more information on enabling or disabling DB2 Connect, Sysplex support, and concentrator functions, see *IBM DB2 Connect User's Guide*, which you can download at <http://www-01.ibm.com/support/docview.wss?rs=71&uid=swg27015148>.

**Recommendation:** Do not use single-member access for the following reasons:

- It is dependent on the specified member being operational.
- It provides no workload balancing or failover capability across the members of a data sharing group

#### Related tasks

 [Connecting systems with TCP/IP \(DB2 Installation and Migration\)](#)

## Setting up DB2 for z/OS as a requester

Much of the processing in a distributed database environment requires the exchange of messages with other locations in the network.

When DB2 for z/OS acts as a requester, it can connect applications that run on the z/OS system to remote database servers, including DB2 for z/OS, DB2 for Linux, UNIX, and Windows, DB2 for i, and DB2 Server for VSE & VM.

In its role as a requester, DB2 for z/OS accepts DB2 location names and translates them into TCP/IP addresses. It uses the communications database (CDB) to register location names and their corresponding network parameters. The data that is stored in the CDB enables DB2 to pass the required information when making distributed database requests over TCP/IP connections.

For this processing to be performed correctly, you must:

1. Define the DB2 for z/OS requester to the local TCP/IP system.
2. Identify the remote data sharing groups to which applications can connect.

#### Related tasks

 [Defining the DB2 subsystem to TCP/IP \(DB2 Installation and Migration\)](#)

### Remote data sharing group requirements

When an application requests data from a remote data sharing group, the DB2 for z/OS requester searches the CDB for information about the remote group. DB2 uses the CDB to store information about how to communicate with remote groups, and the DDF uses the CDB to map DB2 location names and location aliases to IP addresses.

Group access requires:

- A DB2 location name entry in the SYSIBM.LOCATIONS table
- An entry in SYSIBM.IPNAMES table with the group DVIPA and any outbound user authentication requirements

Member-routing access requires:

- Location alias entries in the SYSIBM.LOCATIONS table
- An entry in SYSIBM.IPNAMES table with the group DVIPA and any outbound user authentication requirements
- Optionally, location alias member entries in the SYSIBM.IPLIST table if you use subset location aliases

Single-member access requires:

- A location alias entry in the SYSIBM.LOCATIONS table.
- An entry in SYSIBM.IPNAMES table with the group DVIPA and any outbound user authentication requirements
- A location alias member entry in the SYSIBM.IPLIST table

## **SYSIBM.LOCATIONS:**

SYSIBM.LOCATIONS maps the DB2 location names in connection requests to the port numbers (or service names) of remote systems.

SYSIBM.LOCATIONS must contain at least one row for each remote group, depending on the access method that is used.

- For group access, the LOCATION column of the row contains the group's DB2 location name and PORT name. The LOCATION, PORT and the corresponding IPNAMES row identifies the members used to access DB2.
- For member-routing access, the LOCATION column of each row contains the group PORT number and location alias that identifies one, several, or all members of the group. The LOCATION, PORT and corresponding IPNAMES row controls which member is used to access DB2.
- For single-member access, the LOCATION column of the row contains the PORT number and location alias that identifies one member of the group. The LOCATION, PORT, and corresponding IPNAMES row identifies controls which member is used to access DB2.

**Tip:** You can specify a case-sensitive service name, instead of a port number, in the PORT column of the SYSIBM.LOCATIONS table.

## **SYSIBM.IPNAMES:**

SYSIBM.IPNAMES maps DB2 location names to the IP addresses or domain names of remote systems.

It also maps DB2 location names to the network security information required by the remote system. SYSIBM.IPNAMES must contain at least one row for each remote group, depending on the access method that is used.

- For group access, the IPADDR column of the row contains the group's domain name.
- For member-specific access, the LINKNAME column of each row contains a link name, and the IPADDR column of each row is blank.
- For single-member access, the LINKNAME column of the row contains a link name, and the IPADDR column of the row is blank.

## **SYSIBM.IPLIST:**

SYSIBM.IPLIST supports member-specific access to a remote data sharing group by enabling you to associate location aliases with one or more members of the group.

SYSIBM.IPLIST must contain a row for every member that is associated with a location alias. For members that are associated with multiple location aliases, insert multiple rows.

- For member-specific access, the LINKNAME column of each row contains a location alias and the IPADDR column of that row contains the DVIPA or domain name of a member that is associated with that location alias.
- For single-member access, the LINKNAME column of the row contains a location alias and the IPADDR column of that row contains the DVIPA or domain name of the member that is associated with that location alias.

**Recommendation:** If you are trying to limit the members of a data sharing group that can be accessed from this DB2 for z/OS requester, then the requester should

be setup to access a data sharing group subset setup at the server (data sharing group). If the requester is going to access a group's subset, then the SYSIBM.IPLIST table should not be used.

**Example:** Assume that a remote data sharing group has six members. With member-specific access, you might have two location aliases, one of which is associated with three of the members and the other of which is associated with two of the members. In this example, you insert five rows into the SYSIBM.IPLIST table, three rows for the members that are associated with the first location alias and two rows for the members that are associated with the second location alias.

#### **How DB2 sends requests:**

When sending a request, DB2 uses the value in the LINKNAME column of the SYSIBM.LOCATIONS table to determine which network protocol to use.

- If DB2 finds the same value in the LINKNAME column of the SYSIBM.IPNAMES table, it uses TCP/IP.
- If DB2 finds the same value in the LUNAME column of the SYSIBM.LUNAMES table, it uses SNA.
- If DB2 finds the same value in both SYSIBM.IPNAMES and SYSIBM.LUNAMES, it uses TCP/IP.

#### **Updates to the communications database for TCP/IP connections:**

You can update the CDB while DDF is active.

Changes to the SYSIBM.LOCATIONS, SYSIBM.IPNAMES, SYSIBM.IPLIST tables take effect in the following manner for TCP/IP connections:

- Updates take effect when DDF opens a new connection to the remote group.
- Updates do not affect communication already in progress; existing communication continues as if the updates had not occurred.

## **Configuring data sharing groups as TCP/IP servers**

Configuring a DB2 for z/OS data sharing group as a TCP/IP server is very simple.

The process consists of these steps:

1. Specifying the DRDA port number on which all members listen for incoming SQL requests
2. Specifying a unique resynchronization port number for each member
3. Designating subsets of members, if you want to limit the members to which DRDA requesters can connect
4. Specifying a generic LU name and IPNAME value for the data sharing group, if you use RACF PassTickets

### **Specifying the DRDA port number**

DB2 requires that all members of a data sharing group use the same, well-known port number to receive incoming SQL requests.

446 is the recommended DRDA port number. DRDA port numbers are stored in the bootstrap data sets (BSDSs) of members.

Use installation panel DSNTIP5 or the DSNJU003 (change log inventory) utility to specify the DRDA port number (PORT) for each member. Specify the same DRDA port number for all members of the same data sharing group.

The network administrator must also register the DRDA port number with TCP/IP on each member's z/OS system.

#### Related reference

 [DSNJU003 \(change log inventory\) \(DB2 Utilities\)](#)

#### Reserving the DRDA port:

If a member of a data sharing group is restarted on another LPAR, the TCP/IP on that system must be configured to allow the member to use the DRDA port.

To ensure this, reserve the DRDA port on each system for the DB2 DDF address space by assigning the port to every member that could conceivably start on that system. By explicitly assigning this port, you prevent other programs from using the DRDA port number.

On each system, replicate the TCP/IP PORT configuration profile statement shown here:

```
PORT
:
  446 TCP DB1ADIST SHAREPORT
  446 TCP DB2ADIST SHAREPORT
  446 TCP DB3ADIST SHAREPORT
  446 TCP DB4ADIST SHAREPORT
```

**Recommendation:** Specify the SHAREPORT option, as shown in the TCP/IP PORT configuration profile statement. Specifying this option configures TCP/IP to allow multiple listeners on the DRDA port (port 446). As client connection requests arrive for this port, TCP/IP distributes them across the members. TCP/IP selects the member with the fewest number of connections (both active and queued) at the time that the request is received.

The member chosen by TCP/IP receives all of the DRDA server's workload for that TCP/IP instance, leaving the other members with no TCP/IP server threads for DRDA. This is transparent to the DRDA clients if the member that is processing the TCP/IP requests does not reach the MAX REMOTE CONNECTED thread limit. If this limit is reached, the client's connection request is rejected.

**Tip:** After you resolve a failure situation, move the member back to its original CPC.

#### Specifying the resynchronization port numbers

DB2 requires that each member of a data sharing group have a resynchronization port number that is unique within the Parallel Sysplex.

In the event of a failure, this unique port number allows a requester to reconnect to the correct member so that units of work that require two-phase commit can be resolved.

Use the DSNJU003 (change log inventory) utility to specify a unique resynchronization port number (RESYNC) for each member of the group.

The network administrator must also register the resynchronization port number with TCP/IP on each member's z/OS system.

#### Related reference

 [DSNJU003 \(change log inventory\) \(DB2 Utilities\)](#)

### Configuring subsets for member-specific access

DB2 for z/OS allows you to define subsets of data sharing group members in TCP/IP networks for member-specific access.

By designating subsets of members, you can:

- Limit the members to which DRDA requesters can connect.  
System and database administrators might find this useful for any number of purposes.
- Ensure that initial connections are established only with members that belong to the specified subset.  
Without subsets, requesters can make initial and subsequent connections to any member of the data sharing group.
- Provide requesters with information about only those members in the subset.  
With subsets, a member that receives an initial connection request can return to the requester a list of members that are currently active, able to perform work, and represented by the location alias.

To configure a subset for member-specific access:

- Complete one of the following actions:
  - Use the MODIFY DDF command to modify member specific location aliases without stopping DB2. For example, to modify the port number used by a member for an alias ALIAS1, you would issue the following sequence of commands.
    1. Issue a MODIFY DDF command to stop the alias:  

```
MODIFY DDF ALIAS (ALIAS1) STOP
```
    2. Issue a MODIFY DDF command to modify the member-specific alias port:  

```
MODIFY DDF ALIAS (ALIAS1) PORT (447)
```

You can also add new aliases and define, modify, and remove SECPort, IPV4 and IPV6 values for an existing alias.
    3. Issue a MODIFY DDF command to start the alias:  

```
MODIFY DDF ALIAS (ALIAS1) START
```

      - If any ports are specified for the alias, DDF registers any subset location aliases with z/OS Workload Manager (WLM). The list of members in the subset is managed automatically by z/OS.
      - DDF adds the TCP/IP port numbers of the subset location aliases to a TCP/IP SELECT socket call for the SQL request listener. Doing so enables the Sysplex Distributor to send requests that are intended for subset members to only those members that belong to the subset. It also enables members of the subset to respond to those requests.
  - Stop DB2 and invoke the DSNJU003 utility and specify the *alias-port* parameter of the ALIAS option of the DSNJU003 (change log inventory) utility. When you start DDF, it performs the following tasks:
    - If any ports are specified for an alias, DDF registers any subset location aliases with z/OS Workload Manager (WLM). The list of members in the subset is managed automatically by z/OS.

- DDF listens for SQL requests on the alias ports specified for each subset that it is part of. This enables members of a subset identified by an alias name to respond to SQL requests to that particular alias, either coming in directly from a remote client or through the Sysplex Distributor.
- If you use DVIPAs for failover, use the VIPARANGE statement on each TCP/IP stack of the subset.

### **Specifying a generic LU name and IPNAME value for the data sharing group**

If you use RACF PassTickets for security, you must define either a generic LU name for the data sharing group, an IPNAME for the data sharing group, or both.

If all members of the data sharing group are to be configured to enable TCP/IP communications only, then all members of the data sharing group must share the same IPNAME value. Use the DSNJU003 (change log inventory) utility to define the IPNAME value. If all of the members of the data sharing group are to be configured to enable both SNA/APPC and TCP/IP communications, then all members of the data sharing group must share the same generic LU name. Use the DB2 GENERIC LUNAME parameter on installation panel DSNTIPR or use the DSNJU003 (change log inventory) utility to define the generic LU name.

If some of the members of the data sharing group are to be configured to enable TCP/IP communications only and the others are to enable both, then the IPNAME value of the TCP/IP only members must match the generic LU name of the members that enable both TCP/IP and SNA/APPC communications. The generic LU name or IPNAME value is used as the RACF application name when validating a RACF PassTicket at any member of the data sharing group.

#### **Related tasks**

 [Sending RACF PassTickets \(DB2 Administration Guide\)](#)

### **Example of configuring DB2 Connect to access a subset of a data sharing group**

When you need to configure DB2 Connect to access a subset of a data sharing group, looking at an example configuration can be helpful.

For this example, three aliases are defined, each of which specifies a subset of the data sharing group.

For more detailed information about DB2 Connect, refer to *IBM DB2 Connect User's Guide*, which you can download at <http://www-01.ibm.com/support/docview.wss?rs=71&uid=swg27015148>.

The aliases are associated with ports 5031, 5032, and 5033.

- Group Location: DB2GROUP1, port: 446
- Location ALIAS 1: ALIAS1, port 5031
- Location ALIAS 2: ALIAS2, port 5032
- Location ALIAS 3: ALIAS3, port 5033

One aspect of how the data sharing group is configured is whether dynamic virtual IP addressing (DVIPA) (with or without Sysplex Distributor) is being used. The only way to ensure that an initial connection attempt is made against an available member of the data sharing group that participates in a particular location alias of the group, is to configure:

- Each member of the data sharing group with its own member-specific DVIPA

- The group itself with a distributing DVIPA

The Sysplex Distributor capability of the z/OS Communications Server in a Sysplex environment can instantiate a group distributing DVIPA. Also, the group-distributing DVIPA is configured within the Sysplex to distribute requests to all servers that are listening on ports 446, 5031, 5032, and 5033. For this example, assume that the group distributing DVIPA has a value of Vgrp, which is a character string that represent an IPv4 or IPv6 address. When using the group distributing DVIPA, you do not need to know the member-specific DVIPAs.

The following configuration statements configure DB2 Connect to connect to any member of the group:

```
db2 catalog tcpip node grp1node remote Vgrp server 446
db2 catalog dcs db db2grp1 as db2group1 parms ',,,,,sysplex'
db2 catalog db db2grp1 as db2grp1 at node grp1node authentication server
```

In the preceding configuration statements, a node profile is defined with the group distributing DVIPA, and that node profile is also defined with the group TCP/IP port. You should create additional DB2 Connect node profiles that are configured with the group distributing DVIPA and with each of the locations aliases port values.

The following DB2 Connect configuration statements are required to connect to members of the location aliases:

```
db2 catalog tcpip node alias1 remote Vgrp server 5031
db2 catalog dcs db alias1 as alias1 parms ',,,,,sysplex'
db2 catalog db alias1 as alias1 at node alias1 authentication server

db2 catalog tcpip node alias2 remote Vgrp server 5032
db2 catalog dcs db alias2 as alias2 parms ',,,,,sysplex'
db2 catalog db alias2 as alias2 at node alias2 authentication server

db2 catalog tcpip node alias3 remote Vgrp server 5033
db2 catalog dcs db alias3 as alias3 parms ',,,,,sysplex'
db2 catalog db alias3 as alias3 at node alias3 authentication server
```

When the initial connection is successful to a member of the location or the location alias, a list of available member-specific IP addresses is returned to the DB2 Connect server. When the initial connection is made to a location alias, the IP addresses returned are the member-specific alias IP addresses, if they exist. Because the Sysplex parameter is specified on each of the dcs db profiles, subsequent connection attempts are made to the servers by using the member-specific DVIPAs that are returned.

If you choose not to use the Sysplex Distributor but have configured each member of the data sharing group with a member-specific DVIPA, a group-distributing DVIPA is not available to connect to the various members of the data sharing group. If this is how the group was originally configured, you need to create separate domain names in a DNS that represent the entire group (meaning that DVIPAs of the members must be defined in this domain name). Also, each of the aliases that specifies a subset of data sharing member (DVIPAs for only the members that are defined to part of a subsetting location alias can be defined in a particular domain name). Using this configuration, you can substitute Vgrp in the tcpip node profile named grp1node with the domain name that represents the entire group. For each tcpip node profile that is named ALIAS1, ALIAS2, and ALIAS3, you can substitute Vgrp with the domain name that represents each location alias respectively. If you do not use a group distributing DVIPA, DB2 Connect might attempt to connect to an unavailable member of the group or

location alias, which will result in a TCP/IP connection failure. A connection failure is not returned to the requesting client until connection attempts to each IP address that is defined in the specified domain name have failed.

You might also choose not to use DVIPAs. This choice causes the domain name entries in the DNS to be populated with an IP address of each system that has a member running on it. If a member moves to another system, you need to manually update the DNS entries.

#### Related concepts

➡ Java client support for high availability for connections to DB2 for z/OS servers (Application Programming for Java)

#### Related reference

➡ Non-Java client support for high availability for connections to DB2 for z/OS servers

## Connecting distributed partners in a TCP/IP network

DB2 Connect requesters can be configured to use group access to access remote data sharing groups in TCP/IP networks. DB2 for z/OS requesters can be configured to use group access or member-specific access to access remote data sharing groups in TCP/IP networks.

### Configuring a DB2 Connect requester to use group access

You can configure a DB2 Connect requester to use group access to access remote data sharing groups in TCP/IP networks.

This section outlines the most important configuration steps. For detailed information about these steps, and for complete information about DB2 Connect, refer to the *IBM DB2 Connect User's Guide*, which you can download from <http://www-01.ibm.com/support/docview.wss?rs=71&uid=swg27015148>.

To configure a DB2 Connect requester to use group access:

1. Use the Configuration Assistant to update the database directories that DB2 Connect uses to manage database connection information.

**Important:** You must enable Sysplex support for the data sharing group by specifying the SYSPLEX parameter in the parameter string of the target database name in the DCS directory.

2. If you want applications to be able to update data in multiple remote database servers with guaranteed integrity, use the DB2 Control Center to enable and test multisite updates.
3. Bind the DB2 Connect utilities, and any applications that are developed using embedded SQL, to the databases with which they operate.

The z/OS Sysplex distributor can be used to provide an automated IP layer of connectivity within the Sysplex, but should not be viewed as a replacement of the Connect Server SYSPLEX and concentrator functions. It is a combination of the high availability features of DVIPA, Sysplex distributor, and Connect Server SYSPLEX and concentrator functions that provides the highest availability, best quality of service, and the workload optimization capabilities of WLM. The WLM Routing services and the DB2 weights provided to Connect Server, which are used as routing recommendations, are specific to the DB2 servers. They take several indicators into account, like system load of the member that the DB2 member is

running on, the Performance Index (PI) of the given member or the enclave queue-time waiting for a free database access thread related to the member and others.

In addition, DB2 for z/OS can have health conditions, which are conditions that reduce the ability of the server to handle requests. Usually these conditions cannot be directly realized or measured by WLM, but are only known to the server itself or the application that owns or monitors that server. The health indicator enhancement gives the application the opportunity to inform WLM about health conditions, so that the routing service is able to take them into account when evaluating routing recommendations. DB2 for z/OS storage utilization is used as the indicator of health of the member.

You should use the Sysplex and connection concentrator functions when configuring a DB2 Connect requester. The connection concentrator reduces the resources required on DB2 for z/OS database servers to support large numbers of workstation and web users. The concentrator also allows applications to stay connected without any resources being consumed on the DB2 host server. The database directory should use the group IP address.

The IBM Data Server Driver for JDBC and SQLJ is an architecture-neutral JDBC driver for distributed and local DB2 access. The IBM Data Server Driver for JDBC and SQLJ architecture is independent of any particular JDBC driver-type connectivity or target platform, which allows for both Type 4 and Type 2 connectivity in a single driver instance to DB2 platforms.

#### **Related concepts**

 Java client support for high availability for connections to DB2 for z/OS servers (Application Programming for Java)

### **Configuring a DB2 requester to use group access**

You can update the CDB of a DB2 for z/OS requester to use group access.

Recall that DVIPA network addressing uses dynamic virtual IP addresses (DVIPAs). To enable group access to a remote data sharing group:

1. Identify the DB2 location name of the remote group.  
Insert the group's DB2 location name into the LOCATION column of the requester's SYSIBM.LOCATIONS table.
2. Identify the DVIPA or domain name of the remote group, and specify the security definitions for conversations with group members.  
Insert the group's DVIPA or domain name into the IPADDR column of the requester's SYSIBM.IPNAMES table.
3. Map the DB2 location name of the remote group to its DVIPA or domain name.  
If you use RACF PassTickets, specify the group's generic LU name or IPNAME value as the value of the LINKNAME columns.  
Insert the same link name into the LINKNAME columns of the requester's SYSIBM.LOCATIONS and SYSIBM.IPNAMES tables. Be sure to update only those rows that are associated with the remote group.

This is an example of group access that uses DVIPA network addressing. This example provides sample SQL statements for enabling group access that uses DVIPA network addressing. It also shows the results of those statements in the form of table excerpts. This example assumes that a remote data sharing group exists with a group location name of DB2A and a group DVIPA of Vx.

### SQL statements:

The following SQL statements update the CDB of a DB2 for z/OS requester to use group access (DVIPA network addressing):

1. This statement identifies the DB2 location name of the remote group:

**GUIP**

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, PORT)
VALUES ('DB2A', '446');
```

**GUIP**

2. This statement identifies the DVIPA of the remote group and specifies security definitions for conversations with the group's members:

**GUIP**

```
INSERT INTO SYSIBM.IPNAMES (SECURITY_OUT, USERNAMES, IPADDR)
VALUES ('A', ' ', 'Vx');
```

**GUIP**

3. These statements map the DB2 location name of the remote group to the DVIPA of the remote group:

**GUIP**

```
UPDATE SYSIBM.LOCATIONS
SET LINKNAME='DB2ALINK'
WHERE LOCATION='DB2A';
```

```
UPDATE SYSIBM.IPNAMES
SET LINKNAME='DB2ALINK'
WHERE IPADDR='Vx';
```

**GUIP**

### Table excerpts:

An excerpt of the SYSIBM.LOCATIONS table would look like the table below.

*Table 2. DB2 location name of a remote data sharing group in a DB2 requester's SYSIBM.LOCATIONS table. Not all columns are shown.*

LOCATION	LINKNAME	PORT
DB2A	DB2ALINK	446

An excerpt of the SYSIBM.IPNAMES table would look like the table below.

*Table 3. DVIPA of a remote data sharing group in a DB2 requester's SYSIBM.IPNAMES table. Not all columns are shown.*

LINKNAME	SECURITY_OUT	USERNAMES	IPADDR
DB2ALINK	A		Vx

The following SQL statement connects to the remote group:

```
CONNECT TO DB2A;
```

## Configuring a DB2 requester to use member-specific access

You can update the CDB of a DB2 for z/OS requester to use member-specific access.

To enable member-specific access to a remote data sharing group:

1. Define one or more location aliases that identify different sets of members.

Insert each location alias into the LOCATION column of the requester's SYSIBM.LOCATIONS table.

2. Map each location alias to the security definitions for conversations with each set of members.

If you use RACF PassTickets, then all LOCATION aliases must eventually refer to a single common IPNAMES row whose LINKNAME value must match the remote group's generic LU name or IPNAME value. The IPADDR value in the IPNAMES row must eventually refer to the group distributing DVIPA of the data sharing group.

Insert the same link names into the LINKNAME columns of the requester's SYSIBM.LOCATIONS and SYSIBM.IPNAMES tables. In the SYSIBM.IPNAMES table, insert the appropriate security definitions for your site, and leave the IPADDR column blank.

3. Map each location alias to the domain names or DVIPAs of the data sharing members in that set.

For each member of a set, insert the domain name or DVIPA of the member into the IPADDR column of the requester's SYSIBM.IPLIST table.

4. Define location aliases on the server (remote data sharing group).

For each location alias on the requester's side, you must define a location alias at the server by using the DSNJU003 (change log inventory) utility.

You should disable Sysplex support when configuring a DB2 requester for member-specific access. If you do not disable Sysplex support, DB2 Connect uses DRDA workload balancing by default to allocate requests among the members of the group. The database directory should use the member IP address. For more information on enabling or disabling Sysplex support, see *IBM DB2 Connect User's Guide*, which you can download from <http://www-01.ibm.com/support/docview.wss?rs=71&uid=swg27015148>.

This is an example of member-specific access that uses DVIPA network addressing. This example provides sample SQL statements for enabling member-specific access that uses DVIPAs. It also shows the results of those statements in the form of table excerpts. This example assumes that a remote data sharing group exists with a DB2 location name of DB2A and three members with DVIPAs V1, V2, and V3. Location aliases that are to be defined are DB2B (with members V1 and V2), DB2C (with members V2 and V3), and DB2D (with member V1).

### *SQL statements:*

Use the following SQL statements to update the CDB of a DB2 for z/OS requester to use member-specific access (DVIPA network addressing):

1. These statements define location aliases that identify different sets of members:

#### **GUPI**

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, PORT)
VALUES ('DB2B', '446');
INSERT INTO SYSIBM.LOCATIONS (LOCATION, PORT)
VALUES ('DB2C', '446');
```

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, PORT)
VALUES ('DB2D', '446');
```

**GUIP**

- These statements map each location alias to the security definitions for conversations with each set of members:

**GUIP**

```
UPDATE SYSIBM.LOCATIONS
SET LINKNAME='ALIASB'
WHERE LOCATION='DB2B';
INSERT INTO SYSIBM.IPNames (LINKNAME)
VALUES ('ALIASB')
UPDATE SYSIBM.LOCATIONS
SET LINKNAME='ALIASC'
WHERE LOCATION='DB2C';
INSERT INTO SYSIBM.IPNames (LINKNAME)
VALUES ('ALIASC')
UPDATE SYSIBM.LOCATIONS
SET LINKNAME='ALIASD'
WHERE LOCATION='DB2D';
INSERT INTO SYSIBM.IPNames (LINKNAME)
VALUES ('ALIASD')
```

**GUIP**

- These statements map each location alias to the DVIPAs of the data sharing members in that set:

**GUIP**

```
INSERT INTO SYSIBM.IPLIST (LINKNAME, IPADDR)
VALUES ('ALIASB', 'V1')
INSERT INTO SYSIBM.IPLIST (LINKNAME, IPADDR)
VALUES ('ALIASB', 'V2')
INSERT INTO SYSIBM.IPLIST (LINKNAME, IPADDR)
VALUES ('ALIASC', 'V2')
INSERT INTO SYSIBM.IPLIST (LINKNAME, IPADDR)
VALUES ('ALIASC', 'V3')
INSERT INTO SYSIBM.IPLIST (LINKNAME, IPADDR)
VALUES ('ALIASD', 'V1')
```

**GUIP**

*Table excerpts:*

An excerpt of the SYSIBM.LOCATIONS table would look like the table below.

*Table 4. Location aliases for a remote data sharing group in a DB2 requester's SYSIBM.LOCATIONS table. Not all columns are shown.*

LOCATION	LINKNAME	PORT
DB2B	ALIASB	446
DB2C	ALIASC	446
DB2D	ALIASD	446

Location aliases are not associated with the DVIPA of a remote data sharing group in a DB2 requester's SYSIBM.IPNAMES table. An excerpt of the SYSIBM.IPNAMES table would look like the table below.

*Table 5. Excerpt of the SYSIBM.IPNAMES table. Not all columns are shown.*

LINKNAME	IPADDR
ALIASB	
ALIASC	
ALIASD	

An excerpt of the SYSIBM.IPLIST table would look like the table below.

*Table 6. DVIPAs of the members that are associated with each location alias in a DB2 requester's SYSIBM.IPLIST table. Not all columns are shown.*

LINKNAME	IPADDR
ALIASB	V1
ALIASB	V2
ALIASC	V2
ALIASC	V3
ALIASD	V1

The following SQL statement connects to the remote group using a location alias:  
CONNECT TO DB2C;

---

## SNA access methods

*Systems Network Architecture (SNA)* is a proprietary IBM architecture that describes the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks.

SNA contains several functional layers and includes the following components:

- An application programming interface (API) called Virtual Telecommunications Access Method (VTAM®)
- A communications protocol for the exchange of control information and data
- A data link layer called Synchronous Data Link Control (SDLC)

SNA also includes the concept of nodes, which can contain both physical units that provide certain setup functions, and logical units (LUs), each of which is associated with a particular network transaction.

A data sharing group can simultaneously serve requesters that use member-routing access, group-generic access, and single-member access. As a server, the group is flexible about the access methods that requesters use. A requester can use member-routing access in one session and use group-generic access in another session. However, as a requester, the group always uses the same access method when communicating with a particular server.

**Recommendation:** Configure a data sharing group to use member-routing access whenever possible, unless your requester does not support it. It provides better workload balancing and two-phase commit support than group-generic access.

In SNA networks, each member of a data sharing group has its own LU name (in addition to the net ID that the member obtains from VTAM when the DDF starts). You can also define a *generic LU name* that represents all the members in the group.

An SNA requester can use one of several access methods to connect to a data sharing group:

#### **Member-routing access**

A requester can define a server location name that is associated with many LU names. Unlike with the generic LU name, a requester can establish sessions with one or more subsystems in the group. For member routing, the requests are distributed to members of the data sharing group based on their capacity.

#### **Group-generic access**

A requester uses the group's generic LU name to make an initial connection to any member of the group. With this access method, VTAM chooses one of the group members and establishes a session with that member on behalf of the requester. Subsequent queries from the requester are also directed to that same member.

#### **Single-member access**

A requester uses a member's LU name to connect and send queries to a single member of the group. All connection and query requests from that requester are directed to the same member.

## **Member-routing access**

To access any member of a remote data sharing group with member routing, the location name of the data sharing group is associated with a list of LUNAMEs of the group members.

A subset of the member LUNAMEs can comprise the list to limit which members of the group can be accessed by the requester.

Requesters can use an LULIST to bypass VTAM workload balancing by establishing sessions with one or more members. Workload is balanced among members at the requester's discretion.

With member-routing access, a requester uses an LULIST to make an initial connection to one of the members that is represented by the alias. The DB2 Sysplex transaction program (DB2 STP) works in conjunction with the z/OS Workload Manager (WLM) to return a list of members that are currently active and can perform work. It also returns a weight for each active member that indicates the member's current capacity. The requester uses this information to connect to the member or members with the most capacity. If the LULIST contains a subset of the members, only the members of that subset can be accessed.

## **Two-phase commit resynchronization**

If two-phase commit resynchronization is necessary, due to previous communication or system failure, the resynchronization request is sent to the member that is involved in the failure. DB2 performs resynchronization asynchronously, so that requesters are not prevented from establishing new sessions with other members.

## RACF PassTicket limitation

If you use RACF PassTickets, you can define only one location alias. The generic LU name is used to generate a valid PassTicket for the data sharing group. Generation of a valid PassTicket for individual members or sets of members in a group is not possible.

### Related concepts

 [Two-phase commit process \(DB2 Administration Guide\)](#)

## Group-generic access

With group-generic access, a requester uses the generic LU name of the group to make an initial connection to any member of the group.

VTAM then chooses one of the group members and establishes a session with that member on behalf of the requester. The requester uses the LU name of the chosen member when it makes subsequent requests.

VTAM provides workload balancing at the session level by allocating each connection request to the member with the fewest number of sessions at the time of connection. VTAM determines which member has the most capacity based on either the number of each member's active DDF sessions or the result of a user-written VTAM or z/OS Workload Manager exit routine.

After a connection is established between a requester and a member, all requests from the same requester that are made during the session are directed to the same member. VTAM does not reassess a member's capacity at the time of subsequent requests. Only after all connections between the requester and member are closed, can VTAM choose a different member to process future requests from the requester.

**Exception:** If the connection between the requester and the member is enabled for two-phase commit processing, VTAM continues to direct all requests from this requester to the same member. The mapping between the requester LU and the member LU is preserved until the DB2 command RESET GENERICLU is issued.

If a requester is active at the time of a communication or system failure, VTAM handles the reconnection in one of two ways:

- If the pre-failure session was enabled for two-phase commit support, VTAM reconnects the requester to the same member.  
If that member is unavailable, a communication error is returned.
- If the pre-failure session was not enabled for two-phase commit support, VTAM can connect the requester to any member.

Because VTAM balances the workload based on the number of currently established sessions rather than on true capacity, you should carefully evaluate this access method before implementing it. If the number of sessions is an accurate reflection of your workload, this method can be a good choice. It has the added benefit of being relatively easy to set up.

## Single-member access

Single-member access using SNA is the same method that is used to access DB2 in non-data sharing environments.

With single-member access, a requester specifies the LU name of the specific member of the group as the value of the LINKNAME column of both a SYSIBM.IPNAMES row and a SYSIBM.LOCATIONS row. The requester sends all connection and query requests to that member.

With single-member access, a requester uses the LU name in its TCP/IP configuration to connect to a specific member of the group. The requester sends all connection and query requests to this same member.

**DB2 for z/OS:** Configure DB2 for z/OS to use single-member access by creating and using a location alias that represents only one member of the data sharing group. Otherwise, DB2 for z/OS uses DRDA workload balancing by default to allocate requests among the members of the group.

**Recommendation:** Do not use single-member access for the following reasons:

- It is dependent on the specified member being operational.
- It provides no workload balancing for requesters that cannot perform DRDA workload balancing.

**Restriction:** Version 8 of DB2 Connect does not support single-member access.

## Setting up DB2 for z/OS as a requester

Much of the processing in a distributed database environment requires the exchange of messages with other locations in the network.

When DB2 acts as a requester, it can connect applications that run on the system to remote database servers including DB2 for z/OS, DB2 for i, and DB2 for VSE & VM.

In its role as a requester, DB2 for z/OS accepts DB2 location names and translates them into SNA NETID.LUNAMES. It uses the communications database (CDB) to register DB2 location names and their corresponding network parameters. The data that is stored in the CDB enables DB2 to pass the required information when making distributed database requests over SNA connections.

For this processing to work correctly, you must:

1. Define the DB2 for z/OS requester to the local VTAM system.
2. Identify the remote data sharing groups to which applications can connect.

### Related tasks

 [Connecting systems with VTAM \(DB2 Installation and Migration\)](#)

## Remote data sharing group requirements

This section describes the requirements of member-specific, group-generic, and single-member access.

When an application requests data from a remote subsystem, the DB2 for z/OS requester searches the CDB for information about the remote group. Recall that DB2 uses the CDB to store information about how to communicate with remote groups, and that the DDF uses the CDB to map DB2 location names to LU names.

Member-specific access requires:

- Location entries in the SYSIBM.LOCATIONS table
- Conversational security requirements entries in the SYSIBM.LUNAMES table

- LUNAMES of the members to be accessed in the SYSIBM.LULIST table

Group-generic access requires:

- A DB2 location name entry in the SYSIBM.LOCATIONS table
- A conversational security requirements entry in the SYSIBM.LUNAMES table

Single-member access requires:

- A location entry in the SYSIBM.LOCATIONS table. If you plan to allow group access concurrently with single-member or member-specific access, then the requester should use a server-defined location alias name in the SYSIBM.LOCATIONS table.
- A conversational security requirements entry in the SYSIBM.LUNAMES table
- The LUNAME of a single member in the SYSIBM.LULIST table

### **SYSIBM.LOCATIONS:**

SYSIBM.LOCATIONS maps the DB2 location names in connection requests to the VTAM LU names of remote systems and, if necessary, transaction program names (TPNs).

SYSIBM.LOCATIONS must contain at least one row for each remote group, depending on the access method that is used.

- For member-routing access, the LOCATION column of the row contains the group's location name, and the LINKNAME column of the row should contain the group's generic LU name.
- For group-generic access, the LOCATION column of the row contains the group's DB2 location name and the LINKNAME column of the row contains that group's generic LU name.
- For single-member access, the LOCATION column of the row contains a location alias that identifies one member of the group.

### **SYSIBM.LUNAMES:**

SYSIBM.LUNAMES maps DB2 location names to the LU names of remote systems. It also maps DB2 location names to the security and mode requirements for conversations with those systems.

SYSIBM.LUNAMES must contain at least one row for each remote group, depending on the access method that is used.

- For member-routing access, the LINKNAME column of the row should contain the group's generic LU name.
- For group-generic access, the LUNAME column of the row contains the group's generic LU name.
- For single-member access, the LINKNAME column of the row contains a link name.

The GENERIC column of SYSIBM.LUNAMES is for the specific use of data sharing group members that act as requesters. Its value indicates whether a requester uses the group's generic LU name or its own real LU name when identifying itself to a remote subsystem. If you want a requester to use the group's generic LU name, specify a value of Y; if you want a requester to use its own LU name, specify a value of N.

**Important:** The value of the GENERIC column is ignored if the DB2 GENERIC LUNAME parameter on installation panel DSNTIPR is blank. In this case, DB2 requesters are identified to remote systems by their own LU names.

Be aware that only one member of a data sharing group can access a remote system by using the group's generic LU name. If one member of a group is already connected to a remote system and is using the group's generic LU name, subsequent connections to that remote system by other members of the same group use the requesters' own LU names.

#### **SYSIBM.LULIST:**

SYSIBM.LULIST supports member-routing access to a remote data sharing group by enabling you to associate a list of LU names with one or more members of the group.

SYSIBM.LULIST must contain a row for every member that the requester is allowed to access.

- For member-routing access, the LUNAME column of each row contains a member's LU name.
- For single-member access, the LUNAME column of the row contains the member's LU name.

**Example:** Assume that a remote data sharing group has six members. You might want the requester to be able to access only three members of the group. In this example, you insert three rows into the SYSIBM.LULIST table for the members to be accessed.

#### **How DB2 sends requests:**

When sending a request, DB2 uses the value in the LINKNAME column of the SYSIBM.LOCATIONS table to determine which network protocol to use.

- If DB2 finds that same value in the LINKNAME column of the SYSIBM.IPNAMES table, it uses TCP/IP.
- If DB2 find that same value in the LUNAME column of the SYSIBM.LUNAMES table, it uses SNA.
- If DB2 finds that same value in both SYSIBM.IPNAMES and SYSIBM.LUNAMES, it uses TCP/IP.

#### **Updates to the communications database for SNA connections:**

You can update the CDB while DDF is active.

Changes to the SYSIBM.LOCATIONS, SYSIBM.LUNAMES, and SYSIBM.LULIST table take effect in the following manner for SNA connections:

- If the DDF has not yet tried to communicate with a particular remote group, updates take effect when DDF attempts to communicate with that group.
- If the DDF has already attempted to communicate with a particular remote group, updates take effect take the next time DDF is started.
- Updates do not affect existing conversations; existing conversations continue to operate as if the updates had not occurred.

## Configuring data sharing groups as servers

DRDA requesters can access any member of a data sharing group through SNA if the SYSIBM.LUNAMES table of the communications database contains a default row that enables access by any requester that is not defined by a row in the table.

Installation job DSNTIJSJG normally creates a default row unless the installation process has been altered to remove that step. If the SYSIBM.LUNAMES table does not contain a default row, you must, for each DRDA SNA requester, insert a row that contains the LUNAME of that requester.

### Related concepts

“Configuring data sharing groups as servers”

## Configuring data sharing groups for group-generic access

You can configure a DB2 server for group-generic access to make requests and handle requests.

To enable group-generic access to a data sharing group:

1. Include information in the coupling facility for support of VTAM generic resources (the ISTGENERIC structure).
2. Define a generic LU name for the group.
3. Configure the members to use the group's generic LU name when identifying themselves to remote subsystems.
4. Identify the generic LU names of requesting data sharing groups.
5. Identify the LU names of individual requesters, such as DB2 Connect and members of requesting data sharing groups.

### Related reference

 [Generic resources \(VTAM Network Implementation Guide\)](#)

### Related information

 [IBM eServer zSeries Processor Resource/Systems Manager Planning Guide](#)

### Defining a generic LU name for the group:

Define a generic LU name for the data sharing group by specifying the name for each member in the DB2 GENERIC LUNAME parameter on installation panel DSNTIPR.

A generic LU name can be one to eight characters in length, and it is stored in each member's bootstrap data set (BSDS).

*Using the originating member's LU name:*

Using the LU name of the originating member of the group as the generic LU name, and changing the originating member's LU name can be useful. This is particularly true when the originating member is already acting as a server for applications that will migrate to the data sharing group.

By configuring the group to use the originating member's LU name, you avoid needing to make extensive changes to the CDBs of DB2 for z/OS requesters.

To change the LU name of the originating member of the group, you must change both the LU name value in the member's VTAM APPL statement and the LUNAME value in the member's bootstrap data set (BSDS).

For example, assume that a group is formed whose originating member is already defined to the requesters that are shown in the following figure. Until you define a generic LU name when you enable group-generic access, requesters can access DB2A's data only through the originating member (LUDB2A). If the originating member is unable to handle requests, requesters are unable to access DB2A's data.

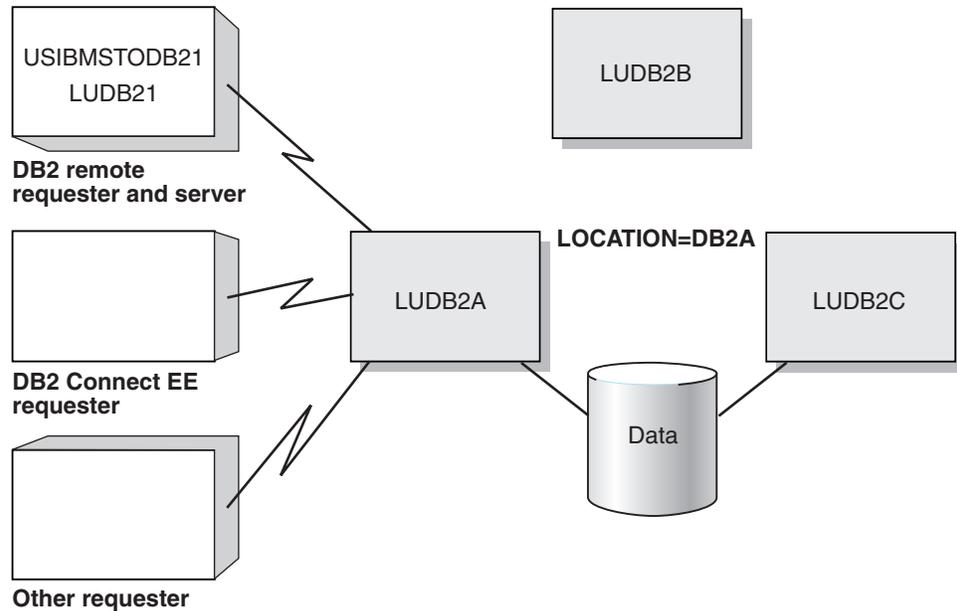


Figure 19. Example configuration before enabling group-generic access.. Access is limited to a single member of the data sharing group.

As shown in the following figure, by enabling group-generic access and by using the LU name of the originating member as the generic LU name, requesters can, with minimal change, access DB2A's data from any member of the group.

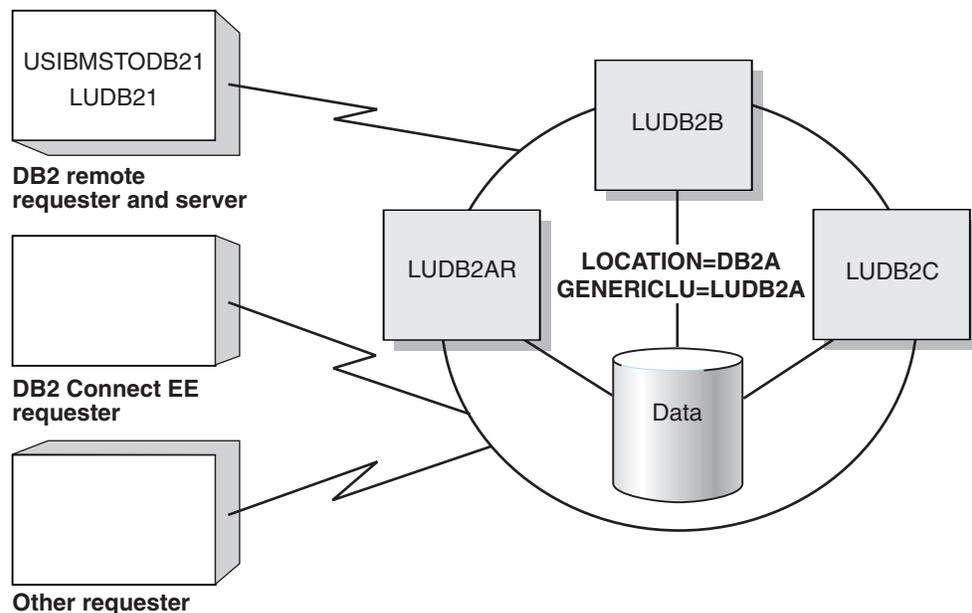


Figure 20. Example configuration after enabling group-generic access.. All members of the group can potentially handle requests.

### Configuring members to use the group's generic LU name:

Specify a value of Y in the GENERIC column of each row of the SYSIBM.LUNAMES table that corresponds to a remote subsystem with which the data sharing group communicates.

The GENERIC column of the SYSIBM.LUNAMES table is for the specific use of group members that act as requesters. Its value indicates whether a member uses the group's generic LU name or its own real LU name when identifying itself to a remote subsystem. The remote subsystem must be able to recognize the generic LU name.

**Important:** The value of the GENERIC column is ignored if the DB2 GENERIC LUNAME parameter on installation panel DSNTIPR is blank.

Be aware that only one member of a data sharing group can access a remote subsystem by using the group's generic LU name. If one member of a group is already connected to a remote subsystem and is using the group's generic LU name, subsequent connections to that remote subsystem by other members of the same group use the requesters' own LU names. If the remote subsystem only accepts generic LU names, all requests from the members of the requesting group must be routed through one member.

**Clarification:** The GENERIC column does **not** determine whether a member uses group-generic or member-specific access. The rows, or lack of rows, in SYSIBM.LULIST determine whether a member uses group-generic or member-specific access.

If the remote subsystem starts CNOS processing first, VTAM uses the name with which the remote subsystem connected, whether this is the real LU name or the generic LU name. Because this behavior is not always predictable, the subsystem that is handling requests from the data sharing group should be able to accept either the generic LU name or the real LU name when group-generic processing is used.

**Example:** Assume that the LU names of two group members are LUDB2NY and LUDB2LA. An excerpt of the SYSIBM.LUNAMES table would look similar to the following table.

*Table 7. SYSIBM.LUNAMES table of a requesting data sharing group that uses group-generic access. Not all columns are shown.*

LUNAME	GENERIC
LUDB2NY	Y
LUDB2LA	Y

Use the following SQL statements to update the SYSIBM.LUNAMES table to specify that members use their group's generic LU name when identifying themselves to remote data sharing groups:

#### GUPI

```
UPDATE SYSIBM.LUNAMES
  SET GENERIC='Y'
  WHERE LUNAME='LUDB2NY';
```

```
UPDATE SYSIBM.LUNAMES
SET GENERIC='Y'
WHERE LUNAME='LUDB2LA';
```

**GUIP**

**Identifying the generic LU names of requesting data sharing groups:**

Specify the generic LU names of requesting data sharing groups in the LUNAME column of the server's SYSIBM.LUNAMES table.

**Example:** Assume that the generic LU names of two groups are LUDSG1 and LUDSG2. An excerpt of the server's SYSIBM.LUNAMES table would look similar to the following table.

*Table 8. Generic LU names of requesting data sharing groups in a DB2 server's SYSIBM.LUNAMES table. Not all columns are shown.*

LUNAME	GENERIC
LUDSG1	
LUDSG2	

**GUIP** Use the following SQL statements to populate SYSIBM.LUNAMES with the generic LU names of requesting data sharing groups:

This statement inserts the generic LU names of requesting data sharing groups into the SYSIBM.LUNAMES table:

```
INSERT INTO SYSIBM.LUNAMES (LUNAME)
VALUES ('LUDSG1');
INSERT INTO SYSIBM.LUNAMES (LUNAME)
VALUES ('LUDSG2');
```

**GUIP**

**Specifying the LU names of requesters:**

You can specify the LU names of requesters in the LUNAME column of the SYSIBM.LUNAMES table of the server.

Installation job DSNTIJSJG inserts a default row into the SYSIBM.LUNAMES table unless the installation process has been altered to remove this step. The default row enables access by any requester that is not defined by a row in the table. You can change the values in the default row before you start DDF. The following table shows the default row in the SYSIBM.LUNAMES table of a DB2 server

*Table 9. An example of the default row in the SYSIBM.LUNAMES table of a DB2 server. This example shows only those columns that are accessed when DB2 is a server.*

LUNAME	SYSMODENAME	SECURITY_IN	GENERIC
(8 blanks)	'A'	'N'	

To limit the number of conversations that a DB2 server can accept from a DRDA SNA LU partner, change or add the row for the partner LU in the

SYsIBM.LUNAMES table to specify an SNA MODENAME. You must also add or update a row in the SYsIBM.LUMODES table that contains the SNA MODENAME.

To block inbound already-verified conversations from another partner LU name, change or add a row for the LU name to specify SECURITY\_IN of 'V'.

To perform inbound userid translation from a partner LU name, update or add a row in the SYsIBM.LUNAMES table to specify 'T' or 'B' in the USERNAMES column. You must also update entries in the SYsIBM.USERNAMES table.

Individual requesters include DB2 for z/OS and the members of requesting data sharing groups.

**Example:** Assume that the LU names of the members of one requesting data sharing group are LUMEM1, LUMEM2, and LUMEM3, and that the LU names of the members of another requesting data sharing group are LUMEMA and LUMEMB. An excerpt of the server's SYsIBM.LUNAMES table would look similar to the following table.

*Table 10. LU names of requesters in a SYsIBM.LUNAMES table of a DB2 server. Not all columns are shown.*

LUNAME	SYSMODENAME	SECURITY_IN	USERNAMES	GENERIC
LUMEM1				
LUMEM2				
LUMEM3				
LUMEMA				
LUMEMB				

**GUPI** Use the following SQL statements to populate the SYsIBM.LUNAMES table with the LU names of requesters.

The following statements insert the LU names of requesters into the SYsIBM.LUNAMES table:

```
INSERT INTO SYsIBM.LUNAMES (LUNAME)
VALUES ('LUMEM1');
INSERT INTO SYsIBM.LUNAMES (LUNAME)
VALUES ('LUMEM2');
INSERT INTO SYsIBM.LUNAMES (LUNAME)
VALUES ('LUMEM3');
INSERT INTO SYsIBM.LUNAMES (LUNAME)
VALUES ('LUMEMA');
INSERT INTO SYsIBM.LUNAMES (LUNAME)
VALUES ('LUMEMB');
```

**GUPI**

## Connecting distributed partners in an SNA network

You can configure DB2 requesters to use member-routing access, group-generic access, or member-specific access to access remote data sharing groups in SNA networks.

For information about how to configure other types of requesters, consult the product documentation for your specific requester.

## Configuring a DB2 requester to use member-routing access

You can update the CDB of a DB2 requester to use member-routing access.

To enable member-routing access to a remote data sharing group:

1. Insert the group's location name into the LOCATION column of the requester's SYSIBM.LOCATIONS table.

If you use RACF PassTickets, define only one location row.

2. Map the location to the security and mode requirements for conversations with each set of members.

Insert the same link name into the LINKNAME column of the requester's SYSIBM.LOCATIONS and SYSIBM.LUNAMES tables. Insert the appropriate security and mode requirements for your site.

If you use RACF PassTickets, specify the group's generic LU name as the value of the LINKNAME columns.

3. Map the location to the LU names of the data sharing members to be accessed.

For each member, insert the LU name of the member into the LUNAME column of the requester's SYSIBM.LULIST table.

This is an example of member-routing access. This example provides sample SQL statements for enabling member-routing access. It also shows the results of those statements in the form of table excerpts. It assumes that a remote data sharing group exists with a DB2 location name of DB2A and a generic LU name of LUDB2A, and three members with names LUDB2AR, LUDB2B, and LUDB2C. Only members DB2B and DB2C with LU names LUDB2B and LUDB2C, respectively, are to be accessed.

*SQL statements:*

Use the following SQL statements to update the CDB of a DB2 for z/OS requester to use member-routing access:

1. This statement defines the location and uses the group generic LU name to identify different sets of members:

**GUIP**

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
VALUES ('DB2A', 'LUDB2A');
```

**GUIP**

2. This statement defines the location's security and mode requirements for conversations with the members:

**GUIP**

```
INSERT INTO SYSIBM.LUNAMES (LUNAME)
VALUES ('LUDB2A');
```

**GUIP**

3. These statements map the location name to the two members of the group to be accessed:

**GUIP**

```

INSERT INTO SYSIBM.LULIST (LINKNAME, LUNAME)
VALUES ('LUDB2A', 'LUDB2B');
INSERT INTO SYSIBM.LULIST (LINKNAME, LUNAME)
VALUES ('LUDB2A', 'LUDB2C');

```



*Table excerpts:*

An excerpt of the SYSIBM.LOCATIONS table would look like the table below.

*Table 11. Location aliases for a remote data sharing group in a DB2 requester's SYSIBM.LOCATIONS table. Not all columns are shown.*

LOCATION	LINKNAME	TPN
DB2A	LUDB2A	

An excerpt of the SYSIBM.LUNAMES table would look like the table below.

*Table 12. Location aliases in a DB2 requester's SYSIBM.LUNAMES table. Not all columns are shown.*

LUNAME	GENERIC
LUDB2A	

An excerpt of the SYSIBM.LULIST table would look like the table below.

*Table 13. LU names of the members that are associated with each location alias in a DB2 requester's SYSIBM.LULIST table. Not all columns are shown.*

LINKNAME	LUNAME
LUDB2A	LUDB2B
LUDB2A	LUDB2C

### Related concepts

“Update the BSDS with the DSNJU003 utility” on page 74

## Configuring a DB2 requester to use group-generic access

You can update the CDB of a DB2 requester to use group-generic access.

To enable group-generic access to a remote data sharing group:

1. Identify the generic LU name of the remote group.
 

Insert the group's generic LU name into the LUNAME column of the requester's SYSIBM.LUNAMES table. Do not include in this table the real LU names of the members of the group.
2. Map the generic LU name to the group's DB2 location name.
 

Insert the generic LU name and the DB2 location name of the group into the LINKNAME and LOCATION columns of the requester's SYSIBM.LOCATIONS table.

This is an example of group-generic access. This example provides sample SQL statements for enabling group-generic access. It also shows the results of those statements in the form of table excerpts. It assumes that a remote data sharing group exists with a DB2 location name of DB2R and a generic LU name of LUDSGA.

SQL statements:

Use the following SQL statements to update the CDB of a DB2 for z/OS requester to use group-generic access:

1. This statement identifies the group's generic LU name:

**GUIP**

```
INSERT INTO SYSIBM.LUNAMES (LUNAME, GENERIC)
VALUES ('LUDSGA', 'Y');
```

**GUIP**

2. This statement maps the group's generic LU name to the group's DB2 location name.

**GUIP**

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
VALUES ('DB2R', 'LUDSGA');
```

**GUIP**

Table excerpts:

An excerpt of the SYSIBM.LUNAMES table would look like the table below.

Table 14. Generic LU name of a remote data sharing group in a DB2 requester's SYSIBM.LUNAMES table. Not all columns are shown.

LUNAME	GENERIC
LUDSGA	Y

An excerpt of the SYSIBM.LOCATIONS table would look similar to the table below.

Table 15. Mapping the generic LU name of a remote data sharing group to its DB2 location name in a DB2 requester's SYSIBM.LOCATIONS table. Not all columns are shown.

LOCATION	LINKNAME	TPN
DB2R	LUDSGA	

## Switching from group-generic to member-specific access

To switch from group-generic to member-specific access, you must break the affinity between systems.

Member-specific access provides several benefits over group-generic access, including better workload balancing and two-phase commit support. However, before a requester that was using group-generic access can take advantage of these benefits by switching to member-specific access, you need to break any affinity that might exist between the requester and the server. This section describes how to break the affinity between systems.

If a requester uses group-generic access to connect to a data sharing group with two-phase commit support enabled on both subsystems, VTAM records information in the coupling facility about which member of the group is involved in the communication. This information is required to ensure that future VTAM sessions are directed to the same group member, thus providing access to the

correct member log for resolution of indoubt threads. When switching the requester from using group-generic access to member-specific access, you must break this affinity between the systems.

To break the affinity between systems:

1. Shut down the network connections between the requester and the data sharing group.

Some ways to do this include:

- Using the VTAM command `VARY NET,INACT,ID=luname`
- Entering the DB2 command `STOP DDF`

2. From an active member of the data sharing group, issue the `RESET GENERICCLU` command.

Issue this command from the member with the VTAM affinity to the requester whose information is being deleted.

- If the requester is configured to pass its own, real LU name to the data sharing group, *netid* is the net ID of the requester, and *luname* is the real LU name of the requester.
- If the requester is configured to pass its generic LU name, *netid* is the net ID of the requesting data sharing group, and *luname* is the generic LU name of the requesting data sharing group.

3. Change the requester's CDB to ensure that the requester uses member-specific access from this point forward.

- Populate the requester's `SYSIBM.LULIST` table.

Make sure that the `GENERIC` column of the requester's `SYSIBM.LUNAMES` table contains a value of `N` for the row that is associated with the data sharing group.

- Delete existing generic LU names from the `SYSIBM.LUNAMES` table.

4. Re-enable the network connections between the requester and the data sharing group.

Unlike group-generic access, which uses generic LU names, member-specific access uses real LU names. When switching the requester from using group-generic access to member-specific access, remember to delete existing generic LU names.

#### Related tasks

“Configuring a DB2 requester to use member-routing access” on page 70

#### Related reference

 `-RESET GENERICCLU (DB2) (DB2 Commands)`

---

## Preventing a member from processing requests

Transparently to end users, you can prevent one or more members of a data sharing group from handling DDF requests while still letting those members make DDF requests.

You can prevent a member from handling DDF requests in one of two ways:

- Set the `MAX REMOTE ACTIVE` option of installation panel `DSNTIPE` to zero for that member.
- Dynamically update the `DSN6SYSP` macro's `MAXDBAT` parameter value to zero for that member.

The MAX REMOTE ACTIVE option and the MAXDBAT parameter specify the maximum number of database access threads (DBATs) that can be active concurrently. By setting the value of either to zero, you restrict DDF server activity on the affected member. Subsequent connection requests are directed to those members whose MAX REMOTE ACTIVE option and MAXDBAT parameter value are greater than zero. Any work that is already in progress by the affected member continues, but new requests are directed to other members of the data sharing group.

The effects of setting the value of the MAX REMOTE ACTIVE option or the MAXDBAT parameter to zero are:

- DDF does not register the member with z/OS Workload Manager (WLM) for member-specific access. If the member is already registered with WLM, the member is unregistered. The member can continue to use WLM for setting priorities on work, but the member's name is not included on the list that is returned on initial connections to requesters that use member-specific access. Therefore, DDF requests are never routed to that member.
- In an SNA network, DDF does not register the member's LU name with the group's generic LU name during DDF startup.
- In a TCP/IP network, DDF does not listen on the DRDA port.

---

## Update the BSDS with the DSNJU003 utility

DSNJU003 can be used to update the information stored in the bootstrap data sets of members of data sharing groups.

You can use the DSNJU003 (change log inventory) utility to update the following information related to DB2 data sharing. This information is stored in the bootstrap data sets (BSDSs) of members of data sharing groups:

- Distributed Data Facility (DDF)  
Updates the LOCATION, and other DDF related information, in the BSDS.  
If you use this statement to insert new values into the BSDS, you must include at least the SNA, TCP/IP information, or both, and any elements you want to remove from the BSDS in the DDF statement. To update an existing set of values, you need to include only those values that you want to change. The DDF record cannot be deleted from the BSDS after it has been added, it can only be modified.
- DRDA port (PORT=*port*)  
The DRDA port is the TCP/IP port number that is used by the DDF to accept incoming connection requests. 446 is the recommended DRDA port.  
If you change this value for one member of the group, you must change it for all members. DB2 requires that all members of a data sharing group use the same, well-known port number to receive incoming connection requests.
- Generic LU name (GENERIC=*gluname*)  
The generic LU name represents all the members of a data sharing group, in either an SNA network or a TCP/IP network.  
If you change this value for one member of the group, you must change it for all members. All members of a data sharing group must specify the same name.
- GRPIPV4  
Identifies a constant IPV4 address to be associated with the data sharing group for which this DDF is a member.

It is used for the purposes of accepting incoming connection requests that can be serviced by any member of the data sharing group. This address must be entered in dotted decimal form. An associated IPV4 subsystem/member address (see IPV4) must also be specified in order to identify the IP address associated to this specific member of the group. If an IP address is not specified, DB2 will automatically determine the IP address from TCP/IP. It is strongly recommended that you refer to a Sysplex distributor owned distributing dynamic virtual IP address (DVIPA).

- GRPIPV6

Identifies a constant IPV6 address to be associated with the data sharing group for which this DDF is a member.

It is used for the purposes of accepting incoming connection requests that can be serviced by any member of the data sharing group. This address must be entered in colon hexadecimal form. An associated IPV6 subsystem/member address (see IPV6) must also be specified in order to identify the IP address associated to this specific member of the group. If an IP address is not specified, DB2 will automatically determine the IP address from TCP/IP. It is strongly recommended that you refer to a Sysplex distributor owned distributing dynamic virtual IP address (DVIPA).

- IPV4

Identifies a constant IPV4 address to be associated to DDF for the purposes of accepting incoming connection requests to this specific subsystem only.

This address must be entered in dotted decimal form. If an IP address is not specified, DB2 will automatically determine the IP address from TCP/IP. When DB2 is a member of a data sharing group, it is strongly recommended that you refer to a dynamic virtual IP address (DVIPA). A group IP address should also be specified.

- IPV6

Identifies a constant IPV6 address to be associated to DDF for the purposes of accepting incoming connection requests to this specific subsystem only.

This address must be entered in colon hexadecimal form. If an IP address is not specified, DB2 will automatically determine the IP address from TCP/IP. When DB2 is a member of a data sharing group, it is strongly recommended that you refer to a dynamic virtual IP address (DVIPA). A group IP address should also be specified.

- Location aliases (ALIAS=*alias-name:alias-port:alias-secport*)

Location aliases can represent one, several, or all members of a data sharing group for member specific access, to enable clients to continue to connect to a member's old location name, and for defining subsets of data sharing groups.

Location aliases can be defined without an *alias-port* or *alias-secport* value. Clients that use a location alias without an *alias-port* or *alias-secport* value receive information only about the member that processes the connection request in the list of servers. To distribute work requests across the members or a subset of a group, clients must be configured to use the DB2 group location name or location alias that is defined for a subset of members. For location aliases that are defined by the DNJU003 utility, *alias-port* and *alias-secport* values are ignored in non-data sharing environments.

**Important:** You can define and manage only eight static location aliases by using DSNJU003 utility. However, you can also define additional dynamic location aliases by using the MODIFY DDF command. When a static alias is defined with the same name as an existing dynamic alias, DB2 uses only the

dynamic alias. Dynamic location aliases, unlike those defined by the DSNJU003 utility, can be added, modified, and deleted without stopping and restarting DB2 or DDF.

- Resynchronization port (`RESPORT=resport`)

The resynchronization port is the TCP/IP port number that is used by the DDF to accept incoming DRDA two-phase commit resynchronization requests.

DB2 requires that each member of a data sharing group in a TCP/IP network have a resynchronization port number that is unique within the Parallel Sysplex. In the event of a failure, this unique port number allows a requester to reconnect to the correct member so that units of work that require two-phase commit can be resolved.

- Secure port (`SECPORT=secport`)

The secure port identifies the TCP/IP port number that is used by DDF to accept inbound secure DRDA connection requests.

This value must be a decimal number between 0 and 65535, including 65535; zero indicates that DDF's secure connection support for TCP/IP is deactivated. If the secure port has not been specified, remote connections can still use the DRDA PORT, and use SSL on it, but DDF will not validate if the connection utilizes SSL protocol or not.

**Important:** Before updating any of the information related to DB2 data sharing, have all members check for the existence of indoubt threads. If any indoubt threads exist, resolve them before making any updates. To check for indoubt threads, use the `DISPLAY THREAD` command.

#### Related tasks

 [Initializing a TCP stack for use with both IPv4 and IPv6 addresses \(DB2 Installation and Migration\)](#)

#### Related reference

 [-DISPLAY THREAD \(DB2\) \(DB2 Commands\)](#)

 [DSNJU003 \(change log inventory\) \(DB2 Utilities\)](#)

---

## Chapter 7. Operating with data sharing

Most data sharing operations are accomplished by using commands to DB2 for z/OS.

---

### Commands for data sharing environments

As you begin to operate your data sharing environment, you should understand how commands work in a data sharing environment.

#### Command routing

You can control operations on an individual member of a data sharing group from any z/OS console by entering commands prefixed with the appropriate command prefix.

For example, assuming you chose -DB1A as the command prefix for member DB1A, you can start a DB2 statistics trace on that member by entering this command at any z/OS console in the Parallel Sysplex:

**GUIP**

```
-DB1A START TRACE (STAT)
```

**GUIP**

Command routing requires that the command prefix scope is registered as S or X on the IEFSSNxx parmlib member.

You can also control operations on certain objects by using commands or command options that affect an entire group. These, also, can be entered from any z/OS console. For example, assuming that DB1A is active, you can start database XYZ by entering this command at any z/OS console in the Parallel Sysplex:

**GUIP**

```
-DB1A START DATABASE (XYZ)
```

**GUIP**

#### Related tasks

 Registering the command prefixes, member group attachment name, and subgroup attachment name (DB2 Installation and Migration)

#### Command scope

Many commands that are used in a data sharing environment affect only the member for which they are issued.

The breadth of a command's impact is called the *scope* of that command.

For example, a STOP DB2 command stops only the member identified by the command prefix. Such commands have *member scope*.

Other commands have *group scope* because they affect an object in such a way that all members of the group are affected. For example, a STOP DATABASE command, issued from any member of the group, stops that database for all members of the group.

#### Related reference

 Scope of DB2 and related commands (DB2 Commands)

## Commands issued from application programs

You can enter commands from an application program that is attached to a DB2 subsystem through any of the attachment facilities: IMS, CICS, TSO, CAF, and RRSAP.

Commands that are entered in this way are executed by the DB2 subsystem to which the application program is attached. The application cannot send a command to a different DB2 subsystem.

## Command authorization

Data sharing does not introduce any new techniques for establishing and checking authorization IDs.

Because all members of a data sharing group share the DB2 catalog, any ID has the same privileges and authorities on every member. It is your responsibility to use the same connection or sign-on exit routines on every member of the data sharing group to avoid authorization anomalies.

## Where messages are received

You can receive messages from all members at a single console.

Hence, a message must include a member identifier as well as a message identifier. The member's command prefix appears in messages to identify the source of a message.

---

## Effect of data sharing on sequence number caching

DB2 always assigns sequence numbers in order of request.

Members in a data sharing environment can use the CREATE SEQUENCE statement with the CACHE option to request a "chunk" of sequence numbers. These numbers become the members local cache. Since each member assigns cache from its local chunk, numbers may not be assigned in order across the data sharing group. Each value is guaranteed to be unique.

For example, the following CREATE SEQUENCE statement results in sequence SEQ1 being defined as a block of 20 cache values numbered 1 through 20:

 **GUPI**

```
CREATE SEQUENCE SEQ1 START WITH 1 INCREMENT BY 1 CACHE 20
```

 **GUPI**

Because each data sharing group member gets its own sequenced block of cache values, all values assigned to that member are from that block. If member DB2A

requests the first cache value for SEQ1, it is assigned value 1. If member DB2B requests the next cache value, it is assigned value 21. When member DB2A requests another cache value, it is assigned value 2, and when member DB2B requests another cache value, it is assigned value 22. When a member's block of cache values is exhausted, the next available block of cache values is allocated.

In data sharing systems, if cache value sequence must be assigned in strict numeric order, then the NOCACHE option of the CREATE SEQUENCE statement must be specified.

#### Related reference

 [CREATE SEQUENCE \(DB2 SQL\)](#)

---

## Starting a data sharing member

To start members of a DB2 data sharing group, you must enter a START DB2 command for each member of the group.

If this is the first startup for the group, **you must start the originating member first.**

*Impact of command prefix scope:* If DB2 is installed with a command prefix scope of STARTED (the default and recommended value), you must issue the START DB2 command from the z/OS system on which you want to start DB2 or you must route the command to that z/OS system. For example, to route the START DB2 command to the z/OS system on which you want to start DB1A, issue the following command:

```
ROUTE MVS1,-DB1A START DB2
```

After DB2 is started, you can issue all other commands from any z/OS system in the Parallel Sysplex, and the commands are routed to the appropriate member.

---

## Stopping a data sharing member

Stop individual members of a DB2 data sharing group by using the STOP DB2 command.

This option speeds up shutdown because DB2 bypasses castout and associated cleanup processing in the group buffer pools. Consider specifying CASTOUT(NO) when you stop an individual member of a data sharing group for maintenance.

#### Related reference

 [-STOP DATABASE \(DB2\) \(DB2 Commands\)](#)

---

## States of connections and structures after stopping DB2

When DB2 allocates its coupling facility structures, it specifies a disposition for the structures and for connections to the structures after a normal or abnormal termination.

When you display the structures, you can see different states for the connections and structures based on how the disposition is defined and whether DB2 was stopped normally or shut down abnormally.

## Normal shutdown

After a normal shutdown of DB2, the coupling facility structures specify different states for the connections and structures.

The following table summarizes the information that you see after a normal termination.

Table 16. States of structures and connections after normal DB2 termination

Structure	Connections	State
SCA	None	Allocated
Lock	Failed-persistent <sup>1</sup>	Allocated
Group buffer pools	None with CASTOUT(YES); Failed-persistent with CASTOUT(NO) <sup>2</sup>	Unallocated with CASTOUT(YES); Allocated with CASTOUT(NO)

**Notes:**

1. If a member has no retained locks, its failed-persistent connection to the lock structure is removed when it shuts down. If this is the last member to shut down, the connection remains in a failed-persistent state.
2. If castout failure occurs during shutdown, group buffer pool connections show as failed-persistent, even though DB2 terminates normally.

## Abnormal shutdown

After an abnormal shutdown of DB2, the coupling facility structures specify different states for the connections and structures.

The following table summarizes the information that you see after an abnormal termination.

Table 17. States of structures and connections after abnormal DB2 termination

Structure	Connections	State
SCA	None	Allocated
Lock	Failed-persistent <sup>1</sup>	Allocated
Group buffer pools	Failed-persistent	Allocated

**Note:**

1. If a member has no retained locks, its failed-persistent connection to the lock structure is removed when it shuts down. If this is the last member to shut down, the connection remains in a failed-persistent state.

---

## Submitting work to be processed

The methods that you use to submit work in a non-data sharing environment do not need to change for data sharing. However, you might find it to your advantage to use group attachment names and subgroup attachment names to direct jobs.

## Group attachment names and subgroup attachment names

Utilities and applications that use TSO, batch, DL/I batch, the RRSAP, the CAF, or DB2I (DB2 Interactive) to connect to DB2 have two methods for specifying the member to which they want to connect.

The first method is to specify the name of the subsystem. The second method is to use the group attachment name, which acts as a generic name for all the members of a data sharing group. The group attachment name can be used in place of the name of the DB2 subsystem that runs on the z/OS system from which the job was submitted. If you use a group attachment name, you can also use a subgroup attachment name to provide more organization and control over connection requests. If you specify a subgroup attachment name, the following rules apply:

- A subgroup attachment cannot have the same name as a group attachment and should not have the same name as a DB2 member.
- A subgroup attachment must belong to one, and only one group attachment.
- A member can belong to one subgroup attachment at most and does not need to belong to a subgroup attachment.

If you specify a group attachment name, the utility or application does not need to be sensitive to a particular member, which makes it easier to move jobs around the Parallel Sysplex as needed. The utility or application connects to the first active group member that it finds. However, when the name of a subsystem is the same as the group attachment name, the utility or application attempts to connect to the member first. If that member is not started and group attachment name processing is not disabled, the utility or application attempts to connect to the next group member that is active. (NOGROUP is the group override and is available for CAF and RRSAF.)

You can specify the group attachment name and subgroup attachment name at DB2 installation on the DSNTIPK installation panel. The installation CLIST places the names in the IEFSSNxx member and in the DSNHDECP load module for the group. The group attachment name and subgroup attachment name are included in the output for the command DISPLAY GROUP.

If you do not explicitly specify a subsystem name or group attachment name, DB2 uses DSN as the name of the intended subsystem. As with any application program, make sure you are accessing the set of DB2 libraries with the correct DSNHDECP programming defaults.

#### **Related concepts**

 [Group attachment names and subgroup attachment names \(DB2 Installation and Migration\)](#)

#### **Related tasks**

 [Preparing an application to run on DB2 for z/OS \(DB2 Application programming and SQL\)](#)

## **CICS and IMS applications with DB2 data sharing**

You can specify the group attachment name when running CICS and IMS applications for data sharing.

CICS and IMS applications must be aware of the particular member to which they are attached so they can resolve indoubt units of recovery in the event of a failure. IMS Version 7 or later allows IMS dependent regions to use the DB2 group attachment name to direct jobs to a data sharing group. CICS Transaction Server Version 2.2 or later allows you to use the CICS RESYNCMEMBER=YES option to handle indoubt units of work.

## Related tasks

 Running CICS application programs (DB2 Administration Guide)

## Online utility jobs in data sharing environments

If you specify a group attachment name, you can also specify a subgroup attachment name. When you submit a utility job, you must specify either the group attachment name or the name of the member to which the utility is to attach.

For example, if you specify the group attachment name, the EXEC statement might look like the following statement:

```
//stepname EXEC PGM=DSNUTILB,PARM='group-attach-name,[uid],[utproc]'
```

If you do not use the group attachment name, the utility job must run on the z/OS system where the specified DB2 subsystem is running. Ensure that the utility job runs on the appropriate z/OS system. You must use one of several z/OS installation-specific statements to make sure this happens. These include:

- For JES2 multi-access spool (MAS) systems, insert the following statement into the utility JCL:

```
/*JOBPARM SYSAFF=cccc
```

where *cccc* is the JES2 name. You can specify an asterisk (SYSAFF=\*) to indicate that the job should run on the system from which it was submitted.

- For JES3 systems, insert the following statement into the utility JCL:

```
/*MAIN SYSTEM=(main-name)
```

where *main-name* is the JES3 name.

Your installation might have other mechanisms for controlling where batch jobs run, such as the use of job classes.

## How to stop and restart utilities

In a data sharing environment, you can use the TERM UTILITY command to stop the execution of an active utility only from the member on which the utility is active. You can then use this same command from any active member of the data sharing group to release the resources that are associated with the stopped utility. If a member fails while a utility is executing, you must restart DB2 on either the same or another z/OS system before stopping the utility. For remote site recovery from a disaster at the local site, you can stop utilities that were active at the local site from any restarted member of the group at the remote site.

**Recommendation:** Define all work data sets that are used by utilities on shared disks. Doing so allows you to restart a utility on any member because all members are able to access all required data sets. Use the same utility ID (UID) to restart the utility. UIDs are unique within a data sharing group. If a member fails while a utility is executing, you must restart DB2 on either the same or another z/OS system before restarting the utility.

## How to alter a REORG job

In a data sharing environment, you can use the ALTER UTILITY command to alter the REORG utility only from the member on which the utility is active. This is true for non-data sharing environments as well.

## Related reference

-  JES2 Control Statements (MVS JCL Reference)
-  JES3 Control Statements (MVS JCL Reference)

## Stand-alone utility jobs

DB2 stand-alone utilities (such as DSN1COPY) run as z/OS jobs that have no direct connection to DB2 services. Therefore, a data sharing group has no indication that one of these utilities is running.

In a data sharing environment, if a table space has inter-DB2 read/write interest, its most recently updated pages might be in the coupling facility and a stand-alone utility might not be running with current data. If it is important that the data is in a consistent state, you must stop or quiesce the table space. Also, the data must not be in the recovery pending (RECP) or group buffer pool recovery pending (GRECP) state nor have any logical page list entries. Use the DISPLAY DATABASE command with the RESTRICT option to find out if there are exception statuses for a given table space or index.

---

## Monitoring the group

Monitoring data sharing groups includes tasks such as monitoring the database, and monitoring information about the group, structures, and group buffer pools.

## Obtaining information about the group

You can use the DISPLAY GROUP command to obtain information about a data sharing group.

To obtain general information about all members of a particular group, use the DISPLAY GROUP command, shown here:

### GUPI

```
-DB1A DISPLAY GROUP
```

The command can be issued from any active member of the group, and it displays the output that is shown in the figure below:

```
- DSN7100I -DB1A DSN7GCMD
- *** BEGIN DISPLAY OF GROUP(DSNDB0A)  GROUPELVL(101)      MODE(C)
-                                     GROUP ATTACH NAME(DB0A)
- -----
- DB2
- MEMBER  ID  SUBSYS  CMDPREF  STATUS  LVL  NAME      SUBSYS  IRLMPROC
- -----
- DB1A    1   DB1A    -DB1A    ACTIVE  101  MVS1      DJ1A    DB1AIRLM
- DB2A    2   DB2A    -DB2A    ACTIVE  101  MVS2      DJ2A    DB2AIRLM
- DB3A    3   DB3A    -DB3A    ACTIVE  101  MVS3      DJ3A    DB3AIRLM
- DB4A    4   DB4A    -DB4A    FAILED  101  MVS4      DJ4A    DB4AIRLM
- -----
- SCA  STRUCTURE SIZE:    2560 KB, STATUS= AC,   SCA IN USE:    48 %
- LOCK1 STRUCTURE SIZE:   16384 KB
- NUMBER LOCK ENTRIES:    4194304
- NUMBER LIST ENTRIES:    59770, LIST ENTRIES IN USE:    719
- *** END DISPLAY OF GROUP(DSNDB0A)
- DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 21. Output of DISPLAY GROUP command

## GUPI

The figure above shows the following information:

- The DB2 group name and group release level
- The group attachment name
- The member names and release levels
- The command prefix for each member
- The status of each member (ACTIVE, QUIESCED with or without additional conditions, or FAILED)
- The DB2 release level of each member
- The name of the z/OS system where each member is running (or was last running, in cases where the member is not active)
- The names of the IRLM subsystems to which members are connected. For more information about the IRLM data sharing group, use the z/OS command `F irlmproc,STATUS,ALLI`
- The procedure names of the connected IRLMs
- The SCA structure size and the percentage currently in use
- Lock structure size

The display also shows the following information:

- The maximum number of lock entries possible for the lock table
- The maximum number of modify lock list entries and how many of those list entries are currently in use

**Parallel query information:** To see the COORDINATOR and ASSISTANT subsystem parameters for all active members of the group, use the DETAIL option of DISPLAY GROUP.

### Related concepts

“Avoiding false contention” on page 162

“Enabling parallel processing within a data sharing group” on page 148

## Obtaining information about structures

Use the z/OS command `D XCF,STR` to display information about coupling facility structures and policy information.

### Related information

 MVS System Commands

### Displaying all structures

The z/OS `D XCF,STR` command displays summary information about all coupling facility structures in the policy.

The command has the following syntax:

```
D XCF,STR
```

The command produces the following output:

```

D XCF,STR
IXC359I 15.57.52 DISPLAY XCF
STRNAME      ALLOCATION TIME  STATUS
DSNCAT_GBP0  11/27/2005 15:57:47 DUPLEXING REBUILD NEW STRUCTURE
                                     DUPLEXING REBUILD
                                     REBUILD PHASE: DUPLEX ESTABLISHED

DSNCAT_GBP0  11/27/2005 15:29:43 DUPLEXING REBUILD OLD STRUCTURE
                                     DUPLEXING REBUILD
DSNCAT_GBP1      --      --      NOT ALLOCATED
DSNCAT_GBP11     --      --      NOT ALLOCATED
DSNCAT_GBP16K0   --      --      NOT ALLOCATED
DSNCAT_GBP16K1   --      --      NOT ALLOCATED
DSNCAT_GBP22     --      --      NOT ALLOCATED
DSNCAT_GBP32K    --      --      NOT ALLOCATED
DSNCAT_GBP33     --      --      NOT ALLOCATED
DSNCAT_GBP44     --      --      NOT ALLOCATED
DSNCAT_GBP8K0    --      --      NOT ALLOCATED
DSNCAT_GBP8K1    --      --      NOT ALLOCATED
DSNCAT_LOCK1     11/27/2005 15:26:16 ALLOCATED
DSNCAT_SCA       11/27/2005 15:26:13 ALLOCATED
ISTGENERIC       11/27/2005 15:11:09 ALLOCATED
IXCLINKS         --      --      NOT ALLOCATED
LOCK2            --      --      NOT ALLOCATED
RRSSTRUCT1       --      --      NOT ALLOCATED

```

*Figure 22. Output from D XCF,STR command*

### Displaying information about specific structures

The STRNAME keyword of the z/OS D XCF, STR command displays detailed information about specific structures.

The following command displays information about the duplexed group buffer pool GBP0 for group DSNCAT:

```
D XCF,STR,STRNAME=DSNCAT_GBP0
```

The command produces the following output:

```

D XCF,STR,STRNAME=DSNCAT_GBP0
IXC360I 11.13.38 DISPLAY XCF
STRNAME: DSNCAT_GBP0
STATUS: REASON SPECIFIED WITH REBUILD START:
        OPERATOR INITIATED
        DUPLEXING REBUILD
        REBUILD PHASE: DUPLEX ESTABLISHED
POLICY SIZE      : 32768 K
POLICY INITSIZE : 5000 K
REBUILD PERCENT : N/A
DUPLEX          : ALLOWED
PREFERENCE LIST: LF01      CACHE01
EXCLUSION LIST IS EMPTY

```

#### DUPLEXING REBUILD NEW STRUCTURE

```

-----
ALLOCATION TIME: 04/12/1999 11:13:31
CFNAME        : CACHE01
COUPLING FACILITY: SIMDEV.IBM.EN.ND0200000000
                PARTITION: 0  CPCID: 00
ACTUAL SIZE   : 5120 K
STORAGE INCREMENT SIZE: 256 K
VERSION      : B2162049 D1E56F02
DISPOSITION  : DELETE
ACCESS TIME  : 0
MAX CONNECTIONS: 32
# CONNECTIONS : 2

```

#### DUPLEXING REBUILD OLD STRUCTURE

```

-----
ALLOCATION TIME: 04/12/1999 11:12:51
CFNAME        : LF01
COUPLING FACILITY: SIMDEV.IBM.EN.ND0100000000
                PARTITION: 0  CPCID: 00
ACTUAL SIZE   : 5120 K
STORAGE INCREMENT SIZE: 256 K
VERSION      : B2162023 45B3CB06
ACCESS TIME  : 0
MAX CONNECTIONS: 32
# CONNECTIONS : 2

```

CONNECTION NAME	ID	VERSION	SYSNAME	JOBNAME	ASID	STATE
DB2_V71A	02	00020001	UTEC277	V71ADBM1	002F	ACTIVE NEW,OLD
DB2_V71B	01	00010001	UTEC277	V71BDBM1	0033	ACTIVE NEW,OLD

Figure 23. Output from D XCF,STR,STRNAME command

## Obtaining information about group buffer pools

DB2 provides a DISPLAY GROUPBUFFERPOOL command that is useful for displaying statistical information about group buffer pool use.

However, you can also obtain information about group buffer pools by using the z/OS command D XCF,STR.

Depending on the options you choose for the command, the display output contains the following information:

- A list of all connections to the group buffer pools. For duplexed group buffer pools, only one set of connections exists for both instances of the group buffer pool. For example, if there are three connections to duplexed structure GBP0, there are just three connections, not six connections.

- Statistical reports on group buffer pool use, either by a specific member or by the whole group. Some statistical information is also available for the secondary allocation of a duplexed group buffer pool.

**Related reference**

“Displaying information about specific structures” on page 85

“Group buffer pool monitoring with the DISPLAY GROUPBUFFERPOOL command” on page 206

## Database monitoring options

DB2 attempts to automatically recover logical page list page sets.

For page sets or partitions that have LPL or GRECP status and that are not automatically recovered, either start the page set or partition using the START DATABASE command with SPACENAM and ACCESS (RW) or (RO), or run the RECOVER utility. If any table or index space that is required to confirm START DATABASE command authority is unavailable, INSTALL SYSADM might be required to issue the command.

**Related concepts**

“Recovery of pages on the logical page list” on page 102

**Related reference**

 -START DATABASE (DB2) (DB2 Commands)

## Data sharing status descriptions

The group buffer pool recovery pending (GRECP) and logical page list (LPL) statuses are specific to DB2 data sharing. These statuses can appear on the output from the DISPLAY DATABASE command.

**GUIP**

**GRECP**

“Group buffer pool recovery pending.” The group buffer pool was lost, and the changes that are recorded in the log must be applied to the page set. When a page set is placed in the GRECP state, DB2 sets the starting point for the merge log scan to the log record sequence number (LRSN) of the last complete group buffer pool checkpoint.

DB2 automatically recovers GRECP page sets when the group buffer pool is defined with AUTOREC (YES) and at least one member was connected when the failure occurred.

**LPL** “Logical page list.” Some pages were not read from, or written to, the group buffer pool because of some failure, such as a channel failure between the group buffer pool and the processor. Or perhaps pages could not be read from, or written to, disk because of a transient disk problem.

**GUIP**

## Pages in error

The logical page list contains a list of pages (or a page range) that could not be read or written for some reason. Reasons could include transient disk read and write problems that can be fixed without redefining new disk tracks or volumes.

Specific to data sharing, the LPL also contains pages that could not be read or written for “must-complete” operations, such as a commit or a restart, because of

some problem with the coupling facility. For example, pages can be added if there is a channel failure to the coupling facility or disk, or if locks are held by a failed subsystem, thus disallowing access to the desired page.

The logical page list is kept in the SCA and is thus accessible to all members of the group.

If an application tries to read data from a page that is on the logical page list, it receives a "resource unavailable" SQLCODE. In order to be accessible, pages in the logical page list must first have their logged changes applied to the page set.

To verify the existence of logical page list entries, issue the DISPLAY DATABASE command. The LPL option of DISPLAY DATABASE can then be used to see the specific list of pages:

**GUPI**

```
-DB1A DIS DB(DSNDB01) SPACENAM(*) LIMIT(*) LPL ONLY
```

Output similar to the following is produced:

```
DSNT360I -DB1A
*****
DSNT361I -DB1A * DISPLAY DATABASE SUMMARY
              * GLOBAL LPL
DSNT360I -DB1A
*****
DSNT362I -DB1A  DATABASE = DSNDB01  STATUS = RW
              DBD LENGTH = 8000
DSNT397I -DB1A
NAME      TYPE PART STATUS          LPL PAGES
-----
DBD01     TS      RW,LPL,GRECP  000001,000004,00000C,000010
-----
SYSLGRNX TS      RW,LPL,GRECP  000000-FFFFFF
***** DISPLAY OF DATABASE DSNDB01 ENDED *****
DSN9022I -DB1A DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

**GUPI**

Automatic LPL recovery is attempted when pages are added to the logical page list. Automatic LPL recovery is also performed when you start the table space, index, or partition by using the START DATABASE command with ACCESS(RW) or ACCESS(RO).

**Note:** When a table space or partition is placed in the LPL because undo processing is needed for a NOT LOGGED table space, automatic LPL recovery is not initiated at restart time or during rollback processing. A -START DATABASE command identifying this table space will have no effect on the LPL status.

**Physical read and write errors:**

In some previous versions of DB2, physical read and write errors were recorded in an error page range.

This is still the case; however, if a read or write problem is of undetermined cause, the error is first recorded in the logical page list. If recovery from the logical page list is unsuccessful, the error is then recorded in the error page range.

## Locks that are held during DB2 failure

When a lock is used to allow an object to be changed (called a *modify* lock), the lock is kept in a list in the coupling facility lock structure to allow for recovery in case a member fails.

If a member fails, modify locks become *retained* locks, which means that they are held until the failed subsystem is restarted.

To determine if there are retained locks, use the DISPLAY DATABASE command with the LOCKS option as shown here:

### GUIP

```
-DB1A DISPLAY DATABASE(TESTDB) LOCKS ONLY
```

You can tell if a lock is retained if there is an R in the LOCKINFO field of the report.

NAME	TYPE	PART	STATUS	CONNID	CORRID	LOCKINFO
⋮						
TBS43	TS	01	RW			R-IX,PP
			MEMBER NAME DB2A			

### GUIP

#### Related concepts

“Active and retained locks” on page 120

#### Retained locks:

A normal restart of DB2 resolves and removes retained locks that are held by that member with the full data integrity control that DB2 restart provides.

However, if you cannot restart DB2 and the failed member has retained locks that are severely affecting transactions on other members, consider the following actions:

- Defer the restart processing of the objects that have retained locks.  
When you defer restart processing, the pages that locks are protecting are placed in the logical page list. Those pages are still inaccessible. However, this approach has the advantage of removing any retained page set P-locks, which have the potential of locking out access to an entire page set.
- Cold start the failed member.  
This approach causes DB2 to purge the retained locks, but **data integrity is not protected**. When the locks are released after the cold start, DB2 looks at data whose status is unclear.
- Use the command, `MODIFY irImproc,PURGE,db2name`.  
Like a cold start, this command causes DB2 to purge the retained locks and with this method, **data integrity is not protected**.
- Restart the failed member in light mode (restart light).  
Restart light is not recommended for a restart in place. It is intended for a cross-system restart in the event of a failed z/OS to quickly recover retained locks. Restart light enables DB2 to restart with a minimal storage footprint and then terminate normally after the locks are released.

### Related concepts

“Restart light” on page 122

“Restarting a member with conditions” on page 132

### Related tasks

 Deferring restart processing (DB2 Administration Guide)

### Related reference

 MODIFY irlmproc,PURGE (z/OS IRLM) (DB2 Commands)

## Determining the data sharing member on which SQL statements run

Use the special register CURRENT MEMBER to determine the member of a data sharing group on which SQL statements execute.

The data type of CURRENT MEMBER is CHAR(8). If necessary, the member name is padded on the right with blanks so that its length is 8 bytes. The value of CURRENT MEMBER is a string of blanks when the application process is connected to a DB2 subsystem that is *not* a member of a data sharing group.

**Example:** To set the host variable MEM to the name of the current member, use one of the following statements:

#### GUPI

```
EXEC SQL SET :MEM = CURRENT MEMBER;  
EXEC VALUES (CURRENT MEMBER) INTO :MEM
```

#### GUPI

---

## Controlling connections to remote systems in a data sharing environment

Controlling DDF connections in a data sharing environment is somewhat different from controlling DDF connections in a non-data sharing environment.

### Starting and stopping DDF

You control the distributed data facility (DDF) on a member basis, not on a group-wide basis.

#### GUPI

This gives you more granular control over DDF processing. For example, assume that you want to devote member DB1A to batch processing for some period of time without disrupting other connections. You can enter the following command to disallow any further distributed connections from coming into this member:

```
-DB1A STOP DDF MODE(QUIESCE)
```

To stop **all** DDF processing for the group, you need to issue the STOP DDF command for every member of the group. For example, you might need to do this when you change the SYSIBM.LOCATIONS table.

To allow for completion of CREATE, ALTER, DROP, GRANT, or REVOKE operations from remote requesters, issue the STOP DDF MODE(SUSPEND) command. 

## Monitoring connections to remote systems

The DISPLAY THREAD command can display information about all threads in the data sharing group. The DISPLAY LOCATION command shows thread information only for the member on which it is issued.

If a data sharing group is defined to have a generic LU, you must use a member's real LU name for *luwid* if you are requesting information by *luwid* (logical unit of work ID).

If the DISPLAY THREAD is being related to global transactions coordinated by *xid*, DISPLAY THREAD shows *xid* information. The thread is managed by an XA transaction manager, such as Websphere, which identifies the transaction with an *xid*. The *xid* is provided to allow correlation with the XA transaction manager. DB2 uses both the logical unit of work identifier, *luwid*, and the XA transaction identifier, *xid*, to coordinate and recover transactions.

When a remote DB2 subsystem issues a DISPLAY LOCATION command to obtain information about connections to a data sharing group, the output displays information about every LU at that location.

### Related reference

 -DISPLAY LOCATION (DB2) (DB2 Commands)

 -DISPLAY THREAD (DB2) (DB2 Commands)

## Resetting generic LU information

If you use a generic LU name to connect to a member of a data sharing group that is using two-phase commit, VTAM permanently records information in the coupling facility about which member of the group was involved in the communication.

This permanently recorded information is required to guarantee that future VTAM sessions are always directed to the same member, making it possible to provide access to the correct member log for resolution of indoubt threads.

At times, you might need to break that affinity between the member and the other system. You would need to do this, for example, if you want to use member-specific access, or if you want to remove a member from the data sharing group.

To break this affinity, issue the RESET GENERICLU command from an active member of the data sharing group. You must issue this command from the member with the VTAM affinity to the particular LU. Here is an example of a command that removes information about USIBMSTODB22 from DB1A:



```
-DB1A RESET GENERICLU(LUDB22)
```



Take great care when using this command because it can potentially cause the specified partner LUs to connect to different members on later sessions. This can cause operational problems if indoubt threads exist at a partner LU when this command is issued.

**Related reference**

 `-RESET GENERICLU (DB2) (DB2 Commands)`

## Logging environment for data sharing

In a data sharing environment, the member subsystems still maintain separate recovery logs. Each manages its own active and archive log data sets and records those in its own bootstrap data set (BSDS).

The shared communications area (SCA) in the coupling facility contains information about all members' BSDSs and log data sets. In addition, every member's BSDS contains information about other members' BSDS and log data sets, in case the SCA is not available.

The following figure illustrates a typical logging environment.

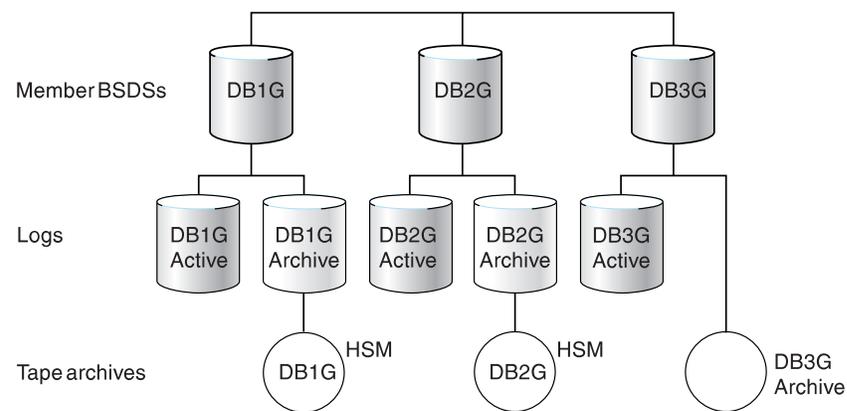


Figure 24. Member BSDSs and logs in the data sharing environment

## The impact of archiving logs in a data sharing group

In data sharing, the DB2 RECOVER utility needs log records from every member that has changed the object that needs to be recovered.

If the logs are archived, the impact on RECOVER depends on how the log data sets are archived:

- Archive to disk without DFSMSHsm to migrate the data sets from disk to tape.  
No major impact on performance is experienced for this type of archive. But you need enough disk space to hold archive logs, and the disk devices must be shared (accessible) by all members in a data sharing group. Because DFSMSHsm or its equivalent is not used, you must manage disk space carefully to avoid running out of space.
- Archive to disk with DFSMSHsm.  
DFSMSHsm can do automatic space and data availability management among storage devices in a system. DFSMSHsm can migrate the archive on disk to less expensive storage (such as tape), and recall the archive back to disk when needed.

Using DFSMSHsm, a particular RECOVER job needs only one tape unit to recall migrated archive data sets. If the archive data sets have been migrated, recovery time might be adversely affected, because the recalls of the migrated archive data sets are done one at a time from the member running the RECOVER job. For example, a RECOVER job started on DB1A might need log data sets from DB1A, DB2A, and DB3A. DB1A sends the recall requests to DFSMSHsm one at a time for the tapes needed for recovery.

- Archive to tape.

The RECOVER job needs at least one tape unit for each member whose archived log records are to be merged. (More might be needed if you run more than one recover at the same time for different partitions of a partitioned table space.) Therefore, **do not archive logs from more than one system to the same tape.**

**Recommendation:** For data sharing, avoid using tape archive logs for data recovery.

If you must archive to tape, ensure that the value for READ TAPE UNITS on installation panel DSNTIPA for each member is high enough to handle anticipated recovery work. For example, if you have eight members, each member should specify at least eight tape drives. You need more if you run more than one recovery job at the same time on a given member, or if multiple members run recovery jobs at the same time.

If there are not enough tape units to do the recovery, DB2 can possibly deadlock. If this happens, use the command SET ARCHIVE to increase the number of tape units that can be used.

Also, ensure that you specify 0 for the DEALLOC PERIOD parameter on installation DSNTIPA to avoid making an archive tape inaccessible to other members of the data sharing group. (If you intend to run all RECOVER jobs from a single member, this suggestion does not apply.)

#### Related concepts

“How recovery works in a data sharing group” on page 94

## How to avoid using the archive log

A recovery cycle for a table space is defined by how often its image copy is taken.

A RECOVER job needs the following copies and records:

- The latest image copy
- The optional incremental copies
- The log records since the last incremental or image copy
- Archived log records, if all of the log records since the last incremental image copy are **not** still in active log data sets

Keeping all of the log records since the last incremental image copy in the active log data sets is to your advantage, because reading log records from the active log is much faster than reading from archive logs, even if those archives are on disk.

Several ways exist to minimize the need to use the archive log:

- Increase the total active log space.

The total amount of active log space is the number of active log data sets multiplied by its size. Currently, DB2 limits the maximum number of active log data sets to 93. Because each member can have up to 93 active log data sets, the total number of active log data sets is effectively increased by the number of members in a data sharing group.

The size of an active log data set is up to 4 GB but is usually limited by the size of a tape cartridge. Most installations prefer not to have an archived data set on more than one tape volume.

**Recommendation:** Do not use tape compression for the DB2 archive log, because DB2 needs to read the log backwards for backout operations. Performance for backout can be severely degraded if tape compression is used.

Tape compression is not DB2 data compression, which compresses the data portion of a DB2 log record. With DB2 data compression, the log record header is not compressed and causes no extra performance degradation for backward scans.

- Increase the frequency of incremental image copies.

Because only the log records generated since the last incremental image copy are needed for recovery, the more often you make incremental image copies, the less chance there is that archive log records will be needed. Weigh this consideration against the time it takes to make the incremental image copies and the effects on SQL transactions.

- Make sure applications commit frequently.

To avoid having to mount an archive log for backing out changes, ensure that applications are committing frequently. Consider using the UR CHECK FREQUENCY parameter or the UR LOG WRITE CHECK parameter of installation panel DSNTIPL to help you track when applications are not committing as frequently as your site guidelines suggest.

#### **Related tasks**

“Improving recovery performance” on page 96

---

## **Recovering data**

The procedures for data recovery are fundamentally the same for data sharing as for non-data-sharing environments. Data sharing involves one catalog, but many logs and BSDSs. In addition to disk and cache controllers, the coupling facility adds a possible point of failure and requires appropriate procedures for data recovery.

In planning for data sharing, consider having more than one coupling facility. If an SCA or lock structure failure occurs, recovery for that structure can proceed automatically if a second coupling facility is available.

### **How recovery works in a data sharing group**

In case you need to recover data, it can be helpful to understand how data recovery works in a data sharing environment.

#### **Logs that are needed for recovery**

Assume that three members of a data sharing group are making updates to table space TS1, as shown in the figure below.

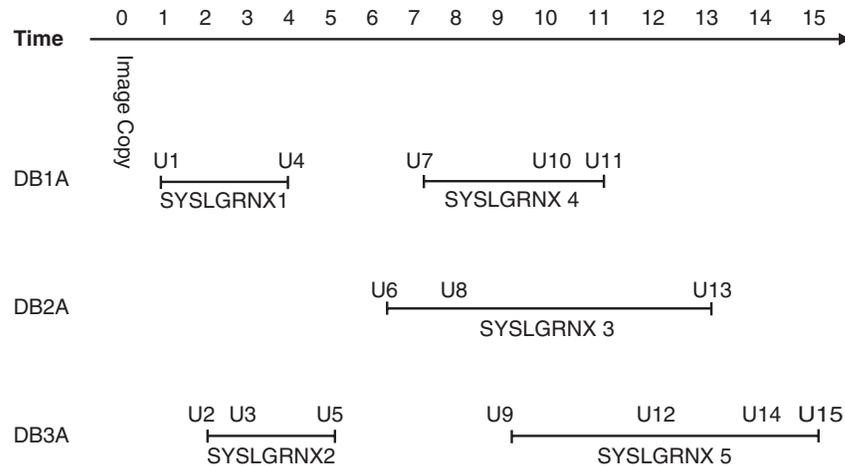


Figure 25. Three members of a data sharing group updating table space TS1

The following is the sequence of overlapping updates leading up to the time of recovery:

1. DB1A updates TS1 between Time 1 and 4 (SYSLGRNX record 1) with two updates (U1 and U4).
2. DB3A updates TS1 between Time 2 and 5 (SYSLGRNX record 2) with three updates (U2, U3, and U5).
3. DB2A updates TS1 between Time 6 and 13 (SYSLGRNX record 3) with three updates (U6, U8, and U13).
4. DB1A updates TS1 again between Time 7 and 11 (SYSLGRNX record 4) with three updates (U7, U10, and U11).
5. DB3A updates TS1 again between Time 9 and 15 (SYSLGRNX record 5) with four updates (U9, U12, U14 and U15).

Now, assume that you want to recover TS1 to time 9. The full image copy taken at T0 is used as the recovery base. All the SYSLGRNX records mentioned previously are selected to determine the log ranges of each system for the log scan. Updates U1 through U9 are applied in order.

## How log records are applied

DB2 can access the logs of other DB2 subsystems in the group and merge them in sequence. The log record sequence number (LRSN) identifies the log records of a data sharing member. Before Version 10 new-function mode, the LRSN is always incremented for log records that pertain to the same page. Starting with Version 10 new-function mode, multiple row insert might produce duplicate LRSNs for changes to the same page. The following figure illustrates the structure of the log record.

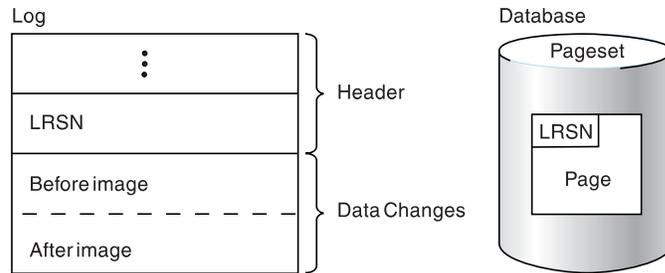


Figure 26. The log and LRSN in the data sharing environment. During recovery, DB2 compares the LRSN in the log record with the LRSN in the data page to determine whether the log record must be applied to the data on disk.

The log record header contains an LRSN. The LRSN is a 6-byte value that is greater than or equal to the timestamp value truncated to 6 bytes. This value also appears in the page header. During recovery, DB2 compares the LRSN in the log record to the LRSN in the page header before applying changes to disk. If the LRSN in the log record is larger than the LRSN on the data page, the change is applied.

## Improving recovery performance

You can improve your recovery performance by taking more frequent image copies.

You might want to limit this activity by determining which table spaces most need fast recovery. The following guidelines are provided as a starting point to help you determine how often you must do incremental image copies. As with a single subsystem, doing frequent image copies can help you avoid using the archive log for recovery.

To improve recovery performance:

Use the following guidelines to increase the frequency of image copies for each member of the data sharing group. Use the output of the DSNJU004 (print log map) utility for each member.

1. Find the starting timestamp of the active log data set with the lowest STARTRBA.
2. Find the ending timestamp of the active log data set with the highest ENDRBA.
3. Calculate the time interval:  

$$\text{time\_interval} = \text{end\_TIMESTAMP} - \text{start\_TIMESTAMP}$$
4. Calculate the interval at which to perform incremental image copies:  

$$\text{interval of copy} = \text{time\_interval} * (n-1) / n$$

*n* is the number of active log data sets.
5. Take the smallest interval for the group and, to account for fluctuations of log activity, decrease the interval by 30%. (30% is an arbitrary figure; you might have to adjust this interval based on your system's workload.)

This is the recommended interval for doing incremental image copies. If the interval is too small to be realistically accomplished, consider increasing the size or number of active log data sets.

Periodically run the MERGECOPY utility with incremental image copies. The RECOVER utility attempts to mount tape drives for all the incremental image

copies at the same time. If it runs out of tape drives, it switches to log apply. MERGECOPY merges what it can and then mounts more incremental image copies.

#### Related reference

 COPY (DB2 Utilities)

## Recovery options for data sharing environments

Use the RECOVER utility to recover to currency or to a prior point in time.

### Recovery to currency

This process is used to recover from damaged data by restoring from a backup and applying all logs to the current time. The recovery process operates similarly in the data sharing and non-data-sharing environments. Image copies are restored and subsequently updated based on changes recorded in the logs. In the data sharing group, multiple member logs are read concurrently in log record sequence.

### Point-in-time recovery

This process discards potentially corrupt data by restoring a database to a prior point of consistency. Corrupt data might result from a logical error. The following point-in-time recovery options are available:

#### TORBA

This option is used to recover to a point on the log defined by a receive byte address (RBA). In a data sharing environment, TORBA can only be used to recover to a point prior to defining the data sharing group.

#### TOLOGPOINT

This option is used to recover to a point on the log defined by a log record sequence number (LRSN). The TOLOGPOINT keyword must be used when you recover to a point on the log after the data sharing group was defined. However, you can also use TOLOGPOINT in a non-data-sharing environment.

The LRSN is a 6-byte hexadecimal number derived from a store clock timestamp. LRSNs are reported by the DSN1LOGP stand-alone utility.

#### TOCOPY

This option is used to recover data or indexes to the values contained in an image copy without subsequent application of log changes.

Successful recovery clears recovery pending conditions and brings data to a point of consistency. In a data sharing environment, all pages associated with the recovered data entity are removed from the group buffer pool and written to disk.

### Recovery using a system-level backup

The RECOVER utility can recover an object or a list of objects using a system-level backup, that is created with the BACKUP SYSTEM utility, as a recovery base.

### Related reference

[📄 RECOVER \(DB2 Utilities\)](#)

## System-level point-in-time recovery

Use the `BACKUP SYSTEM` and `RESTORE SYSTEM` online utilities to perform system-level point-in-time recovery.

DB2 provides the following solutions for system-level recovery:

- Recovery to a specific point in time  
This point in time is either between two backup times or between the last backup time and the current time. After the appropriate volume copies are restored, the outstanding logs are applied to the databases to recover the data to the designated point in time.
- Recovery to the point in time of a backup  
This recovery restores the appropriate volume copies of the data and logs. The logs are used only to back out inflight transactions on restart.
- Remote site recovery from disaster at a local site  
This recovery is determined by what you keep at the remote site. For example, in addition to offsite tapes of backups, current logs might have been transmitted electronically. If so, the current logs can be applied to the databases after the data and logs are restored from the offsite tapes.

### BACKUP SYSTEM online utility

The `BACKUP SYSTEM` online utility provides two types of system copies.

- A data-only system backup, which contains only the databases  
The `RESTORE SYSTEM` online utility uses these backups to recover the system to an arbitrary point in time.
- A full system backup, which contains both logs and databases  
The `RESTORE SYSTEM` online utility uses these backups to recover the system to the point in time when the copy was taken by using normal DB2 restart recovery.

### RESTORE SYSTEM online utility

The `RESTORE SYSTEM` utility recovers a DB2 subsystem to a prior point in time. The utility restores volume copies that are provided by the `BACKUP SYSTEM` online utility with the `DATA ONLY` option.

After restoring the data, you can use the `RESTORE SYSTEM` online utility to recover to an arbitrary point in time.

### Related concepts

[📄 Point-in-time recovery with system-level backups \(DB2 Administration Guide\)](#)

### Related reference

[📄 BACKUP SYSTEM \(DB2 Utilities\)](#)

[📄 RESTORE SYSTEM \(DB2 Utilities\)](#)

## Recovering a data sharing group in case of a disaster

To develop a procedure for recovering a data sharing group by using a remote site, you can use the disaster recovery procedure as a base. In most cases, you must complete those steps for each member of the data sharing group.

As an alternative, you can set up the remote data sharing group as a tracker site. The advantage of a tracker site is that it dramatically reduces the amount of time that is needed for takeover if a disaster occurs at the primary site. The disadvantage is that the tracker site must be dedicated to shadowing the primary site. You cannot use the tracker site to perform transactions of its own.

### Related concepts

“Recovery procedure differences” on page 101

### Related tasks

-  Creating essential disaster recovery elements (DB2 Administration Guide)
-  Recovering from disasters by using a tracker site (DB2 Administration Guide)

## Configuration of the recovery site

The hardware configuration can be different at the recovery site, as long as it supports data sharing.

The recovery site must have a data sharing group that is identical to the group at the local site. It must have the same name, the same number of members, and the members must have the same names as those at the local site. The CFRM policies at the recovery site must define the coupling facility structures with the same names as those at the local site (although the sizes can be different).

You can run the data sharing group on as few or as many z/OS systems as you want.

Conceptually, there are two ways to run the data sharing group at the recovery site. Each way has different advantages that can influence your choice:

- Run a multi-system data sharing group.  
The local site is most likely configured this way. You have a Parallel Sysplex containing many CPCs, z/OS systems, and DB2 subsystems. This configuration requires a coupling facility, the requisite coupling facility channels, and the Sysplex Timer.  
Using a multi-system data sharing group at the recovery site, you have the same availability and growth options that you have at the local site.
- Run a single-system data sharing group.  
In this configuration, you centralize all of your DB2 processing within a single, large processor, such as an IBM System z9<sup>®</sup> or System z10<sup>®</sup>, and eServer<sup>™</sup> zSeries<sup>®</sup> 900 or 990. As the following figure shows, you must *install* a multi-member data sharing group. After the group starts up, you shut down all but one of the members and access data through the remaining member.

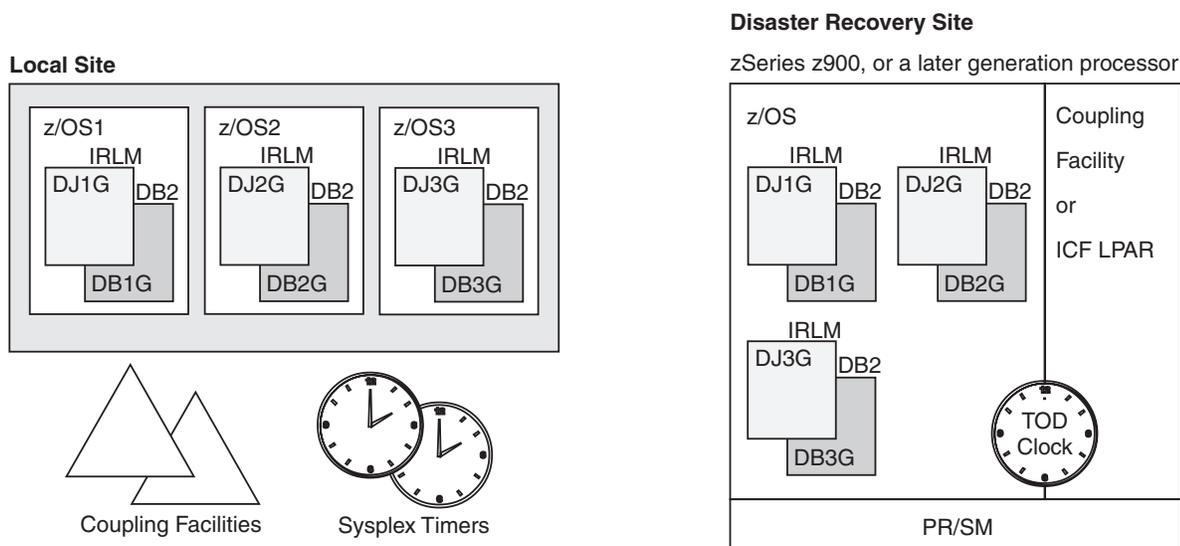


Figure 27. Example of local and disaster recovery site configurations. With this configuration, the recovery site can be a single-system data sharing group. After the data sharing group is started on the recovery site, you can stop all but one of the members.

With a single-system data sharing group, you lose the availability benefits of the Parallel Sysplex, but the group has fewer hardware requirements:

- The Sysplex Timer is not needed; you can use the CPC's time-of-day clock.
- You can use any available coupling facility configuration for the recovery site system, including Integrated Coupling Facilities (ICFs).

With a single-system data sharing group, there is no longer inter-DB2 read/write interest, and the requirements for the coupling facility are as follows:

- A lock structure (which can be smaller)
- An SCA

Group buffer pools are not needed for running a single-system data sharing group. However, you do need to have at least small group buffer pools for the initial startup of the group. DB2 allocates them and uses them to do its damage assessment processing. When you are ready to do single-system data sharing, you can remove the group buffer pools by stopping all members and then restarting the member that is handling the workload at the disaster recovery site.

#### Related information

[IBM eServer zSeries Processor Resource/Systems Manager Planning Guide](#)

### What to send to the recovery site

You must send to the recovery site the same information as for single-system remote recovery: logs and BSDSs, image copies, and so on.

To prepare the logs for the remote site, you have three options:

- Use the command ARCHIVE LOG with the MODE (QUIESCE) option to ensure a point of consistency for each of the log data sets. If the quiesce is not successful, the command fails and the logs are not archived.

At the recovery site (just as for non-data-sharing disaster recovery), the ENDRBA value you use for restarting each member is the end RBA +1 of the latest archive log data set from each member in the data sharing group.

- Use the command ARCHIVE LOG SCOPE(GROUP). This version of the command does not ensure a point of consistency for all members' logs, but the logs are archived on each of the active members of the data sharing group. You can use the ENDLRSN option of the DSNJU003 (change log inventory) utility on the remote site to truncate all logs to the same point in time.

To determine the truncation value, look at the print log map output from the latest copies of the archived BSDS.

Another way to determine the truncation value is to ship the SYSLOG containing message DSNJ003I with your archive log data sets to the recovery site. This message is issued when archive log data sets are created (when you issue the ARCHIVE LOG command). The message contains the starting and ending LRSN and RBA values for the archive log data set. For example, the following messages appear when the command ARCHIVE LOG SCOPE(GROUP) is issued from one of the members at the local site:

```
DSNJ003I -DB1A DSNJ0FF3 FULL ARCHIVE LOG VOLUME
DSNAME=DSNC510.ARCHLOG1.A0000003, STARTRBA=000001C68000,
ENDRBA=000001D4FFFF, STARTLRSN=ADFA208AA36C, ENDLRSN=AE3C45273A77,
UNIT=SYSDA, COPY1VOL=SCR03, VOLSPAN=00, CATLG=YES

DSNJ003I -DB2A DSNJ0FF3 FULL ARCHIVE LOG VOLUME
DSNAME=DSNC518.ARCHLOG1.A0000001, STARTRBA=000000000000,
ENDRBA=0000000D6FFF, STARTLRSN=ADFA00BB70FB, ENDLRSN=AE3C45276DD7,
UNIT=SYSDA, COPY1VOL=SCR03, VOLSPAN=00, CATLG=YES
```

Compare the ending LRSN values for all members' archive logs, and choose the lowest LRSN as the truncation point. For the two members shown previously, the lowest LRSN is AE3C45273A77. To get the last complete log record, you must subtract 1 from that value, so you would enter AE3C45273A76 as the ENDLRSN value in the CRESTART statement of the DSNJU003 utility for each of the members at the remote site. All log records with a higher LRSN value are discarded during the conditional restart.

- Use the command SET LOG SUSPEND if you are using the IBM RAMAC Virtual Array (RVA) storage control with the peer-to-peer remote copy (PPRC) function or Enterprise Storage Server® Flashcopy to create point-in-time backups of entire DB2 subsystems for faster recovery at a remote site. Using either of these methods to create a remote copy requires the suspension of logging activity, which prevents database updates. The SUSPEND option of the SET LOG command suspends logging and update activity until a subsequent SET LOG command with the RESUME option is issued.

**Important:** Make sure that all members of the group are active when you archive the logs. If you have a quiesced member whose logs are necessary for a recovery base at the disaster recovery site, you must start that member with ACCESS(MAINT) to archive its log.

For read-only members, DB2 periodically writes a log record to prevent those members from keeping the LRSN value too far back on the log.

#### Related tasks

 Performing remote site recovery from a disaster at a local site (DB2 Administration Guide)

#### Recovery procedure differences

The procedure for data sharing at the recovery site differs in that extra steps exist for cleaning out old information in the coupling facility.

Old information exists in the coupling facility from any practice startups. In addition, you must prepare each subsystem (rather than just a single system) for a conditional restart.

#### Related tasks

 [Performing conditional restart \(DB2 Administration Guide\)](#)

## Recovery of pages on the logical page list

DB2 attempts to automatically recover pages when they are added to the logical page list. Recovered pages are deleted from the logical page list when recovery processing successfully completes.

In some situations—such as a channel failure between the group buffer pool and the processor, a transient disk problem, or DB2 in restart—the automatic LPL recovery processor cannot be initiated, or the recovery cannot complete. In these situations, the pages are kept in the logical page list. The following tasks show several ways to perform manual LPL recovery:

- Start the object with access (RW) or (RO). This command is valid even if the table space is already started.  
When you issue the command `START DATABASE`, you see message `DSNI006I`, indicating that LPL recovery has begun. Message `DSNI022I` might be issued periodically to give you the progress of the recovery. When recovery is complete, you see message `DSNI021I`.
- Run the `RECOVER` utility on the object.  
The only exception to this is when a logical partition of a type 2 nonpartitioned index has both LPL and `RECP` status. If you want to recover the logical partition using `RECOVER INDEX` with the `PART` keyword, you must first use the command `START DATABASE` to clear the logical page list pages.
- Run the `LOAD` utility with the `REPLACE` option on the object.
- Issue an SQL `DROP` statement for the object.
- Use the utility `REPAIR SET` with `NORCVRPEND`. This can leave your data in an inconsistent state.
- Use `START DATABASE ACCESS(FORCE)`. This can leave your data in an inconsistent state.

None of the items in the preceding list work if retained locks are held on the object. You must restart any failed member that is holding those locks.

When a table space or partition is placed in the LPL because undo processing is needed for a `NOT LOGGED` table space, either at restart time or during rollback processing, automatic LPL recovery is not initiated. A `-START DATABASE` command identifying this table space will have no effect on the LPL status.

## Recovery from coupling facility failures

Failures of the coupling facility can be classified into two main groups: structure failures and connectivity failures.

- Structure failure  
A structure failure is a rare event in which structures are damaged in some way but the coupling facility continues to operate.
- Connectivity failures

Connectivity failures can be caused by a problem with the attachment of the z/OS system to the coupling facility. They can also occur when the following types of failures occur:

- Power failure that affects the coupling facility but leaves one or more z/OS systems running
- Deactivation of the coupling facility partition
- Failure of the coupling facility control code
- Failure of the coupling facility CPC or LPAR

### **Preparation for structure and connectivity failures**

If properly configured, DB2 and IRLM can recover very quickly and with very little disruption from any kind of coupling facility failure. If coupling facilities are not properly configured, coupling facility failures can cause serious outages for users.

Not having a lock structure or SCA causes the entire data sharing group to come down abnormally. Group buffer pool failure can mean loss of availability for applications depending on the data in that group buffer pool.

Careful preparation can greatly reduce the impact of coupling facility outages to your users. To best prepare yourself for structure and connectivity failures, you must have the following:

- Group buffer pool duplexing enabled.
- Alternative coupling facility information provided on the preference list of each of the structures in the CFRM policy.
- For simplex coupling facility structures:

- An active system failure management (SFM) policy with system weights specified.

This is strongly recommended for simplex coupling facility structures. Descriptions of failure scenarios in this section assume that you have done this. If you have not, it is not possible to automatically rebuild simplex coupling facility structures. If the SCA and lock structure cannot be rebuilt, DB2 abnormally terminates the members affected by the loss of those structures, or the loss of connectivity to those structures. If the group buffer pool cannot be rebuilt, which is only attempted when a subset of members lose connectivity, those members disconnect from the group buffer pool.

- A REBUILDPERCENT value specified in the CFRM policy for all DB2-related structures

In general, specifying a low REBUILDPERCENT value is recommended to allow for automatic rebuild when a member loses connectivity.

- Adequate storage in an alternate coupling facility to rebuild or reallocate structures as needed.

For rebuild, z/OS uses the current size structure of the CFRM policy on the alternate coupling facility to allocate storage. If z/OS cannot allocate enough storage to rebuild the SCA or lock structure, the rebuild fails. If it cannot allocate enough storage for the group buffer pool, DB2 must write the changed pages to disk instead of rebuilding them into the alternate group buffer pool.

## Related concepts

 [Coupling facility availability \(DB2 Installation and Migration\)](#)

 [Allocating a Structure in a Coupling Facility \(z/OS MVS Programming: Sysplex Services Guide\)](#)

## Failure scenarios

Coupling facility failure scenarios include connectivity failures and structure failures.

### Connectivity failure to the SCA or lock structure

The following table summarizes what happens when there are connectivity failures to the SCA.

Table 18. Summary of connectivity failures to the SCA

Active SFM policy?	Weighted loss < REBUILDPERCENT?	DB2 response	Operational response
No	Not applicable	Each affected member issues: DSN7501A 00F70600  DB2 comes down. Connection is deleted; structure remains allocated.	Options include: <ul style="list-style-type: none"><li>• Fix problem</li><li>• Restart failed member on system that is connected to coupling facility</li><li>• Manually rebuild onto another coupling facility.</li></ul>
Yes	Yes	Each affected member issues: DSN7501A 00F70600  DB2 comes down. Connection is deleted; structure remains allocated.	Options include: <ul style="list-style-type: none"><li>• Fix problem</li><li>• Restart failed member on system that is connected to coupling facility</li><li>• Manually rebuild onto another coupling facility.</li></ul>
Yes	No	Automatic rebuild. DSN7503I	None needed.

The following table summarizes what happens when there are connectivity failures to the lock structure.

Table 19. Summary of connectivity failures to the lock structure

Active SFM policy?	Weighted loss < REBUILDPERCENT?	DB2 response	Operational response
No	Not applicable	Each affected member issues: DXR136I 00E30105  DB2 comes down. Connection is failed-persistent; structure remains allocated.	Options include: <ul style="list-style-type: none"><li>• Fix problem</li><li>• Restart failed member on system that is connected to coupling facility</li><li>• Manually rebuild onto another coupling facility.</li></ul>

Table 19. Summary of connectivity failures to the lock structure (continued)

Active SFM policy?	Weighted loss < REBUILDPERCENT?	DB2 response	Operational response
Yes	Yes	Each affected member issues: DXR136I 00E30105  DB2 comes down. Connection is failed-persistent; structure remains allocated.	Options include: <ul style="list-style-type: none"> <li>• Fix problem</li> <li>• Restart failed member on system that is connected to coupling facility</li> <li>• Manually rebuild onto another coupling facility.</li> </ul>
Yes	No	DXR143I  Automatic rebuild. DXR146I	None needed.

### Connectivity failure to non-duplexed group buffer pools

The following table summarizes what happens when there are connectivity failures to the group buffer pools.

Table 20. Summary of connectivity failures for non-duplexed group buffer pools

Connectivity lost from	Weighted loss < REBUILDPERCENT?	DB2 response	Operational response
100% of members connected to a group buffer pool that is defined with GBPCACHE(YES)	No	Each affected member issues: DSNB303E DSNB228I  Add pages to logical page list, if necessary. DSNB250E DSNB314I* rsn=100% DSNB304I*  Damage assessment, GRECP page sets. DSNB320I DSNB321I DSNB353I DSNI006 DSNI021 DSNB354I DSNB305I*	None needed if the group buffer pool is defined with AUTOREC(YES), and DB2 successfully recovers the page sets. Otherwise, enter START DATABASE commands.
100% of members connected to a group buffer pool that is defined with GBPCACHE(NO)	No	Automatic rebuild. DSNB331I DSNB338I	None needed.

Table 20. Summary of connectivity failures for non-duplexed group buffer pools (continued)

Connectivity lost from	Weighted loss < REBUILDPERCENT?	DB2 response	Operational response
A subset of members connected to some or all group buffer pools.	Yes	<p>Each affected member issues:</p> <p>DSNB303E            DSNB228I            DSNB313I rsn=LOSSCONN</p> <p>Quiesce applications that use the group buffer pool. Add pages to logical page list if necessary.</p> <p>DSNB250E            DSNB311I, DSNB312I</p> <p>Disconnect GBPx failed-persistent.            DSNB309I</p>	<p>Options include:</p> <ul style="list-style-type: none"> <li>• Fix the problem.</li> <li>• Manually rebuild the structure onto another coupling facility.</li> <li>• Stop and restart DB2 on a system that is connected to the coupling facility.</li> </ul> <p>Enter START DATABASE commands to recover logical page list entries.</p>
A subset of members connected to some or all group buffer pools.	No	<p>Automatic rebuild.</p> <p>DSNB331I            DSNB332I            DSNB333I <sup>1</sup>            DSNB338I</p>	None needed.

**Note:**

1. Issued by the structure owner.

### Structure failures

The following table summarizes what happens to each structure if there is a structure failure. This information is for non-duplexed group buffer pools.

Table 21. Summary of structure failures, by structure type

Failed structure	DB2 response	Operational response
SCA	<p>DSN7502I</p> <p>Automatic rebuild            DSN7503I</p>	None needed.
Lock structure	<p>DXR143I</p> <p>Automatic rebuild            DXR146I</p>	None needed.

Table 21. Summary of structure failures, by structure type (continued)

Failed structure	DB2 response	Operational response
Group buffer pool that is defined with GBPCACHE(YES)	DSNB228I DSNB314I rsn=STRFAIL  Add pages to logical page list, if necessary. DSNB250E  Damage assessment, GRECP page sets. DSNB304I DSNB320I DSNB321I DSNI006 DSNI021 DSNB305I	None needed if the group buffer pool is defined with AUTOREC(YES) and DB2 successfully recovers the page set. Otherwise, enter START DATABASE commands.
Group buffer pool that is defined with GBPCACHE NO	Automatic rebuild. DSNB331I DSNB338I	None needed.

### Connectivity failures for the duplexed group buffer pools

For duplexed group buffer pools, a failure response is the same for both a loss of structure and for lost connectivity, as shown in the following table.

Table 22. Summary of scenarios for both a loss of structure and lost connectivity for duplexed group buffer pools

Structure loss	DB2 response	Operational response
Primary	Switch to secondary in simplex mode DSNB744I DSNB745I  If DUPLEX(ENABLED) then reduplexing is attempted.	If the system did not automatically reduplex, correct the problem with the failed coupling facility. If you want to restart duplexing, use the z/OS SETXCF command.
Secondary	Revert to primary in simplex mode DSNB743I DSNB745I  If DUPLEX(ENABLED) then reduplexing is attempted.	If the system did not automatically reduplex, correct the problem. If you want to restart duplexing, use the SETXCF command.
Both (loss of structure or 100 % LOSSCONN)	Damage assessment, GRECP page sets.	None needed if the group buffer pool is defined with AUTOREC(YES) and DB2 successfully recovers the page set. Otherwise, enter START DATABASE commands.

### Related concepts

“Coupling facility recovery scenarios”

“Problem: Loss of SCA structure” on page 111

“Problem: Loss of lock structure” on page 110

“Problem: Loss of group buffer pool structure (non-duplexed)”

## Coupling facility recovery scenarios

Looking at a few problem scenarios might help if you need to recover a coupling facility.

### Problem: loss of coupling facility (CF)

You might encounter a failure that is treated by z/OS and DB2 as a total loss of connectivity to the coupling facility and coupling facility structures.

In these example scenarios, an active SFM policy is assumed.

### Problem: Loss of group buffer pool structure (non-duplexed):

In your data sharing environment, you might encounter an error that is caused by the loss of a non-duplexed group buffer pool structure.

### Symptom

Some or all of the following messages appear, depending on which structures DB2 tries to access:

```
DSNB303E -DB1A csect-name A LOSS OF CONNECTIVITY
WAS DETECTED TO GROUP BUFFER POOL gbpname.
DSNB228I csect-name GROUP BUFFER POOL gbpname
CANNOT BE ACCESSED FOR function
MVS IXLCACHE REASON CODE=reason
DSNB314I csect-name DAMAGE ASSESSMENT TO BE TRIGGERED FOR
GROUP BUFFER POOL gbpname REASON=100%LCON
DSNB250E csect-name A PAGE RANGE WAS ADDED TO THE
LOGICAL PAGE LIST
DATABASE NAME = dbn
SPACE NAME = spn
DATA SET NUMBER = dsno
PAGE RANGE = lowpg TO highpg
START LRSN = startlrsn
END LRSN = endlrsn
START RBA = startrba
```

### System action

A group buffer pool failure restricts access to the data assigned to that group buffer pool; it does not cause all members of the group to fail. For a loss of connectivity to the group buffer pools, applications that need access to group buffer pool data are rejected with a -904 SQL return code. Any of the following reason codes can be returned:

```
00C20204
00C20205
00C20220
```

- DB2 puts the group buffer pool in “damage assessment pending” status. The following message appears:

```
DSNB304I -DB1A csect-name GROUP BUFFER POOL
gbpname WAS SET TO 'DAMAGE ASSESSMENT PENDING' STATUS
```

- DB2 adds entries to the logical page list, if necessary.
- DB2 marks the affected table spaces, indexes, or partitions as group buffer pool recovery pending (GRECP) (DSNB320I or DSNB321I), indicates that the group buffer pool is recovering page sets (DSNB353I), and initiates recovery for the page set (DSNI006I).
- As each page set is recovered, the castout owner for the page set issues DSNI021I. After the last page set is recovered, any member that has issued a DSNB353I now issues a DSNB354I.
- After damage assessment is complete, the structure owner issues DSNB305I. The first new connection to the group buffer pool causes z/OS to reallocate the group buffer pool in the same or an alternate coupling facility as specified on the preference list in the CFRM policy.

### System programmer action

Correct the coupling facility failure. For lost connectivity to group buffer pools, DB2 automatically recovers data that was in any group buffer pool defined with AUTOREC(YES). If the automatic recovery is not successful or if any pages remain in the logical page list after recovery, notify the database administrator to recover the data from the group buffer pool by using the command START DATABASE (*dbname*) SPACENAM (*spacename*) to remove the GRECP status. You must issue separate START DATABASE commands in the following order:

1. DSNDB01
2. DSNDB06

If any table or index space that is required for authorization checking is unavailable, Installation SYSADM or Installation SYSOPR authority is required to issue the START DATABASE commands.

### Problem: Loss of group buffer pool structure (duplexed):

In your data sharing environment, you might encounter an error that is caused by the loss of a duplexed group buffer pool structure.

### System action

The following messages will appear if the secondary group buffer pool structure is lost:

```
DSNB743 csect-name DUPLEXING IS BEING STOPPED FOR GROUP BUFFER POOL gbpname
      FALLING BACK TO PRIMARY,
      REASON = STRFAILSEC
      DB2 REASON CODE = xxxxxxxx
```

```
DSNB745 csect-name THE TRANSITION BACK TO SIMPLEX MODE HAS COMPLETED FOR
      GROUP BUFFER POOL gbpname
```

The following messages will appear if the primary group buffer pool structure is lost:

```
DSNB744 csect-name DUPLEXING IS BEING STOPPED FOR GROUP BUFFER POOL gbpname
      SWITCHING TO SECONDARY,
      REASON = STRFAILPRI
      DB2 REASON CODE = xxxxxxxx
```

```
DSNB745 csect-name THE TRANSITION BACK TO SIMPLEX MODE HAS COMPLETED FOR
      GROUP BUFFER POOL gbpname
```

### System programmer action

No action is required at this time.

If there are 2 coupling facilities originally: When the CF that failed is brought back online, the structures will duplex themselves automatically if DUPLEX(ENABLED) was specified as recommended in the CFRM policy.

If there are more than 2 coupling facilities specified in the preference list of the CFRM policy for the structure: The structure will duplex itself using the remaining coupling facilities if DUPLEX (ENABLED) was specified in the structure statement of the CFRM policy.

After all coupling facilities have been restored to service, you should issue the SETXCF START,REALLOCATE at a low time of activity, to ensure that the group buffer pools are restored to their preferred coupling facilities.

### Problem: Loss of lock structure:

In your data sharing environment, you might encounter an error that is caused by a loss of lock structure.

### Symptom

Locking requests are suspended until the lock structure is rebuilt. If the lock structure cannot be rebuilt, the following messages appears:

```
DXR136I irlmx HAS DISCONNECTED FROM THE DATA SHARING GROUP
DXR143I irlmx REBUILDING LOCK STRUCTURE BECAUSE IT HAS FAILED OR AN IRLM
      LOST CONNECTION TO IT
```

### System action

DXR146I is issued for a successful rebuild of the lock structure. If the structure cannot be rebuilt, all active members of the group terminate abnormally with a 00E30105 abend code.

**Important:** If the lock structure cannot be rebuilt, the lost connectivity causes the members of the group to abend. This can happen, for example, if both coupling facilities are volatile and lose power. In this case, a group restart is required.

To avoid situations in which a group restart is necessary, put structures in nonvolatile coupling facility structures.

### System programmer action

You must fix the problem that is causing the loss of connectivity. If the problem is not an obvious one (such as a power failure), call IBM Software Support. See message DXR135E for the root cause of the problem and the corrective procedure.

After the problem is fixed, restart any failed members as quickly as possible to release retained locks. For the loss of lock structure, if the automatic rebuild occurred normally, processing can continue while you wait for the coupling facility problem to be fixed.

## Related concepts

 Coupling facility volatility (DB2 Installation and Migration)

### Problem: Loss of SCA structure:

In a data sharing environment, you might encounter an error that is caused by a loss of SCA structure.

### Symptom

The following message appears:

```
DSN7501I -DB1A csect-name SCA STRUCTURE sca-structure-name CONNECTIVITY FAILURE.
```

DB2 suspends processing until the SCA is rebuilt, using information contained in DB2 memory.

If the SCA cannot be rebuilt, the following message appears:

```
DSN7504I -DB1A csect-name SCA STRUCTURE sca-structure-name REBUILD UNSUCCESSFUL.
```

### System action

DSN7503I is issued for a successful rebuild of the SCA. If the rebuild is unsuccessful, all members of the group terminate abnormally with a 00F70600 abend code.

**Important:** If the SCA cannot be rebuilt, the lost connectivity causes the members of the group to abend. This can happen, for example, if both coupling facilities are volatile and lose power. In this case, a group restart is required.

To avoid situations in which a group restart is necessary, put structures in nonvolatile coupling facility structures.

### System programmer action

You must fix the problem that is causing the loss of connectivity. If the problem is not an obvious one (such as a power failure), call IBM Software Support. Check the termination code for the reason that the rebuild was unsuccessful.

After the problem is fixed, restart any failed members as quickly as possible to release retained locks. For the loss of SCA structure, if the automatic rebuild occurred normally, processing can continue while you wait for the coupling facility problem to be fixed.

## Related concepts

 Coupling facility volatility (DB2 Installation and Migration)

### Problem: a subset of members have lost connectivity (non-duplexed)

In a data sharing environment with simplexed structures, one or more members might lose connectivity to the coupling facility, while other members do not. This might happen if a link is detached between a system and the coupling facility.

This scenario assumes that the group buffer pool, lock, and SCA structures are simplexed, and have REBUILDPERCENT(1) specified in the CFRM policy, which is the recommended setting. With REBUILDPERCENT(1) specified, the group buffer

pool, lock or SCA structures will automatically be rebuilt into an alternate coupling facility whenever a member loses connectivity to the structure.

### Symptom

Some or all of the following messages appear, depending on which structures DB2 tries to access:

```
DSNB303E -DB1A csect-name A LOSS OF CONNECTIVITY
WAS DETECTED TO GROUP BUFFER POOL gbpname
DSNB331 csect-name REBUILD STARTED FOR GROUP BUFFER POOL gbpname
REASON = LOSSCON
DSN7501A -DB1A csect-name SCA STRUCTURE sca-structure-name
CONNECTIVITY FAILURE
DXR143I irlm-subsys-name REBUILDING LOCK STRUCTURE
BECAUSE IT HAS FAILED OR AN IRLM LOST CONNECTION TO IT
DSN7503I csect-name SCA STRUCTURE sca-strname
REBUILD SUCCESSFUL
```

### System action

The structures are automatically rebuilt into the alternate coupling facility.

When DB2 rebuilds the group buffer pool, it writes changed pages from the group buffer pool to the alternate structure specified in the CFRM policy. If DB2 determines that there is not enough space to hold the changed pages, it casts the pages out to disk instead and the following messages are issued:

```
DSNB331I
DSNB332I
DSNB333I
DSNB338I
```

For the SCA structure, the following message is issued:

```
DSN7503I csect-name SCA STRUCTURE sca-strname
REBUILD SUCCESSFUL
```

For the lock structure, the following message is issued:

```
DXR143I irlm-subsys-name REBUILDING LOCK STRUCTURE
BECAUSE IT HAS FAILED OR AN IRLM LOST CONNECTION TO IT
```

### Problem: a subset of members have lost connectivity (duplexed)

In a data sharing environment with duplexed structures, one or more members might lose connectivity to the coupling facility, while other members do not. This problem might happen if a link is detached between a system and the coupling facility.

This scenario assumes that the group buffer pool, lock, and SCA structures are duplexed. For a duplexed structure, the system will automatically fall out of duplexing when a member loses connectivity to one of the structure instances, and the system will use the structure instance with full connectivity in a simplex state.

If DUPLEX(ENABLED) is specified in the CFRM policy, then the system will attempt to automatically reestablish duplexing, if another coupling facility exists that has full connectivity and that coupling facility exists in the structure's preference list. If DUPLEX(ENABLED) is specified, but no preferred and connected

coupling facility exists, then the system will attempt to automatically reestablish duplexing when full connectivity is restored to the original coupling facility.

### Symptom

The following messages will appear if the secondary group buffer pool structure is lost:

```
DSNB743 csect-name DUPLEXING IS BEING STOPPED FOR GROUP BUFFER POOL gbpname
        FALLING BACK TO PRIMARY,
        REASON = LOSSCONNSEC
        DB2 REASON CODE = xxxxxxxx
DSNB745 csect-name THE TRANSITION BACK TO SIMPLEX MODE HAS COMPLETED FOR
        GROUP BUFFER POOL gbpname
```

The following messages will appear if the primary group buffer pool structure is lost:

```
DSNB744 csect-name DUPLEXING IS BEING STOPPED FOR GROUP BUFFER POOL gbpname
        SWITCHING TO SECONDARY,
        REASON = LOSSCONNPRI
        DB2 REASON CODE = xxxxxxxx
DSNB745 csect-name THE TRANSITION BACK TO SIMPLEX MODE HAS COMPLETED FOR
        GROUP BUFFER POOL gbpname
```

### System action

The duplexed group buffer pool, lock, and SCA structures will fall back to simplex when either copy of the duplexed structure is lost.

### System programmer action

No action is required at this time.

If there are 2 coupling facilities originally: When the CF that failed is brought back online, the structures will duplex themselves automatically if DUPLEX(ENABLED) was specified as recommended in the CFRM policy.

If there are more than 2 coupling facilities specified in the preference list of the CFRM policy for the structure: The structure will duplex itself using the remaining coupling facilities if DUPLEX (ENABLED) was specified in the structure statement of the CFRM policy.

After all coupling facilities have been restored to service, you should issue the SETXCF START,REALLOCATE at a low time of activity, to ensure that the group buffer pools are restored to their preferred coupling facilities.

### Problem: allocation failure of the group buffer pool

One problem that you might encounter is an allocation failure of the group buffer pool.

### Symptom

The following message appears:

```
DSNB301E -DB1A csect-name GROUP BUFFER POOL
gbpname CANNOT BE CONNECTED
DB2 REASON CODE = reason1
MVS IXLCONN REASON CODE = xxxx0C08
```

## System action

Applications needing access to group buffer pool data are rejected with a -904 SQL return code (reason code 00C20204).

If the group buffer pool cannot be allocated in an alternate coupling facility as specified on the preference list of the CFRM policy, no inter-DB2 RW activity can exist on the table spaces, indexes, or partitions that are assigned to this buffer pool. If the group buffer pool that cannot be allocated is group buffer pool 0, no update activity can exist on the DB2 catalog and directory.

## System programmer action

Use IFCID 0250 in performance class 20 to determine the reason for the allocation failure. If the trace indicates that the reason for the allocation failure is inadequate storage in the coupling facility, you can:

- Change the CFRM policy to decrease the amount of storage for the group buffer pool, or redefine that group buffer pool to a different coupling facility that has more storage.
- Have the database administrator reassign some of the table spaces or indexes that are using that group buffer pool to a different group buffer pool.

## Related concepts

“Changing the size of the group buffer pool” on page 221

## Problem: storage shortage in the group buffer pool

In a data sharing environment, you might encounter a problem that is caused by a storage shortage in the group buffer pool.

## Symptoms

The following message appears when the group buffer pool is 75% full:

```
DSNB319A -DB1A csect-name THERE IS A SHORTAGE OF SPACE IN GROUP  
BUFFER POOL gbpname
```

If you do not do anything to relieve the shortage, the following message appears when the group buffer pool is 90% full:

```
DSNB325A -DB1A csect-name THERE IS A CRITICAL SHORTAGE OF  
SPACE IN GROUP BUFFER POOL gbpname
```

If the group buffer pool is full, DB2 cannot write to it, and the following message appears:

```
DSNB228I csect-name GROUP BUFFER POOL gbpname  
CANNOT BE ACCESSED FOR function  
MVS IXLCACHE REASON CODE=xxxx0C17
```

Performance problems are evidence that the group buffer pool is not large enough. You can avoid having writes to the group buffer pool fail because of a lack of storage.

## System action

Some critical functions that cannot be completed can cause one or more members of the group to come down with reason code 00F70609.

## System programmer action

If you cannot increase the size of the group buffer pool, use the ALTER GROUPBUFFERPOOL command to decrease the castout thresholds. If decreasing the castout threshold negatively impacts performance, this should only be used as a temporary solution.

### Related concepts

“Group buffer pool size is too small” on page 210

## Problem: storage shortage in the SCA

In a data sharing environment, you might encounter a problem that is caused by a storage shortage in the SCA.

## Symptoms

The following message appears:

```
DSN7505I -DB1A csect-name THERE IS A SHORTAGE OF FREE STORAGE  
IN SCA STRUCTURE sca-structure-name.
```

If you do not do anything to reclaim space, such as recovering pages from the logical page list, the following message appears when the SCA is 90% full:

```
DSN7512A -DB1A csect-name THERE IS A CRITICAL SHORTAGE OF  
FREE STORAGE IN SCA STRUCTURE sca-structure-name.
```

## System action

Some critical functions that cannot be completed can cause one or more members of the group to come down with reason code 00F70609.

## System programmer action

Perform the following steps:

1. Reclaim space in the SCA by removing exception conditions. You can issue START DATABASE commands with the SPACENAM option, or use the RECOVER utility to remove pages from the logical page list.
2. Restart any failed members.

If these actions do not free up enough space, or if this problem continues to occur, you have the following options, depending on what level of z/OS and coupling facility you have.

- If **both** of the following conditions are true:
  - The SCA is allocated in a coupling facility with a CFLEVEL greater than 0.
  - The currently allocated size of the SCA is less than the maximum structure size as defined by the SIZE parameter of the CFRM policy.

You can enter the following command to increase the size of the SCA:

```
SETXCF START,ALTER,STRNAME=DSNDB0A_SCA,SIZE=newsize
```

This example assumes that the group name is DSNDB0A, and that *newsize* is less than or equal to the maximum size defined in the CFRM policy for the SCA structure.

- If **any** of the following conditions are true:
  - The SCA is allocated in a coupling facility with CFLEVEL=0.
  - The allocated size of the structure is already at the maximum size defined by the SIZE parameter in the CFRM policy.

Then you must:

1. Increase the storage for the SCA in the CFRM policy SIZE parameter.
2. Use the z/OS command SETXCF START,POLICY to start the updated policy.
3. Use the following z/OS command to rebuild the structure:

```
SETXCF START,REBUILD,STRNAME=DSNDB0A_SCA
```

- If **all** members are down, and you cannot increase the size of the SCA, you must do the following:
  1. Delete the SCA structure by using the following command:
 

```
SETXCF FORCE,STRUCTURE,STRNAME=DSNDB0A_SCA
```
  2. Increase the size of the SCA in the CFRM policy.
  3. Restart DB2 to rebuild the SCA using group restart.

### Related concepts

“Group restart phases” on page 123

### Problem: storage shortage in the lock structure

In a data sharing environment, you might encounter a problem that is caused by a storage shortage in the lock structure.

### Symptom

A DXR170I message indicates when storage reaches 50, 60, and 70% full. However, because this problem is detected along with other timer driven function, some messages with lower percentages might be missing or skipped if the coupling facility is filling up rapidly. The following message appears at increasing thresholds starting at 80% full:

```
-DB1A DXR142I irlmx THE LOCK STRUCTURE structure-name  
IS zzz% IN USE
```

### System action

DB2 continues processing, but some transactions might obtain a “resource unavailable” code because they are unable to obtain locks.

### System programmer action

First, make sure that no members are down and holding retained locks. Restarting any failed members can remove the locks retained in the coupling facility lock structure and release that space.

If a failed member is not the problem, you have two courses of action:

- Lower the lock escalation values to get fewer locks. You do this either by lowering the value on the LOCKS PER TABLE(SPACE) of installation panel DSNTIPJ or by using the LOCKMAX clause of CREATE TABLESPACE.
- Increase the size of the lock structure.

### Related concepts

“Changing the size of the lock structure” on page 176

### Deallocating structures by force

In a few exceptional cases, you might need to deallocate a structure by force to get DB2 restarted.

When you forcibly deallocate an SCA or lock structure, it causes a group restart on the next startup of DB2. DB2 can then reconstruct the SCA or lock structure from the logs during group restart.

To deallocate structures, use z/OS SETXCF FORCE commands to delete persistent structures or connections. Each DB2 structure requires a different set of commands.

- For the group buffer pools:  
SETXCF FORCE,CONNECTION,STRNAME=*strname*,CONNAME=ALL
- For the SCA:  
SETXCF FORCE,STRUCTURE,STRNAME=*strname*
- For the lock structure:  
SETXCF FORCE,CONNECTION,STRNAME=*strname*,CONNAME=ALL  
  
SETXCF FORCE,STRUCTURE,STRNAME=*strname*

**Important:** If your site is running z/OS with APAR OA02620 applied, you cannot delete failed-persistent connections to the lock structure unless you also deallocate the lock structure. Deleting failed-persistent connections without also deallocating the associated structure can result in a loss of coupling facility data. This situation can then cause undetectable losses of data integrity. APAR OA02620 protects your site from data corruption problems that can occur as a result of deleting retained locks. In doing so, the APAR also prevents extended outages that would result from long data recovery operations.

APAR OA02620 makes deleting persistent connections and structures easier. When you forcibly deallocate the lock structure, the operating system deletes failed-persistent connections to the structure for you. Instead of issuing the SETXCF FORCE command twice (once for failed-persistent connections to the lock structure and once for the lock structure itself), you need issue it only one time:

```
SETXCF FORCE,STRUCTURE,STRNAME=strname
```

## Resolution of transaction manager indoubt units of recovery

A *global transaction* is a unit of work that involves operations on DB2. All of the operations that comprise a global transaction are managed by a transaction manager, such as WebSphere® Application Server.

When the transaction manager receives transactionally demarcated requests from an application, it translates the requests into a series of transaction control commands, which it directs to the participating resource managers.

**Example:** An application requests updates to multiple databases. The transaction manager translates these update requests into transaction control commands that are passed to several resource managers. Each resource manager then performs its own set of operations on a database. When the application issues a COMMIT, the transaction manager coordinates the commit of the distributed transaction across all participating resource managers by using the two-phase commit protocol. If any resource manager is unable to complete its portion of the global transaction, the transaction manager directs all participating resource managers to roll back the operations that they performed on behalf of the global transaction.

If a communication failure occurs between the first phase (prepare) and the second phase (commit decision) of a commit, an indoubt transaction is created on the resource manager that experienced the failure. When an indoubt transaction is created, a message like this is displayed on the console of the resource manager:

```

00- 17.24.36 STC00051 DSNL405I = THREAD
- G91E1E35.GFA7.00F962CC4611.0001=217
- PLACED IN INDOUBT STATE BECAUSE OF
- COMMUNICATION FAILURE WITH COORDINATOR 9.30.30.53.
- INFORMATION RECORDED IN TRACE RECORD WITH IFCID=209
- AND IFCID SEQUENCE NUMBER=00000001

```

After a failure, the transaction manager, such as WebSphere Application Server, is responsible for resolving indoubt transactions and for handling any failure recovery. To perform these functions, the server must be restarted and the recovery process initiated by an operator. You can also manually resolve indoubt transactions with the RECOVER INDOUBT command.

## Resolving indoubt transactions and failure recovery

After a failure, the transaction manager, such as WebSphere Application Server, is responsible for resolving indoubt transactions and for handling any failure recovery. Alternatively, you can manually resolve indoubt transactions with the RECOVER INDOUBT command.

**GUIP** To manually resolve indoubt transactions:

1. Display indoubt threads from the resource manager console:

```
-DISPLAY THD(*) T(I) DETAIL
```

This command produces output like this:

```

=dis thd(*) t(i) detail
DSNV401I = DISPLAY THREAD REPORT FOLLOWS -
DSNV406I = INDOUBT THREADS -
COORDINATOR          STATUS      RESET URID          AUTHID
::FFFF:9.30.30.53..4007  INDOUBT          0000111F049A  ADMF002
V437-WORKSTATION=jaijeetsvl, USERID=admf002,
APPLICATION NAME=db2jccmain
V440-XID=7C7146CE 00000014 00000021 F6EF6F8B
                F36873BE A37AC6BC 256F295D 04BE7EE0
                8DFEF680 D1A6EFD5 8C0E6343 67679239
                CC15A350 65BFB8EA 670CEBF4 E85938E0
                06
V467-HAS LUWID G91E1E35.GFA7.00F962CC4611.0001=217
V466-THREAD HAS BEEN INDOUBT FOR 00:00:17
DISPLAY INDOUBT REPORT COMPLETE

```

In this example, only one indoubt thread exists. A transaction is identified by a transaction identifier, called an XID. The first 4 bytes of the XID (in this case, 7C7146CE) identify the transaction manager. Each XID is associated with a logical unit of work ID (LUWID) at the DB2 server. Record the LUWID that is associated with each transaction, for use in the recovery step.

2. Query the transaction manager to determine whether a commit or abort decision was made for each transaction.
3. Based on the decision of the transaction manager, recover each indoubt thread from the resource manager console by either committing or aborting the transaction. Specify the LUWID from the DISPLAY THREAD command, which you recorded in step 1. Use one of the following commands:

```

-RECOVER INDOUBT ACTION(COMMIT) LUWID(217)
-RECOVER INDOUBT ACTION(ABORT) LUWID(217)

```

These commands produce output like this:

```
=RECOVER INDOUBT ACTION(COMMIT) LUWID(217)
DSNV414I = THREAD
LUWID=G91E1E35.GFA7.00F962CC4611.0001=217 COMMIT
SCHEDULED
DSN9022I = DSNVRI '-RECOVER INDOUBT' NORMAL COMPLETION
```

4. Display indoubt threads again from the resource manager console:  
-DISPLAY THD(\*) T(I) DETAIL

This command produces output like this:

```
=dis thd(*) t(i) detail
DSNV401I = DISPLAY THREAD REPORT FOLLOWS -
DSNV406I = INDOUBT THREADS -
COORDINATOR STATUS RESET URID AUTHID
::FFFF:9.30.30.53..4007 COMMITTED-H 0000111F049A ADMF002
V437-WORKSTATION=jaijeetsv1, USERID=admf002,
APPLICATION NAME=db2jccmain
V440-XID=7C7146CE 00000014 00000021 F6EF6F8B
F36873BE A37AC6BC 256F295D 04BE7EE0
8DFEF680 D1A6EFD5 8C0E6343 67679239
CC15A350 65BFB8EA 670CEBF4 E85938E0
06
V467-HAS LUWID G91E1E35.GFA7.00F962CC4611.0001=217
DISPLAY INDOUBT REPORT COMPLETE
```

Notice that the transaction now appears as a heuristically committed transaction.

5. If the transaction manager does not recover the indoubt transactions in a timely manner, reset the transactions from the resource manager console to purge the indoubt thread information. Specify the IP address and port from the DISPLAY THREAD command.  
-RESET INDOUBT IPADDR(::FFFF:9.30.30.53..4007) FORCE

This command produces output like this:

```
=RESET INDOUBT IPADDR(::FFFF:9.30.30.53..4007) FORCE
DSNL455I = DB2 HAS NO INFORMATION RELATED TO
IPADDR 9.30.30.53:4007
DSNL454I = QUALIFYING INDOUBT INFORMATION FOR
IPADDR 9.30.30.53:4007 HAS BEEN PURGED
```




---

## Restarting DB2 after termination in a data sharing environment

After a failure or after a normal shutdown of DB2, you can restart DB2 with the START DB2 command.

You can also choose to have DB2 automatically restart after a failure by using the automatic restart manager of z/OS.

During restart, DB2 resolves inconsistent states. Restart is different in a data sharing environment in the following ways:

- Database exception states exist on both the SCA and the log in a data sharing environment. The database exception states exist solely on the log in non-data-sharing environments.
- Locks that are retained in the event of a failure must be processed.
- If the SCA or the lock structure is lost and cannot be rebuilt on another coupling facility, all members of the group come down. If this unlikely event occurs, then DB2 must perform *group restart*. Group restart is distinguished from normal

restart by the activity of rebuilding the information that was lost from the SCA or lock structure. Group restart does not necessarily mean that all members of the group start up again, but information from all non-starting members must be used to rebuild the lock structure or SCA.

- You can set the DEL\_CFSTRUCTS\_ON\_RESTART subsystem parameter to YES to automatically attempt to delete and rebuild the coupling facility structures whenever DB2 is restarted.

#### Related concepts

 Automatic restart of z/OS (DB2 Installation and Migration)

#### Related reference

 DEL CF STRUCTS field (DEL\_CFSTRUCTS\_ON\_RESTART subsystem parameter) (DB2 Installation and Migration)

## Normal restart for a data sharing member

Normal restart for a member of a data sharing group is very much the same as for a non-data-sharing DB2 subsystem. However, in a data sharing environment, locks can affect the availability of data.

### Active and retained locks

Active locks, retained locks, and retained utility locks can be held by a failed member. Locks that are held by a failed member can affect the availability of data to the other members of the group that are still running DB2 applications.

#### Active locks

When a member is active, the locks that it holds are called *active locks*. For transaction locks (L-locks), the normal concurrency mechanisms apply, including suspensions and timeouts when incompatible locks are requested for a resource. For physical locks (P-locks), DB2 uses a negotiation process to control access.

#### Retained locks

The particular concern for availability is what happens to locks when a member fails. For data sharing, active locks used to control updates to data (modify locks) become *retained* in the event of a failure. This means that information about those locks is stored in the coupling facility until the locks are released during restart. Retained locks are necessary to protect data in the process of being updated from being accessed by another active member of the group.

DB2 has various types of retained locks, among them are L-locks, page set P-locks, and page P-locks. As long as a retained lock is held by a failed member, another member cannot lock the resource in a mode that is incompatible with the mode of the retained lock on that resource. Incompatible requests from other members are suspended if you specify a non-zero value for the RETAINED LOCK TIMEOUT parameter of installation panel DSNTIPI. Incompatible requests from other members are immediately rejected if you specify a value of 0 for this parameter.

In the case of a page set P-lock, it is conceivable that an entire page set could be unavailable. For example, an X mode page set P-lock is retained if the page set is non-GBP-dependent at the time of the failure. Incompatible lock requests from other members can be processed after the retained locks are removed, which occurs during restart. To keep data available for all members of the group, **restart failed members as soon as possible**, either on the same or another z/OS system.

You can restart a failed member in “light” mode to quickly recover the locks held by that member. Restarting in light mode is primarily intended for a cross-system restart in the event of a failed z/OS system.

### Retained utility locks

When a member is running a utility, it holds a lock on the utility ID (UID) for that utility. That lock is also retained if the member fails. This means that you cannot restart a utility until the failed member is restarted and the retained lock is converted to an active lock.

#### Related concepts

“Improving concurrency” on page 158

“Protection of retained locks: failed-persistent connections” on page 127

“Restart light” on page 122

### When retained locks are reacquired or purged

The process of reacquiring or purging locks can happen at different times in the restart process, depending on the type of retained lock.

During the restart process, DB2 removes its retained locks in one of two ways:

- DB2 converts the lock to active. This is called *reacquiring* the lock, and it is what DB2 does for page set P-locks.
- DB2 purges the lock. This is what DB2 does for page P-locks and for L-locks.

The following table shows how the restart processing for locks can happen at different times.

*Table 23. Restart processing for locks*

Lock type	Processing
Page set P-lock <sup>1</sup>	Reacquired when page sets are opened for log apply. This generally happens during forward recovery or before restart completes.
Page P-lock	Purged at the end of forward recovery.
L-lock	Purged at the end of restart processing.

#### Note:

1. The earliest that a page set P-lock can become negotiable is at the end of forward recovery. If no log apply is needed, it can happen later, such as the end of restart processing.

If a member has a non-zero value for RETAINED LOCK TIMEOUT and it requests a lock that is incompatible with a retained lock, its applications can be suspended as long as the retained locks are held. Its request can go through as soon as the retained lock is purged or becomes negotiable. For example, if an application is suspended because of an incompatible retained page set P-lock, that retained lock most likely becomes active and available for negotiation at the end of forward log recovery.

## Related concepts

“Page set P-Locks” on page 186

## Restart light

The LIGHT(YES) or LIGHT(NOINDOUBTS) clause of the START DB2 command restarts a member in “light” mode and recovers the retained locks that are held by the member.

Restart-light mode is not recommended for a restart in place. It is intended only for a cross-system restart of a system that does not have adequate capacity to sustain the DB2 and IRLM in the event of a failed z/OS system.

Restart-light mode does the following:

- Minimizes the overall storage required to restart the member.
- Removes retained locks as soon as possible, except for the following:
  - Locks that are held by postponed-abort units of recovery, if the LBACKOUT subsystem parameter is set to LIGHT or LIGHTAUTO.
  - IX mode page set P-locks. These locks do not block access by other members; however, they do block drainers, such as utilities.
- Terminates the member normally after forward and backward recovery is complete. No new work is accepted.

A data sharing group started with the light option is not registered with the automatic resource manager (ARM). Therefore, ARM will not automatically restart a member that has been started with LIGHT(YES).

## Indoubt units of recovery

When you restart a member by using the LIGHT(NOINDOUBTS) clause of the START DB2 command, the member terminates without waiting for resolution of indoubt units of recovery (URs).

When you restart a member by using the LIGHT(YES) clause, and indoubt URs exist, DB2 issues message DSNR052I at the end of restart. The member continues to run in light mode to allow the indoubt URs to be recovered.

When DB2 for z/OS operates in restart light mode, it resolves indoubt URs in environments in which DB2 for z/OS is a participant relative to a local or a remote coordinator that makes the decision to commit or abort an indoubt UR. DB2 for z/OS can also be a coordinator relative to local or remote participants. As a coordinator, DB2 provides commit and abort decisions to remote participants as long as DB2 and DDF remain started while resolving participant-related indoubt URs. In either of those environments, DB2 automatically terminates after all indoubt URs have been resolved.

XA recovery processing for a remote XA Transaction Manager (XA TM) requires that the XA TM connect to the SQL port (tcpport) of the data sharing group to retrieve a list of transaction identifiers (XIDs) that are indoubt. Because the SQL port is unavailable for the restart light member, another member of the data sharing group must be available and accessible through a group DVIPA. After the available member is contacted, that member returns the member DVIPA and resynchronization port (resport) for any indoubt XIDs that are owned by the restart light member. The XA TM resolves work at the restart light member because its resynchronization port is available.

Indoubt URs can be resolved in two ways: automatically through resynchronization processing or manually.

- Automatically: Ensure that the appropriate commit coordinators (IMS or CICS subsystems that have indoubt URs on the restart-light member) are started so that they can resynchronize with the member to resolve the indoubt URs.
- Manually: Use the RECOVER INDOUBT command to manually resolve the indoubt URs.

Use the DISPLAY THREAD command to monitor the progress of indoubt UR resolution.

After the last indoubt UR is resolved, the member terminates. Alternatively, if you want to stop the member before all indoubt URs are resolved, you can issue the STOP DB2 command. DB2 issues message DSNR046I to indicate that incomplete URs still exist.

## DB2 commands and restart-light mode

The following commands are not allowed when a member is started in light mode:

- DISPLAY DATABASE, START DATABASE, STOP DATABASE
- DISPLAY RLIMIT, START RLIMIT, STOP RLIMIT
- SET SYSPARM
- START DDF, STOP DDF

### Related tasks

 [Creating an automatic restart policy \(DB2 Installation and Migration\)](#)

## Group restart phases

Group restart requires scanning the logs of each member to rebuild the SCA or retained lock information. It is recommended that you have an alternate coupling facility on which these vital structures can be automatically rebuilt in the event of a coupling facility failure.

The automatic rebuild that occurs during a coupling facility failure does not require the log scans that group restart does.

During group restart, all restarting members update the SCA or lock structure from information contained in their logs. If you do not issue a START DB2 command for all members of the group, the started members perform group restart on behalf of the non-starting members by reading their logs.

Although one member can perform restart on behalf of the group, you should restart all of the non-quiesced members together, perhaps by using an automated procedure. This shortens the total restart time. Also, because retained locks are held for non-starting members, it is best to start all members of the group for maximum data availability.

Because all members must synchronize at the end of current status rebuild (CSR) and at the end of forward log recovery, the time taken for group restart done in parallel is determined by the member that has the longest CSR and, if the lock structure is lost, by the member that has the longest forward log recovery.

When the members are synchronized after forward log recovery, backward log recovery proceeds in parallel for the started members.

The phases of group restart are generally the same as in a non-data-sharing environment, with the addition of function for group restart. The phases of group restart vary based on whether the SCA, lock structure, or both are lost, and whether information is needed from the logs of inactive members. The following table summarizes the phases, depending on which structure is lost.

*Table 24. Summary of group restart phases based on which structure is lost*

SCA lost	Lock structure lost
Initialization	Initialization
CSR (rebuild SCA)	CSR (reacquire page set P-locks)
Peer CSR (rebuild SCA)	Peer CSR (rebuild page set P-locks)
Forward-log recovery (rebuild locks)	Forward-log recovery (rebuild locks) or Peer forward recovery (rebuild locks)
Backward-log recovery	Backward-log recovery

## DB2 initialization

The initialization phase verifies BSDSs, logs, and the integrated catalog facility catalog.

In this phase, the RBA of the last log record is retrieved, and logging is set to begin at the next control interval (CI) following the last RBA. Also during this phase, DB2 determines if the lock structure or SCA is lost and needs to be recovered.

The following message output shows a group restart controlled by member DB3A on behalf of members DB1A and DB2A. During initialization, you see messages similar to the following messages:

```
$HASP373 DB3AMSTR STARTED
:
:
DSNJ127I @DB3ADB2 SYSTEM TIMESTAMP FOR BSDS= 95.040 13:03:05.32
DSNJ001I @DB3ADB2 DSNJW007 CURRENT COPY 1 ACTIVE LOG 753
DATA SET IS DSNAME=DSNC410.THIRD.LOGCOPY1.DS01,
STARTRBA=000000000000,ENDRBA=000000167FFF
DSNJ001I @DB3ADB2 DSNJW007 CURRENT COPY 2 ACTIVE LOG
DATA SET IS DSNAME=DSNC410.THIRD.LOGCOPY2.DS01,
STARTRBA=000000000000,ENDRBA=000000167FFF
DSNJ099I @DB3ADB2 LOG RECORDING TO COMMENCE WITH
STARTRBA=000000010000
:
:
$HASP373 DB3ADB1M1 STARTED
```

## Current status rebuild

When a restarting member has completed its own current status rebuild (CSR), it checks and waits for every other member to finish CSR. If non-starting members exist, peer CSR is performed.

During current status rebuild, the following tasks are accomplished:

- The SCA is rebuilt from the log by reading it forward from the last checkpoint. All restarting members add entries to the indoubt transaction ID (XID) list in the SCA from information contained in their logs. If an indoubt XID entry cannot be added to the SCA, the member abnormally terminates with reason code 00F70606.
- DB2 determines all outstanding units of recovery (URs) that were interrupted by the previous termination.

- If the lock structure is lost, all partition and page set P-locks are reacquired by reading information from the log. These locks are retained locks until the end of restart.

During CSR, you see messages similar to the following messages. (The phrase in parentheses is not part of the output.)

```
DSNR001I @DB3ADB2 RESTART INITIATED
DSNR003I @DB3ADB2 RESTART...PRIOR CHECKPOINT RBA=00000000DC4E
DSNR004I @DB3ADB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of current status rebuild for member DB3A)

DSNR021I @DB3ADB2 DSNRRGRC DB2 SUBSYSTEM MUST PERFORM
GROUP RESTART FOR PEER MEMBERS
```

## Peer CSR

Peer CSR is skipped unless it is necessary to perform group restart on behalf of non-starting members. Peer CSR is not performed on non-starting members that are normally quiesced.

A restarting member can select an inactive member on which to perform peer initialization and peer CSR:

- If the SCA is lost, the restarting member rebuilds SCA information from the information contained in the non-starting member's logs.
- If the lock structure is lost, the restarting member reacquires page set and partition P-locks (as retained locks) for the non-starting member. Those locks are now retained locks.

During peer CSR, you see messages similar to the following messages. (The phrases in parentheses are not part of the output.)

```
DSNR023I @DB3ADB2 DSNRRGRC GROUP RESTART INITIATED TO
RECOVER THE SCA FOR GROUP MEMBER DB1A
DSNR003I @DB3ADB2 RESTART...PRIOR CHECKPOINT RBA=00000201CC4E
DSNR004I @DB3ADB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
DSNR024I @DB3ADB2 DSNRRGRC GROUP RESTART COMPLETED TO
RECOVER THE SCA FOR GROUP MEMBER DB1A
(End of peer current status rebuild for member DB1A)
```

```
DSNR023I @DB3ADB2 DSNRRGRC GROUP RESTART INITIATED TO
RECOVER THE SCA FOR GROUP MEMBER DB2A
DSNR003I @DB3ADB2 RESTART...PRIOR CHECKPOINT RBA=000000009C4E
DSNR004I @DB3ADB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
DSNR024I @DB3ADB2 DSNRRGRC GROUP RESTART COMPLETED TO
RECOVER THE SCA FOR GROUP MEMBER DB2A
(End of peer current status rebuild for DB2A)
```

```
DSNR022I @DB3ADB2 DSNRRGRC DB2 SUBSYSTEM HAS
COMPLETED GROUP RESTART FOR PEER MEMBERS
(End of peer processing)
```

When all members have completed CSR—either by performing it on their own or by having a peer perform it for them—the SCA has been rebuilt, and page set and partition P-locks have been reacquired.

## Forward-log recovery

In the forward-log recovery phase, DB2 applies log records and completes any database write operations that were outstanding at the time of the failure.

It also rebuilds retained locks during this phase by reading that information from the log. Restart time is longer when lock information needs to be recovered during a normal restart, because DB2 needs to go back to the earliest begin\_UR for an inflight UR belonging to that subsystem. This is necessary to rebuild the locks that member has obtained during the inflight UR. (A normal restart goes back only as far as the earliest RBA that is needed for database writes or is associated with the begin\_UR of indoubt units of recovery.)

If a problem exists that prevents an object's log record from being applied (for example, if the disk version of the data could not be allocated or opened), or if the page set is deferred, DB2 adds the relevant pages and page ranges to the logical page list. Only pages affected by the error are unavailable.

When each restarting member has completed its own forward-log recovery, it checks and waits for all members to finish. If there are non-starting members, peer forward-log recovery is performed.

During forward-log recovery, you see messages similar to the following messages:

```
DSNR005I @DB3ADB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR021I @DB3ADB2 DSNRRGRH DB2 SUBSYSTEM MUST PERFORM
GROUP RESTART FOR PEER MEMBERS
```

### Peer forward-log recovery

Peer forward-log recovery is skipped unless it is necessary to rebuild lock information from information contained in inactive, non-quieted members' logs.

Peer retained-lock recovery requires that DB2 do a peer initialization, a partial CSR phase to rebuild UR status, and then do the forward-log recovery for the non-started member.

During peer forward-log recovery, you see messages similar to the following messages. (The phrases in parentheses are not part of the output.)

```
DSNR025I @DB3ADB2 DSNRRGRH GROUP RESTART INITIATED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB1A
DSNR003I @DB3ADB2 RESTART...PRIOR CHECKPOINT RBA=00000201CC4E
DSNR004I @DB3ADB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of peer partial current status rebuild for DB1A)
```

```
DSNR005I @DB3ADB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR026I @DB3ADB2 DSNRRGRH GROUP RESTART COMPLETED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB1A
(End of peer forward-log recovery for member DB2A)
```

```
DSNR025I @DB3ADB2 DSNRRGRH GROUP RESTART INITIATED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB2A
DSNR003I @DB3ADB2 RESTART...PRIOR CHECKPOINT RBA=000000009C4E
DSNR004I @DB3ADB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of partial current status rebuild for member DB2A)
```

```
DSNR005I @DB3ADB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR026I @DB3ADB2 DSNRRGRH GROUP RESTART COMPLETED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB2A
```

```
DSNR022I @DB3ADB2 DSNRRGRH DB2 SUBSYSTEM HAS  
COMPLETED GROUP RESTART FOR PEER MEMBERS  
(End of peer forward-log recovery for member DB1A)
```

When all members have completed forward-log recovery—either by performing it on their own or by having a peer perform it for them—the lock structure has been rebuilt.

## Backward-log recovery

During the backward-log recovery phase, DB2 completes recovery processing by reversing all changes performed for inflight and in-abort units of work.

At this point, all forward-log applies have been performed, and inflight, indoubt, and in-abort transactions are protected by locks. Any updates that cannot be externalized to the group buffer pool or to disk cause the affected pages to be added to the logical page list.

During backward-log recovery, you see messages similar to the following messages. (The phrase in parentheses is not part of the output.)

```
DSNR006I @DB3ADB2 RESTART...COUNTS AFTER BACKWARD  
RECOVERY  
INFLIGHT=0, IN ABORT=0  
(End of backward-log recovery for member DB3A)  
DSNR002I @DB3ADB2 RESTART COMPLETED  
DSN9022I @DB3ADB2 DSNYASCP 'START DB2' NORMAL COMPLETION
```

The backward-log recovery messages for other members do not appear until those members are actually started. Backward-log recovery can occur in parallel for all the started members. No peer backward-log recovery exists; all members must be started to complete backward-log recovery and to release the locks held by inflight and in-abort transactions.

## Protection of retained locks: failed-persistent connections

Use extreme care when deleting failed-persistent connections to the lock structure. IRLM and XES use failed-persistent connections to the lock structure to track retained lock information. Retained locks might be lost and data integrity might be exposed by arbitrarily deleting a failed-persistent lock structure connection.

You should delete failed-persistent lock structure connections only in the following situations:

- Disaster recovery. All DB2 and IRLM-related failed-persistent connections and structures must be deleted before restarting the data sharing group at the remote site. During the restart, DB2 uses the group restart process to rebuild the retained locks from the logs.
- A Parallel Sysplex-wide outage when the lock structure is forced. Failed-persistent connections can be safely forced when all members are down and the lock structure is also forced. During the restart phase of a disaster recovery process, DB2 uses the group restart process to rebuild the retained locks from the logs.
- After a hard failure occurs, such as a check-stop or abnormal re-IPL of a z/OS image that contains an active member and the DB2 or IRLM member has not been restarted.

**Important:** This information about deleting failed-persistent connections is not relevant for sites running z/OS with APAR OA02620 applied. With this APAR, you cannot delete failed-persistent connections to the lock structure unless you also

deallocate the lock structure. Deleting failed-persistent connections without also deallocating the associated structure can result in a loss of coupling facility data. This situation can then cause undetectable losses of data integrity. APAR OA02620 protects your site from data corruption problems that can occur as a result of deleting retained locks. In doing so, the APAR also prevents extended outages that would result from long data recovery operations.

**Do not delete a member's failed-persistent connection just because that member was normally quiesced.**

When a member is shut down while holding retained locks, those retained locks are transferred to another member to hold until the original member is restarted. Therefore, although a normally quiesced member does not hold retained locks for itself, it might hold retained locks for another member that was shut down. The following situations can cause a transfer of retained locks:

- Member failure. Assume that three members, DB1A, DB2A, and DB3A, are running normally. If z/OS incurs a hard failure that takes down DB2A, DB2A's retained locks are transferred to one of the other members; assume that it was DB1A. If DB1A is subsequently shut down normally, you might assume that no locks are held by DB1A and that you can safely delete DB1A's failed-persistent connection. In actuality, because DB1A is holding the retained locks from DB2A, deleting DB1A's failed-persistent connection deletes DB2A's retained locks and exposes the DB2 data to potential data integrity errors.
- A lock structure rebuild when one or more members are down and holding retained locks. This could be caused by either the z/OS SETXCF command or a coupling facility-related failure.

A coupling facility structure rebuild deletes any failed-persistent connections that existed before the rebuild. The retained locks belonging to failed members are recreated and held during the rebuild by one of the active members until the failed members are restarted.

## Handling coupling facility connections that hang

If a connection to a coupling facility structure hangs, an operator should complete some actions to recover from the hanging connection.

When a member abnormally terminates, its connections to coupling facility structures are put into a FAILING state by cross-system extended services for z/OS (XES). The member remains in this FAILING state until all surviving members of the group have responded to the XES Disconnected/Failed Connection (DiscFailConn) event for each structure. XES sends this event to each surviving member of the group so that the surviving members can take the necessary recovery actions in response to the failed member.

After all surviving members of the group perform the necessary recovery actions and provide DiscFailConn responses to XES for a given coupling facility structure, XES changes the failed member's connection status for that coupling facility structure from FAILING to FAILED PERSISTENT. The member can reconnect to the coupling facility structure during restart when the member's status is FAILED PERSISTENT.

When you restart the member immediately following a connection failure, the member can attempt to reconnect to a coupling facility structure while its connection is still in a FAILING state. If this occurs, XES denies the reconnect

request with a 0C27 reason code. DB2 responds to this by entering a connection-retry loop until the connection succeeds or until it reaches the maximum retry count.

For the SCA, the maximum retry count is 200 times with a 3-second interval between each attempt. For the group buffer pools, the maximum retry count is 5 times with a 10-second interval between each attempt. You might notice a message similar to the following message, which indicates a failed connection attempt:

```
IXL013I IXLCONN REQUEST FOR STRUCTURE DB2GR0W_SCA FAILED.  
JOBNAME: DB2VMSTR ASID: 05E1 CONNECTION NAME: DB2_DB2V  
IXLCONN RETURN CODE: 0000000C, REASON CODE: 02010C27
```

The preceding message might be displayed multiple times while DB2 is in a connection-retry loop. This is normal.

In rare cases, one or more of the surviving members of a group encounters difficulties in providing the DiscFailConn response to XES for a given coupling facility structure. When this happens, XES issues a message similar to the following message for each member from which it does not receive a response within two minutes:

```
IXL041I CONNECTOR NAME: DB2_DB2M, JOBNAME: DB2MMSTR, ASID: 0086  
HAS NOT RESPONDED TO THE DISCONNECTED/FAILED CONNECTION EVENT FOR  
SUBJECT CONNECTION: DB2_DB2V.  
DISCONNECT/FAILURE PROCESSING FOR STRUCTUR DB2GR0W_SCA  
CANNOT CONTINUE.  
MONITORING FOR RESPONSE STARTED: 08/08/2002 23:50:23.  
DIAG: 0000 0000 00000000
```

In extreme cases, the maximum number of connection retries might be reached. If encountered for the SCA, this situation prevents the failed member from restarting and DB2 issues a message similar to the following message:

```
DSN7506A -DB2V DSN7LSTK  
CONNECTION TO THE SCA STRUCTURE DB2GR0W_SCA FAILED.  
MVS IXLCONN RETURN CODE = 0000000C,  
MVS IXLCONN REASON CODE = 02010C27.
```

To recover from coupling facility structure connections that hang:

1. Save a dump of all DB2 and IRLM members along with SDATA= (COUPLE, XESDATA) so that IBM Software Support can determine what is causing the hung connections. See message III10850 for more information.
2. Attempt a REBUILD of the lock structure. This can sometimes clear the condition that is causing the DiscFailConn response to hang. If the REBUILD of the lock structure works, XES issues a message similar to the following message for each group member as it provides the required DiscFailConn response:

```
IXL043I CONNECTION NAME: DB2_DB2M, JOBNAME: DB2MMSTR, ASID: 0086  
HAS PROVIDED THE REQUIRED RESPONSE. THE REQUIRED RESPONSE  
FOR THE DISCONNECTED/FAILED CONNECTION EVENT  
FOR SUBJECT CONNECTION DB2_DB2V,  
STRUCTURE DB2GR0W_SCA IS NO LONGER EXPECTED.
```

If the REBUILD does not work, proceed to step 3.

3. Issue the D XCF,STR,STRNM=<strname>,CONNM=<conname> command for the structure or connector that is in the FAILING state. Alternatively, issue the D XCF,STR,STRNM=<strname>,CONNM=ALL command. Both commands display the status of the structures and connectors that are used by XES.

If this command identifies the unresponsive members, skip to Step 6. If it does not identify the unresponsive members, proceed to Step 4.

4. Attempt a structure REBUILD for the affected structure, if you have not already done this.
5. If the REBUILD hangs, issue the `D XCF,STR,STRNM=<strname>` command to identify the unresponsive connector.

This identifies the members that are unresponsive to the REBUILD. These members are probably the same members that are unresponsive to the DiscFailConn event.

6. Cancel and recycle the unresponsive members. The `STOP D` command might not work because internal DB2 processes are hung, so cancel IRLM or DB2 MSTR.

As each member terminates, verify that XES issues message IXL043I to indicate that it no longer expects a DiscFailConn response from that member. When all members that owe responses have been stopped, all connections to the SCA should be ACTIVE or FAILED PERSISTENT.

7. Issue the `D XCF,STR,STRNM=<sca>,CONNM=ALL` command to verify the status of the connections to SCA.
8. Restart all members with FAILED PERSISTENT connections.

As each member successfully reconnects to the SCA, XES issues message IXL014I. If a problem still exists, proceed to step 9.

9. Stop and restart the systems on which the unresponsive members are running. If restarting the system does not fix the unresponsive members, proceed to step 10.
10. Cancel and recycle all connectors to the coupling facility structure. If a problem still exists, proceed to step 11.
11. Stop and restart all systems.

#### Related information

 [z/OS MVS Recovery and Reconfiguration Guide](#)

## Postponed backout in a data sharing environment

Two options on the DSNTIPL1 installation panel allow you to enable and control postponed backout.

You can postpone backout processing for in-abort and inflight units of recovery (URs). The advantage of postponing backout processing is that DB2 can be up and accepting new work before handling backout processing.

The following fields on installation panel DSNTIPL1 enable and control postponed backout:

- LIMIT BACKOUT

The AUTO, YES, LIGHT, and LIGHTAUTO values of the LIMIT BACKOUT field enable postponed backout. AUTO and LIGHTAUTO are the recommended values because you do not have to remember to issue commands to start backout processing. The AUTO and LIGHTAUTO values result in the same behavior for normal DB2 restart. However, the LIMIT BACKOUT field is ignored when AUTO is specified during DB2 restart with the LIGHT(YES) or LIGHT(NOINDOUBTS) option.

- BACKOUT DURATION

A multiplier that is used to calculate how much log processing to back out during restart. This option is valid only when AUTO, YES, LIGHT, or LIGHTAUTO is specified for the LIMIT BACKOUT field.

#### Related tasks

➤ Resolving postponed units of recovery (DB2 Administration Guide)

#### Related reference

➤ BACKOUT DURATION field (BACKODUR subsystem parameter) (DB2 Installation and Migration)

➤ LIMIT BACKOUT field (LBACKOUT subsystem parameter) (DB2 Installation and Migration)

### Why postponed backout works in a data sharing environment

While a member is restarting, it cannot accept new work. If a restarting member has much backout work to perform, and the work is not postponed, the restart can be time-consuming.

In a data sharing environment with spare capacity, you can reroute work to another member of the group. However, if the other system does not have enough capacity, or if you are not able to switch workloads because your configuration is such that there is a strong one-to-one relationship between a member and the workload that runs on that member, postponing backout processing for long-running URs can shorten the outage significantly.

### What data is unavailable?

One difference between the non-data-sharing and the data sharing implementation of postponed backout is the degree of data unavailability.

In non-data-sharing, DB2 places any page set or partition that has pending backout work into a restrictive status called *restart pending*, which blocks access to data at the page set or partition level.

In data sharing, no restrictive status is set. Access to data with pending backout work is blocked by transaction locks that persist through restart. The following retained locks persist through restart when postponed backout processing is active:

- Retained transaction locks held on page sets or partitions for which backout work has not been completed
- Retained transaction locks held on tables, pages, rows, or LOBs of those table spaces or partitions

The retained transaction locks on any particular page set or partition are freed when all URs using that page set or partition have completed their backout processing. Until that happens, the page set or partition is placed in an advisory status called *advisory restart-pending* (AREST).

### Identifying objects in advisory restart-pending status

Objects with backout work pending display a status of AREST (advisory restart-pending).

To identify objects in advisory restart-pending status:

1. Use the DISPLAY GROUP command to determine if a member has work pending,

The following statuses indicate that work is pending for that member:

**A I** This active member has indoubt URs, URs for which backout work is postponed, or both.

**Q I** This quiesced member has indoubt URs, URs for which backout work is postponed, or both.

2. Use the `DISPLAY THREAD TYPE(INDOUBT)` and `DISPLAY THREAD TYPE(POSTPONED)` commands to determine whether the pending work is indoubt URs or postponed backout URs.
  - To recover indoubt URs, use the `RECOVER INDOUBT` command.
  - To recover postponed-abort URs, use the `RECOVER POSTPONED` command. (If you specify `AUTO` or `LIGHTAUTO` for the `LBACKOUT` subsystem parameter, DB2 automatically recovers the postponed URs after a normal restart. Starting DB2 with the `LIGHT(YES)` or `LIGHT(NOINDOUBTS)` option causes postponed-abort URs to be automatically recovered unless `LIGHTAUTO` or `LIGHT` is specified for the `LBACKOUT` subsystem parameter.)
3. Use the `DISPLAY DATABASE` command to determine which objects have backout work pending (status of `AREST`).

The `AREST` status is removed when the backout processing is complete for the object. You cannot use the command `START DATABASE` with `ACCESS(FORCE)` to remove the advisory status.

Utilities are not restricted by the `AREST` status, but any write claims that are held by postponed-abort URs on the objects in `AREST` status prevent draining utilities from accessing that page set.

#### Related reference

 `-DISPLAY DATABASE (DB2)` (DB2 Commands)

 `-DISPLAY GROUP (DB2)` (DB2 Commands)

 `-DISPLAY THREAD (DB2)` (DB2 Commands)

 `-START DB2 (DB2)` (DB2 Commands)

 `LIMIT BACKOUT` field (`LBACKOUT` subsystem parameter) (DB2 Installation and Migration)

## Restarting a member with conditions

Some installations use conditional restart to bypass a long active UR backout, such as might occur when a long-running batch job fails without having issued interim commits. In data sharing environments, this use of conditional restart is not recommended.

It is safer and provides better availability to reroute work to another member or to postpone backout processing rather than suffer the total outage necessary for a conditional restart.

If you do perform conditional restart, you need to stop all members of the group except the one that is conditionally restarting to ensure that applications on those other members do not change the data that is not locked by the restarting member.

You might, in unusual circumstances, choose to make inconsistent data available for use without recovering it. This might be the case for certain test groups, for example, where data consistency is not important.

### Related tasks

 [Using locks for data consistency \(DB2 Administration Guide\)](#)

### Performing a cold start

In some cases, you might need to perform a cold start in a data sharing environment. A cold start occurs when STARTRBA and ENDRBA are equal on the CRESTART statement of the DSNJU003 (change log inventory) utility.

To cold start DB2 in a data sharing environment:

1. Stop all other members of the data sharing group.
2. Cold start the chosen member using ACCESS(MAINT). The cold start deallocates the group buffer pools to which this member was connected.
3. Resolve all data inconsistency problems resulting from the cold start.
4. Start all the other members and restart this one without ACCESS(MAINT).

### Related tasks

 [Resolving inconsistencies \(DB2 Administration Guide\)](#)

### Truncating the log without a cold start

Use this procedure when you are truncating the log but you are not performing a cold start.

1. Stop all other members of the data sharing group.
2. Conditionally restart the chosen member using ACCESS(MAINT).
3. Resolve all data inconsistency problems resulting from the conditional start.
4. After you have resolved the data inconsistencies resulting from the conditional restart, start all the other members and restart this one without ACCESS(MAINT).

### Related tasks

 [Resolving inconsistencies \(DB2 Administration Guide\)](#)

## Deferring recovery during restart

It is possible to defer recovery for an object during restart. If you use the DEFER installation option, that defers the log apply processing of the object for only the member who specified that option.

All the pages that would have been applied to disk or the group buffer pool are instead added to the logical page list. This can affect the rest of the group; any member who needs a page that is on the logical page list will not be able to access that page until the object is restarted.

To make those pages available after DB2 restarts, use the START DATABASE command with the SPACENAM option.

Deferring recovery does not change restart time significantly.

---

## Starting duplexing for a structure

When duplexing starts, a period exists in which activity to the structure is quiesced. For this reason, you should start duplexing during a period of low activity in the system.

While duplexing is being established, and for the entire time duplexing is in effect, a display of the structure shows the structure as being in DUPLEXING REBUILD. The rebuild phase is called DUPLEX ESTABLISHED, which means that duplexing is truly active for the structure.

You can start duplexing for a structure in one of two ways:

- Activate a new CFRM policy with DUPLEX(ENABLED) for the structure. The structure must have one or more actively-connected DB2 instances that support duplexing, and no connectors that do not support duplexing. If the structure is currently allocated, z/OS can automatically initiate the process to establish duplexing as soon as you activate the policy. If the structure is not currently allocated, the duplexing process can be initiated when the structure is allocated.
- Activate a new CFRM policy with DUPLEX(ALLOWED) for the structure. If the structure is currently allocated, use the following command to start the duplexing rebuild:

```
SETXCF START,REBUILD,DUPLEX,STRNAME=strname
```

If the structure is not currently allocated, wait until it is allocated before starting the duplexing rebuild.

#### Related concepts

“Group buffer pool monitoring with the z/OS DISPLAY XCF,STR command” on page 204

---

## Stopping duplexing for a structure

To stop duplexing, you must first decide which instance of the structure is to remain as the surviving simplex structure.

If you have a choice, use the primary structure as the surviving one. For example, the primary group buffer pool has intact page registration information. If you choose the secondary group buffer pool—which does not have intact page registration information—as the surviving structure, all this information is lost. As a result, all locally cached pages revert to an invalid state and these pages must then be refreshed from the group buffer pool or disk.

If you want to switch **temporarily** to use one of the duplexed structures as a simplex structure, use the following method:

1. **Optional:** If DUPLEX(ENABLED) is specified for the active CFRM policy, activate a new policy specifying DUPLEX(ALLOWED). For the new DUPLEX value to take effect, all other CFRM policy parameters must remain unchanged.
2. Use the SETXCF STOP,REBUILD command, specifying KEEP=OLD to use the primary structure as a simplex structure, or specifying KEEP=NEW to use the secondary structure as a simplex structure.

For example, the following command reverts to using the primary structure as the simplex structure:

```
SETXCF STOP,RB,DUPLEX,STRNAME=strname,KEEP=OLD
```

If you do not plan to reestablish duplexing for the structure in the near future, activate a new CFRM policy specifying DUPLEX(DISABLED) for the structure. Doing so, **permanently** switches to using one of the duplexed structures as a simplex structure.

If you want to perform maintenance on a coupling facility that contains primary or secondary structure instances, use the SETXCF STOP, RB, DUPLEX command and specify the target coupling facility.

#### Related tasks

“Shutting down the coupling facility”

---

## Shutting down the coupling facility

Create a plan for those cases in which it is necessary to shut down a coupling facility to apply maintenance or perform some other type of reconfiguration.

For the least disruptive shutdown, move all of your structures to another coupling facility before shutting it down. This section provides some guidelines for handling this event. For other structures in the coupling facility, see the appropriate product documentation.

Consider the following guidelines:

1. Prepare for the move:
  - Ensure that you have enough room on the alternate coupling facility for all structures you intend to move.
  - Ensure that the preference list for the group buffer pool, SCA, and lock structures contains the alternate coupling facility information.
2. Move simplex structures to the **new** coupling facility by using the following command:

```
SETXCF START,REBUILD,CFNAME=newcf,LOC=OTHER
```

This command rebuilds all structures that allow rebuild in the alternate coupling facility.

3. Deallocate duplexed structures on the original (**target**) coupling facility by using the following command:

```
SETXCF STOP,REBUILD,DUPLEX,CFNAME=targetcf
```

If the CFRM policy specifies DUPLEX(ALLOWED) or DUPLEX(DISABLED), the structure goes into simplex mode.

If the CFRM policy specifies DUPLEX(ENABLED), z/OS might try to automatically restart duplexing. If you have a third coupling facility specified in the CFRM policy, it is possible to continue duplexing during the outage of the target coupling facility. When an operator command causes the structure to drop from duplex to simplex mode, z/OS avoids automatically reduplexing the structure back into the same coupling facility from which one of the duplexed instances of the structure was just deallocated. Instead, it duplexes the structure in the third coupling facility.

4. Return the coupling facility to service by performing the following steps:
  - a. Evacuate the target coupling facility.

```
SETXCF START,REBUILD,CFNAME=cfname
```

- b. Perform the maintenance operation on the target coupling facility and bring it back into the Parallel Sysplex.

- c. Repopulate the target coupling facility with the structures that were in it before the coupling facility was brought down.

```
SETXCF START,REBUILD,POPULATECF,CFNAME=cfname
```



---

## Chapter 8. Performance monitoring and tuning

Actions such as tuning your use of locks and managing your group buffer pools can improve the performance of your data sharing group.

One of the main objectives of the data sharing function of DB2 for z/OS is to increase processing capacity while using the lower cost zSeries Parallel Sysplex technology. Work load capacity is increased by allowing many DB2 subsystems to access shared DB2 data with full integrity. DB2 data sharing has been designed to address this objective while providing balanced performance for a broad range of SQL applications.

DB2 gives you the power of data sharing while avoiding overhead, whenever possible, for such things as global locking and data caching. However, you can take additional action to reduce the performance cost of data sharing.

---

### Monitoring tools

Monitoring tools that are available for data sharing environments include Resource Measurement Facility™ (RMF™) reports, DB2 trace, and DB2 Performance Expert.

#### Resource Measurement Facility reports

The Resource Measurement Facility (RMF) provides single system and Parallel Sysplex views by reporting on collected resource usage data

- The Sysplex Summary report provides an integrated view of the entire Parallel Sysplex on one screen.
- The Response Time Distribution report contains details about the distribution of response times on a Parallel Sysplex level and includes the capability of zooming into a single system that indicates problems.
- The Coupling Facility reports include information about storage allocation, structure activity, and subchannel activity which allows you to plan for better resource utilization.
- The Shared Device report provides information about how disks and tape resources are shared among the different systems in the Parallel Sysplex.

#### Related reference

“Group buffer pool monitoring with the coupling facility activity report of RMF” on page 205

“Lock monitoring with the coupling facility structure activity report” on page 171

#### DB2 trace

DB2 writes trace records to help you monitor events in the data sharing group.

Many trace classes include information about locking and use of the group buffer pool.

## Related concepts

 Interpreting trace output (DB2 Performance)

## DB2 Performance Expert

OMEGAMON can monitor the use of shared DB2 resources such as the catalog and directory, group buffer pools, databases for entire data sharing groups, and databases for individual members of the group.

- Reports and traces with *member scope* present a group's instrumentation data member by member, without merging the data. These reports and traces are similar to those generated in a non-data-sharing environment, where each report or trace is produced for an individual DB2 subsystem.
- Reports and traces with *group scope* merge instrumentation data for individual members and present it for the entire group. The traces show events in chronological order and indicate which member is reporting the event, and the reports show events summarized for the entire group.

The following report sets provide both group-scope and member-scope reporting:

- Accounting
- Audit
- Locking
- Statistics

Group-scope reporting is also available in exception processing and graphics: you can define exception thresholds for groups, and you can create graphs showing performance trends for an entire data sharing group.

With OMEGAMON, you can have processor times reported in service units. This allows times from different central processor complex (CPC) models in the group to be normalized before they are reported.

### Related reference

“Lock monitoring with the DB2 statistics trace” on page 174

“Using the DB2 statistics trace” on page 172

---

## Improving the performance of data sharing applications

Several actions can help your resource-intensive applications run at their best in the Parallel Sysplex.

Many of the things you currently do for a single DB2 subsystem to improve response time or reduce processor consumption also hold true in the Parallel Sysplex. However, if the processing speed of the CPU is slower than that on which you are currently running, be sure to plan carefully for applications or DB2 utility processes where processor resource consumption represents the significant part of the elapsed time. Some examples are some batch applications, complex queries, and many DB2 utilities.

### General recommendation

Take advantage of data partitioning and design for parallel processing whenever possible. DB2 can use parallel processing effectively only when the data is partitioned.

## Related concepts

“Improving concurrency” on page 158

“Improving the response time for read-only queries” on page 142

 Parallel processing (DB2 Performance)

## DB2 address spaces involved in distributed data processing

With data sharing, as you increase throughput by spreading work across more systems, you can expect some processing increases. These increases are caused by the cost of managing and controlling locks, buffers, and data sets. However, these increases are limited to the percentage of your workload that is accessing data with inter-DB2 read/write interest.

The z/OS address spaces that are involved in distributed database processing with DB2 for z/OS and the purpose of each address space is as follows (*ssnm* is a placeholder for the actual DB2 subsystem name):

- *ssnm*MSTR

The system services address space is responsible not only for starting and stopping DB2 for z/OS, but for controlling local access to it.

Activities that occur in this address space include commit processing after updates, inserts, and deletes; logging; backout processing; and archiving. With DB2 data sharing, writes to the group buffer pool and the global unlocking that occurs during commit processing both occur under service request blocks (SRBs) in this address space.

- *ssnm*DBM1

The database services address space is responsible for accessing relational databases controlled by DB2 for z/OS. The input and output to database resources is performed on behalf of SQL application programs in this space.

Activities that occur in this address space include prefetching, space management, and deferred write. With DB2 data sharing, the following activities occur under SRBs in this address space:

- Castouts
- Prefetch interactions with the coupling facility
- P-lock negotiation
- Updates to SYSLGRNX
- Group buffer pool checkpoints

- *ssnm*DIST

The distributed services address space is responsible for that portion of DB2 for z/OS that provides distributed database capabilities: Distributed Data Facility (DDF).

When a distributed database request is received, DDF passes the request to *ssnm*DBM1, so that the required database I/O operations can be performed.

- *ssnm*SPAS

The DB2 stored procedures address space is responsible for processing stored procedures.

- IRLM

The IRLM address space is responsible for controlling access to database resources.

Activities that occur in this address space include global lock conflict resolution. When a system is running normally, IRLM's SRB time is substantially lower than either *ssnm*DBM1 or *ssnm*MSTR.

- Allied address space

The allied agent's address space is responsible for handling global lock requests and requests to read from the group buffer pool.

## Migration of batch applications

When migrating a batch application to a data sharing environment, ensure that you account for any changes in processor speed.

Be aware of contention that can occur when there are hot spots in the data as it is updated from multiple members of the group.

### Parallel scheduling

If a significant portion of the elapsed time of a long-running batch application is due to processor resource consumption and you are currently having scheduling problems, consider running more than one copy of the same program in parallel. You can run each copy on a different key range, typically the partitioning key of the main table.

If your batch application cannot be redesigned to run on separate partitions, consider running batch jobs on the CPC in the group that has the fastest single-central processor speed. This approach is recommended only if the application does not access the coupling facility at a high intensity.

To avoid disk contention, make sure the data partitions are placed on disk devices with separate I/O paths. Consider using the PIECESIZE option of the CREATE or ALTER INDEX SQL statements to give you more control over the data set size of nonpartitioning indexes.

## Designing table spaces for heavy insert and update activity

### MEMBER CLUSTER option

If your application does heavy sequential insert processing from multiple members of the group, consider putting the data in a table space that is defined with the MEMBER CLUSTER option. The MEMBER CLUSTER option causes DB2 to insert the data based on available space rather than respecting the clustering index or the first index. For sequential insert applications, specifying MEMBER CLUSTER can reduce P-lock contention and give better insert performance at the cost of possibly higher query times.

### TRACKMOD option

When an application is rapidly updating a single table space, contention can exist on the space map pages as DB2 tracks changes. DB2 tracks these changes to help improve the speed of incremental image copies. With TRACKMOD NO, DB2 does not track changes and thus avoids contention on the space map. This option is not recommended if you depend on fast incremental image copies for your backup strategy. If your application does heavy sequential inserts, consider also using the MEMBER CLUSTER option.

### Related concepts

“Options for reducing space map page contention” on page 189

### Related reference

 Statements (DB2 SQL)

## Resource limit facility implications for data sharing

The resource limit facility (governor) controls the amount of processor time that any dynamic, manipulative SQL statement (SELECT, INSERT, UPDATE, DELETE) can consume in DB2.

Different members of a data sharing group can use the same or different resource limit specification tables (RLSTs). Each RLST must have a unique name within the data sharing group.

### Controlling the RLST

The commands that control an RLST (DISPLAY RLIMIT, START RLIMIT, and STOP RLIMIT) affect only the member on which the command is issued. The same holds true if you start an RLST at DB2 startup by using a system parameter.

### Dropping objects in the resource limit facility

While an RLST is active in any member, you cannot drop any object associated with the RLST.

### Restrictions for STOP and START DATABASE

While an RLST is active in any member, you cannot enter a STOP DATABASE command for a database or table space that contains the RLST, nor can you start the database or table space with ACCESS(UT).

### Related concepts

“Ways to control the resources used by parallel operations” on page 155

## Removal of group buffer pool dependency

Before running large batch processes, you may need to remove group buffer pool dependency for a table space, index space, or partition to improve performance in a data sharing environment.

You can do this by issuing the ACCESS DATABASE MODE(NGBPDEP) command on the data sharing member where the batch programs run. This will attempt to non disruptively convert the specified pageset or partition to non group buffer pool dependent. The command should be issued to the member on which you plan to run the batch programs.

## Physical open of a page set of partition

To eliminate the overhead of a physical open for an SQL thread, you may want to force a physical open of a table space, index space, or partition to improve performance in a data sharing environment.

You can do this by issuing the ACCESS DATABASE MODE(OPEN) command on the data sharing member where you plan to run your applications. This command

will force the physical opening of the specified pageset or partition on the local member only. This will move the overhead of the physical open from a SQL thread to the command thread.

---

## Improving the response time for read-only queries

The response time of a complex SQL query can be constrained by processor resources when it runs on a single central processor.

To improve the response time, run a complex query in parallel within a member of a data sharing group, or, with Sysplex query parallelism, use the full power of the data sharing group to process the query on multiple members simultaneously.

Applications that are primarily read-only and processor-intensive benefit from Sysplex query parallelism. If a query qualifies for parallel processing, DB2 determines the optimal degree of parallelism at bind time. DB2 can then distribute parts of the query for processing on different members of the data sharing group at run time.

**Terminology:** The member that is attached to the application that issued the query is the *parallelism coordinator* or *coordinating member*. A member who helps process part of the query is a *parallelism assistant* or *assisting member*.

**How data is returned to the parallelism coordinator:** Data is returned to the parallelism coordinator in one of two ways:

- By work files  
If the query requires the use of work files (because the data needs to be sorted, for example), the parallelism coordinator can access the data in the work files of the parallelism assistants. Each assistant writes to its own work file, and the coordinator reads the results from the assistants' work files.
- By cross-system coupling facility (XCF) links  
For data that does not require the use of work files, XCF is used. XCF traffic can be transported using channel-to-channel connections between the CPCs, or by using the coupling facility.

### Related tasks

 [Interpreting query parallelism \(DB2 Performance\)](#)

## Planning for Sysplex query parallelism

Before you use Sysplex query parallelism in a data sharing environment, you must configure the data sharing group members and define how z/OS will handle the work.

### Configuring the data sharing group members

Before you can use Sysplex query parallelism, you must configure the members of the data sharing group to use it.

In summary:

- The query must originate on a member for which YES is specified for the COORDINATOR parameter of installation panel DSNTIPK. The COORDINATOR parameter controls whether the member can send parallel tasks to other members of the data sharing group. If the value of the COORDINATOR

parameter is NO at run time, the query runs only on that single member. This is shown in field **C** of the OMEGAMON accounting report in the partial accounting trace.

- To be considered for the role of parallelism assistants, each of the other members must have YES specified in the ASSISTANT parameter of installation panel DSNTIPK. The ASSISTANT parameter controls whether the member can receive parallel tasks from other members of the data sharing group. If this member is the parallelism coordinator for a particular query, the value of its ASSISTANT parameter is not relevant.
- Put work files on shared disks that are accessible by all members that can process parallel queries. Keep work files in a separate buffer pool that is not backed by a group buffer pool. Use the same buffer pool number for all members of the data sharing group. At run time, DB2 assumes that all work files are in the same numbered buffer pool.
- Define group-wide goals for workload management of work that originates on a particular member and of work that is processed by that member on behalf of another.
- Individual member buffer pools that can be used for processing queries from other members must have a VPXPSEQT buffer pool threshold greater than 0. Because VPXPSEQT is a subset of VPPSEQT and VPSEQT, these values must also be greater than 0.
- Designate only one member on a given CPC as an assistant for optimal use of processor resources. No benefit exists for splitting a query across multiple members on the same CPC.

#### **Related concepts**

“Buffer pool threshold for parallelism assistants” on page 146

“How the accounting trace monitors processor use” on page 151

#### **Related tasks**

“Setting workload management goals”

### **Setting workload management goals**

You should define how you want z/OS to handle the work of Sysplex query parallelism.

In addition to classifying work that runs on the coordinating member, you must also classify work that runs on assisting members. If you do not classify a member as an assistant, that part of the query that runs on the assistant is discretionary work. Discretionary work runs at the priority usually reserved for very low-priority batch work.

To set workload management goals for parallel queries with which a member assists, you must:

1. Define a service class that is used for queries that run on the assistant.  
This service class can have the same goals as those of the coordinator. Alternatively, you can apply modified goals to the assistants to take into account the numerous amount of enclaves per assistant for a single query.
2. Install the service definition and activate the service policy.

#### **Related concepts**

 MVS Planning: Workload Management

#### **How period switches work on parallelism assistants:**



```

*****
Subsystem-Type Xref Notes Options Help
-----
Modify Rules for the Subsystem Type          Row 1 to 2 of 2
Command ==> _____ SCROLL ==>
Subsystem Type . : DB2          Fold qualifier names?  Y (Y or N)
Description . . . Participants in complex queries
Action codes: A=After      C=Copy      M=Move      I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
-----Qualifier-----
Action  Type      Name      Start      Service      Report
-----
_____ 1 SI      TSO      _____  ZTSODB2      _____
_____ 1 SI      JES      _____  ZJESDB2_     _____
***** BOTTOM OF DATA *****

```

Figure 29. Classifying query work on the assistants. The DB2 in the SUBSYSTEM TYPE field specifies that these classifications are for query work originating from another member of the data sharing group.

The following figure shows another way that work can be classified. It more clearly illustrates the relationship between the work that is classified on the parallelism coordinator and the work that is classified on the parallelism assistant. The SUBSYSTEM TYPE is DB2 and the service class names are different.

```

*****
Subsystem-Type Xref Notes Options Help
-----
Modify Rules for the Subsystem Type          Row 1 to 7 of 7
Command ==> _____ SCROLL ==> PAGE
Subsystem Type . : DB2          Fold qualifier names?  Y (Y or N)
Description . . . Participants in complex queries
Action codes: A=After      C=Copy      M=Move      I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
-----Qualifier-----
Action  Type      Name      Start      Service      Report
-----
_____ 1 SI      TSO      _____  ZRTDISC      _____
_____ 2  UI      EMMES    _____  ZRTVEL      _____
_____ 3  AI      D44*    _____  _____    PROJ1
_____ 3  AI      D88*    _____  _____    PROJ2
_____ 2  UI      BRYAN   _____  ZRTONLY     _____
_____ 3  AI      P73*    _____  _____    PROJ1
_____ 1 SI      JES      _____  ZRTDISC_    PROJ2
***** BOTTOM OF DATA *****

```

Figure 30. Classifying work for the parallelism assistants

**Related concepts**

“Example: setting goals for the parallelism coordinator” on page 144

**Way to display buffer pool thresholds**

The DB2 command DISPLAY BUFFERPOOL displays all buffer pool thresholds, including the assisting parallel sequential threshold.

 The following figure shows the DISPLAY BUFFERPOOL command displaying all buffer pool thresholds.

```

-DB1A DISPLAY BPOOL(BP1)
DSNB401I -DB1A BUFFERPOOL NAME BP1, BUFFERPOOL ID 1, USE COUNT 76
DSNB402I -DB1A VIRTUAL BUFFERPOOL SIZE = 20000 BUFFERS
          ALLOCATED      = 20000   TO BE DELETED   =      0
          IN-USE/UPDATED = 16345
DSNB406I -VIRTUAL BUFFERPOOL TYPE -
          CURRENT        = PRIMARY
          PENDING        = PRIMARY
          PAGE STEALING METHOD = LRU
DSNB404I -DB1A THRESHOLDS -
          VP SEQUENTIAL  =100
          DEFERRED WRITE = 85   VERTICAL DEFERRED WRT = 80
          PARALLEL SEQUENTIAL = 50   ASSISTING PARALLEL SEQT=100
DSNB9022I -DB1A DSNB1CMD '-DISPLAY BPOOL' NORMAL COMPLETION

```

Figure 31. Displaying buffer pool thresholds. In this particular buffer pool, 50% of the buffer space is available for parallel processing. All of that parallel space is available to assist with processing queries from other members of the data sharing group.



### Buffer pool threshold for parallelism assistants:

Use the *assisting parallel sequential threshold* (VPXPSEQT) value to tell DB2 how much of the buffer pool parallel sequential threshold (VPPSEQT) applies to assisting a coordinating member with processing parallel queries.

The following figure shows the relationship among the various buffer pool thresholds.

Example:

```
-DB1G ALTER BUFFERPOOL(BPn) VPSIZE(1000) VPSEQT(80) VPPSEQT(50) VPXPSEQT(50)
```

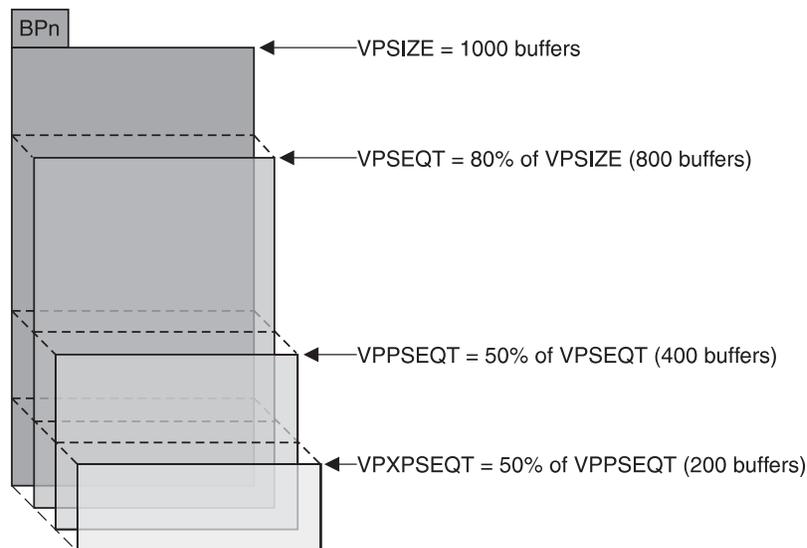


Figure 32. Relationships of buffer pool thresholds. The threshold that is used to determine the allocation of resources for Sysplex query parallelism is determined by VPXPSEQT, which is a subset of the VPPSEQT threshold.

Sample configurations:

Looking at sample configurations can be helpful when you set up your own buffer pool parallel sequential threshold (VPPSEQT) values.

The following figure shows how the online systems (IMS and CICS) are configured for maximum transaction throughput. These systems send all of their parallel queries to the query processor members on CPC7 and CPC8, and they are not configured to act as parallelism assistants.

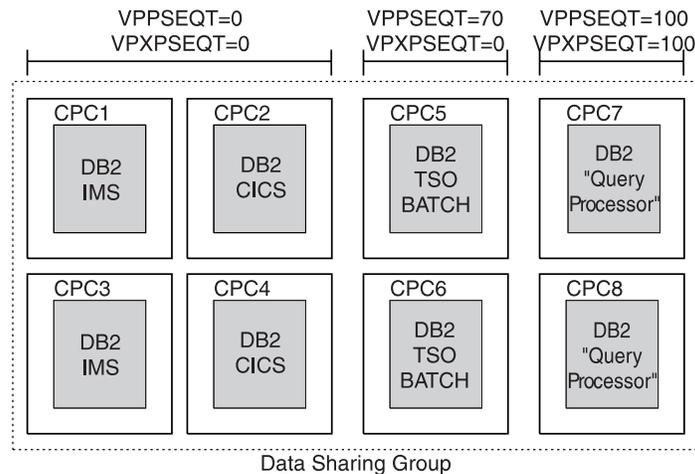
- Their COORDINATOR subsystem parameters are set to YES.
- Their ASSISTANT subsystem parameters are set to NO.

The TSO and BATCH systems are configured to run their own queries in parallel, and they can send parallel queries to the query processor members on CPC7 and CPC8. Like the online systems, they are not configured to act as parallelism assistants.

- Their COORDINATOR subsystem parameters are set to YES.
- Their ASSISTANT subsystem parameters are set to NO.

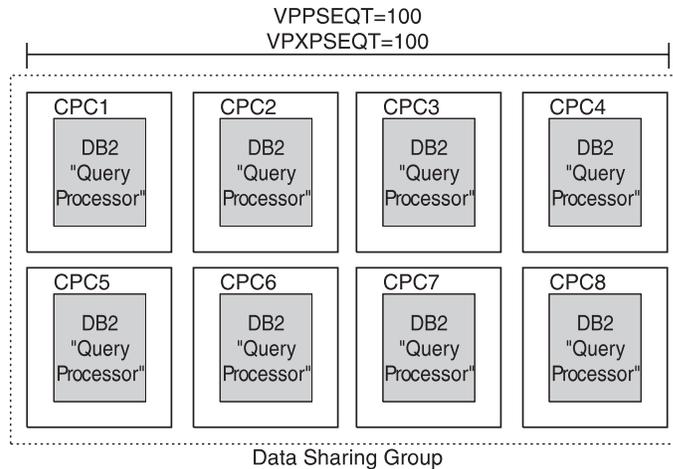
The query processor systems are the only members that have allocated buffer pool space to assisting other members with parallel query processing.

- Their COORDINATOR subsystem parameters are set to YES.
- Their ASSISTANT subsystem parameters are set to YES.



*Figure 33. Separate query processor configuration.* In this configuration, the designated query processor systems are the only members configured as parallelism assistants.

The following figure shows a data sharing group in which all members can coordinate and assist with parallel queries. The COORDINATOR and ASSISTANT subsystem parameters on all members are set to YES.



*Figure 34. Dedicated query processor configuration.* In this configuration, every member of the data sharing group is a designated query processor system. The virtual pool sequential threshold (VPSEQT) is set very high (100).

## Enabling parallel processing within an application

Your application requires no changes to enable parallel processing, but specific settings should be used for parallel processing of static and dynamic queries.

**GUIP** Use the following settings for parallel processing.

- If the query is static:
  - Use DEGREE(ANY) in a BIND or REBIND subcommand.
  - Use PARALLEL DEGREE ==> ANY in the DB2I DEFAULTS panel for a BIND, REBIND PACKAGE, or REBIND PLAN subcommand.
- If the query is dynamic:
  - Use SET CURRENT DEGREE = 'ANY' in a previous SQL statement.

**Tip:** Installation panel DSNTIPF allows you to set the default degree (1 or ANY) for the CURRENT DEGREE special register.

*Not all queries are eligible:* Even if you turn on parallelism, not all read-only queries are eligible. For example, queries must be bound with isolation UR or CS. Queries with result sets that contain LOBs are also not eligible for Sysplex query parallelism. To get the effect of RR or RS isolation with Sysplex query parallelism, bind with isolation CS or UR, and use a LOCK TABLE *table-name* IN SHARE

MODE statement before the query. **GUIP**

### Related tasks

 Programming for parallel processing (DB2 Performance)

## Enabling parallel processing within a data sharing group

For a member to be a parallelism coordinator, you must specify YES for the COORDINATOR parameter of installation panel DSNTIPK.

To allow a member to be a parallelism assistant, you must specify YES for the ASSISTANT parameter of the same panel. At run time, assistants must have buffer pools defined to allow parallelism, or DB2 does not send work to those members.

*Displaying coordinator and assistant parameter values:* The command DISPLAY GROUP with DETAIL option allows you to see the values of the COORDINATOR and ASSISTANT subsystem parameters for all active members of a data sharing group. For example, the following command displays output similar to the figure below:

```

GUIP
-DB1A DISPLAY GROUP DETAIL

DSN7100I -DB1A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNDB0A ) GROUPELVEL(610)
                                     GROUP ATTACH NAME(DB0A)  MODE(C)
-----
DB2      DB2      IRLM
MEMBER   ID  SUBSYS  CMDPREF  STATUS  LVL  NAME  SUBSYS  IRLMPROC
-----
DB1A     1  DB1A   -DB1A   ACTIVE  610  MVS1  DJ1A   DB1AIRLM
DB2A     2  DB2A   -DB2A   ACTIVE  610  MVS2  DJ2A   DB2AIRLM
DB3A     3  DB3A   -DB3A   QUIESCED 610  MVS3  DJ3A   DB3AIRLM
DB4A     4  DB4A   -DB4A   ACTIVE  610  MVS4  DJ4A   DB4AIRLM
-----
DB2      PARALLEL  PARALLEL
MEMBER   COORDINATOR ASSISTANT
-----
DB1A                YES      NO
DB2A                YES      YES
DB3A                ****     ****
DB4A                NO       NO
-----
SCA  STRUCTURE SIZE: 1024 KB, STATUS= AC,   SCA IN USE: 11 %
LOCK1 STRUCTURE SIZE: 1536 KB
NUMBER LOCK ENTRIES: 262144
NUMBER LIST ENTRIES: 7353, LIST ENTRIES IN USE: 0
*** END DISPLAY OF GROUP(DSNDB0A )
DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION

```

Figure 35. DISPLAY GROUP with DETAIL option

**GUIP**

**Related reference**

 -DISPLAY GROUP (DB2) (DB2 Commands)

## Monitoring and tuning parallel queries

You can monitor and tune parallel queries in a data sharing environment to improve response time.

### Ways to display data sharing information

DISPLAY THREAD can display information about parallel tasks, and associate them with their originating tasks.

**GUIP** If a thread is participating in the processing of a parallel query, issuing a DISPLAY THREAD on any parallelism assistant shows you the thread token of the originating task and the name of the coordinating member. A status of PT indicates a parallel task. All parallel tasks are displayed immediately after their corresponding originating thread.

## Displayed information about the parallelism coordinator

The following figure shows an allied, non-distributed originating thread (TOKEN=30) that is established (plan allocated) along with all of its parallel tasks, which are running on members DB1A and DB2A. Because the originating thread is running on DB1A, DB1A is the parallelism coordinator.

```
- 17.08.44          -DB1A DISPLAY THREAD(*)
- 17.08.44 STC00090 DSNV401I -DB1A DISPLAY THREAD REPORT FOLLOWS -
- 17.08.44 STC00090 DSNV402I -DB1A ACTIVE THREADS -
- NAME      ST A   REQ ID      AUTHID   PLAN      ASID TOKEN
- BATCH     T *    1 PUPPYDML   USER001  DSNTDP2  0025  30
-           PT *   612 PUPPYDML   USER001  DSNTDP2  002A  35
-           PT *   545 PUPPYDML   USER001  DSNTDP2  002A  34
-           PT *   432 PUPPYDML   USER001  DSNTDP2  002A  33
-           PT *   443 PUPPYDML   USER001  DSNTDP2  002A  32
-           PT *   252 PUPPYDML   USER001  DSNTDP2  002A  31
- DISPLAY ACTIVE REPORT COMPLETE
- 17.08.45 STC00090 DSN9022I -DB1A DSNVDT '-DB1A DISPLAY THREAD' NORMAL COMPLETION
```

Figure 36. Display on the parallelism coordinator

## Displayed information about the parallelism assistant

The following figure shows the display on a parallelism assistant. The PARALLELISM COORDINATOR field tells you which member is the coordinator. The ORIGINATING TOKEN field identifies the originating task on the coordinator.

```
- 17.10.12          -DB2A DISPLAY THREAD(*)
- 17.10.12 STC00044 DSNV401I -DB2A DISPLAY THREAD REPORT FOLLOWS -
- 17.10.12 STC00044 DSNV402I -DB2A ACTIVE THREADS -
- NAME      ST A   REQ ID      AUTHID   PLAN      ASID TOKEN
- BATCH     PT *   641 PUPPYDML   USER001  DSNTDP2  002D   6
- V443-PARALLELISM COORDINATOR=DB1A, ORIGINATING TOKEN=30
- BATCH     PT *   72 PUPPYDML   USER001  DSNTDP2  002D   7
- V443-PARALLELISM COORDINATOR=DB1A, ORIGINATING TOKEN=30
- BATCH     PT *   549 PUPPYDML   USER001  DSNTDP2  002D   8
- V443-PARALLELISM COORDINATOR=DB1A, ORIGINATING TOKEN=30
- BATCH     PT *   892 PUPPYDML   USER001  DSNTDP2  002D   9
- V443-PARALLELISM COORDINATOR=DB1A, ORIGINATING TOKEN=30
- BATCH     PT *   47 PUPPYDML   USER001  DSNTDP2  002D  10
- V443-PARALLELISM COORDINATOR=DB1A, ORIGINATING TOKEN=30
- DISPLAY ACTIVE REPORT COMPLETE
- 17.10.12 STC00044 DSN9022I -DB2A DSNVDT '-DISPLAY THREAD' NORMAL
- COMPLETION
```

Figure 37. Display on the parallelism assistant

## Displayed information about parallel tasks

To see parallel tasks, you can use an online performance monitor such as RMF Monitor III, which displays information about enclaves. Parallel tasks on assistants execute within an enclave. The RMF monitor shows the classification attributes such as the plan name, package name, SQLID, and so on. 

### Related reference

 -DISPLAY THREAD (DB2) (DB2 Commands)

## How DB2 reports parallelism information

DB2 reports information about parallelism at bind time and at run time. The information that is reported at bind time describes what DB2 plans to do. The information that is reported at run time describes what DB2 actually did.

## Bind time activity

As with parallelism in general, DB2 tries to determine a parallel degree and a parallel mode (I/O, CP, Sysplex) at bind time. DB2 looks at processor speeds and the number of central processors on each member when it determines the degree of parallelism.

Use DB2 Visual Explain to view the access path that DB2 plans to use for a particular query. An “X” in the PARALLELISM\_MODE column indicates that Sysplex query parallelism is chosen for the explainable statement. If only one member of the data sharing group is active at bind time, DB2 can still choose Sysplex query parallelism. This lets any assistants that are active at run time help with processing the query.

## Run-time activity

DB2 can use a different parallel degree at run time if any of the following events occur between bind time and run time:

- The number of active members changes, or the processor configuration changes. This event is recorded on the parallelism coordinator at run time in the QXREPOP1 field of the statistics and accounting traces. (See **F** in the partial accounting trace.)
- The amount of buffer space is not sufficient to handle the optimal degree of parallelism. This event is recorded by the parallelism coordinator at run time in the QXREPOP2 field of the statistics or accounting traces. (See **G** in the partial accounting trace.)

To determine whether work was sent to parallelism assistants, use the accounting trace. (See **A** in the partial accounting trace to see how OMEGAMON displays that information.)

To determine the **actual degree** that was used to process a specific query, use IFCID 0221 in performance trace class 8 (same as with CP parallelism). The QW0221XC field of IFCID 0221 indicates on how many members the parallel group ran. It also indicates when a particular buffer pool is constrained on a particular member. (A *parallel group* is a set of consecutive operations that executed in parallel and that have the same number of parallel tasks.) IFCIDs 0222 and 0231 include the names of members that participated in processing that parallel group.

When a member does not have enough buffer pool resources to participate in processing a query, the coordinator must skip that member when it distributes work to the group.

### Related concepts

“How the accounting trace monitors processor use”

“Techniques for improving response time” on page 152

## How the accounting trace monitors processor use

As with CP parallelism, the accounting trace record fields for processor execution time for the originating task and all of the parallel tasks must be added together to yield the total processor time used by a DB2 thread.

In order to perform this same function with Sysplex query parallelism, the accounting trace records from all members involved in the query must be assembled and used. OMEGAMON reports this total time in its accounting report and normalizes the time to the processor size of the originating task. See the

following figure for an example of a partial accounting report from OMEGAMON.

QUERY PARALLEL.	TOTAL
MAXIMUM MEMBERS	<b>A</b> 2
MAXIMUM DEGREE	13
GROUPS EXECUTED	1
RAN AS PLANNED	0
RAN REDUCED	<b>B</b> 1
ONE DB2 COOR=N	<b>C</b> 0
ONE DB2 ISOLAT	0
SEQ - CURSOR	0
SEQ - NO ESA	0
SEQ - NO BUF	<b>D</b> 0
SEQ - ENCL.SER	0
MEMB SKIPPED(%)	<b>E</b> 0
DISABLED BY RLF	NO
REFORM PAR-CONF	<b>F</b> 0
REFORM PAR-BUF	<b>G</b> 0
⋮	

Figure 38. Partial accounting trace that shows Sysplex query parallelism

### Techniques for improving response time

When DB2 runs a query on more than one DB2 member, the query elapsed time should improve when compared to running the query within a single member.

The actual elapsed time improvement is affected by dynamic factors such as processor utilization, DB2 buffer pool availability, I/O contention, and XCF capacity.

As with CP parallelism, to determine whether performance tuning is needed to further improve the elapsed time of a query, examine the elapsed time and CP execution time of each parallel task (via the IFCID 0231 class 8 performance trace record), especially the ones with the largest times.

### Is the buffer pool too small?

An indicator of buffer pool shortages are non-zero values in the QXREDGRP (see **B** in the partial accounting trace) and QXDEGBUF (see **D** in the partial accounting trace) fields in the statistics or accounting trace record. If response time goals are not being met, further analysis can help you determine which buffer pools are causing the problems. Use performance class 8 and inspect the IFCID 0221 trace record to pinpoint which buffer pools are too small. You might need to increase the size of the buffer pools or increase the number of buffers available to assist with processing query work.

### Reformulating the degree

When buffer pool resources are scarce, DB2 reformulates the degree of parallelism (indicated by **G** in the partial accounting trace) and determines a distribution scheme that works best with the reformulated degree. **G** is incremented once for each parallel group that was reformulated.

The percentage of members that were unable to perform some or all of the work is indicated by **E** in the partial accounting trace.

The interaction of these two fields ( **G** and **E** ) is best illustrated by an example. Assume that you have a four-member data sharing group of similar processor models with a planned degree of 40. In this case, DB2 might send 10 parallel tasks to each member. However, if buffer pool resources are scarce on two of the members and those members can only process five parallel tasks each, DB2 can reduce the parallel degree to 30, incrementing **G** once and setting **E** to 50%.

### Work file buffer pools

Do not forget to set the parallel sequential threshold for work file buffer pools. DB2 assumes that all of your work files are of the same size (4 KB or 32 KB) in the same buffer pool number on all members, and it makes run time decisions based on a single buffer pool. A lack of buffer pool resources for the work files can lead to a reduced degree of parallelism or cause the query to run sequentially.

### Is there I/O contention?

One possible cause of poor response time is contention for I/O resources. Contention can occur in any place in the I/O subsystem. Here are some ways to determine if and where I/O contention is causing the problem:

#### 1. Monitor

If you have previous accounting reports, look for changes in those reports. If the application has not changed (that is, the SQL Profile and the number of GETPAGES per commit are relatively constant), refer to the next step.

#### 2. Analyze

If you see increased class 3 I/O times from the accounting reports (specifically the SYNCHRON I/O and OTHER READ I/O fields from a OMEGAMON report), check whether the number of I/O operations per commit has increased. If so, consider some form of buffer pool tuning. By increasing the size of the pool, isolating the data into a separate pool, or tuning thresholds, you might be able to reduce the number of I/O operations and speed up the remaining I/O operations in the system.

If tuning the buffer pool does not solve the problem, try to pinpoint the I/O trouble spot. Use the following reports:

- SMF type 42 records, subtypes 5 and 6, show the response times by data set for an interval.
- The RMF Direct Access Device Activity Report shows response time by volume for an interval. It also details where in the I/O subsystem the time is spent. The detail report is of the components of the average response time to the volume. Use this information to find the bottlenecks in the setup or capacity of the I/O subsystem.

#### 3. Relieve the constraint

When you have determined where the problem is occurring, you can take steps to relieve the constraint. This could be a matter of adding to the I/O subsystem by taking such actions as adding extra channel paths to the disk controller, adding storage directors in the disk controller, adding extra cache/NVS in the controller, or adding extra disk volumes. More typically, it means that you have to move data sets from one volume to another.

### Where to place data sets

In general, you want frequently used data sets or partitions to be allocated across your available disks so that I/O operations are distributed as evenly as possible.

Ensure that device and control unit utilization is balanced. This helps balance I/O operations across the I/O configuration and takes maximum advantage of parallel I/O operations.

To determine whether the partitioning of a table or the physical placement of the partitions are reasonable, see performance trace records, IFCIDs 0221 and 0222.

The IFCID 0221 record describes how the tables within a parallel group are partitioned by specifying the key range or page range for each partition. The IFCID 0222 record shows the elapsed time statistics for each parallel operation. The elapsed time for each parallel operation within a parallel group should be comparable. An operation with a much higher elapsed time means that DB2 is performing more I/O operations on a particular logical partition than is desirable, or a significant contention exists on the disk volume or channel path.

If an uneven distribution of work exists and is causing the I/O problems, consider moving data that has low activity close to data that is more frequently accessed. Also be sure that your high-priority work is not sharing I/O resources with work that ties up the I/O subsystem.

If the elapsed times of the parallel tasks are comparable but are still too high, consider repartitioning the table space to have more partitions, assuming that processor time is not a bottleneck.

### **Is there lock contention?**

To avoid lock contention, run with an isolation level of UR, if that is possible. If it is not possible to run with an isolation level of UR, try to take advantage of lock avoidance by running with ISOLATION(CS) and specifying CURRENTDATA(NO). This can minimize the locking effect.

### **Is there XCF signaling contention?**

The resource measurement facility (RMF) of z/OS provides an XCF Activity Report that contains useful measurement data for analyzing the performance of the XCF signaling service and for doing capacity planning for a z/OS system in a Parallel Sysplex. The report shows the data collected on a system during Parallel Sysplex processing. Each system collects its own data, and the RMF on each system produces reports only about that system's data. You might need to run the RMF reports on two or more systems to get data for corresponding outbound and inbound signaling paths in order to better understand the message traffic patterns.

RMF also provides a Coupling Facility Activity report. It provides information about the usage of a coupling facility, structures within a coupling facility, and subchannel connections to a coupling facility in a Parallel Sysplex.

IRLM traces XCF messages in the IRLM Exception (EXP) trace, which you can use to tune XCF performance.

### **Is there inter-DB2 read/write interest?**

Because DB2 can split query processing onto different members, updates of the same page set cause inter-DB2 read/write interest as part of the normal data sharing integrity process. To ensure that updates on a page set are seen by the query, DB2 flushes all changed pages from the buffer pool before processing the query on other members.

Inter-DB2 read/write interest can also cause additional locking overhead. Child locks (page or row) are propagated to XES and the coupling facility based on the inter-DB2 interest of the page set P-lock. If the held state of the page set P-lock is IX, indicating inter-system read/write interest, all child locks must be propagated. To avoid locking overheads, use isolation level UR, or try to limit the table space locks to IS on all data sharing members to avoid child lock propagation.

For partitioned table spaces, DB2 can avoid locking all partitions in a table space, thus reducing the number of child locks that must be propagated. To take advantage of this, you must bind the plan or package with ACQUIRE(USE).

#### **Related concepts**

“How the accounting trace monitors processor use” on page 151

#### **Related tasks**

 [Tuning parallel processing \(DB2 Performance\)](#)

#### **Related reference**

 [z/OS MVS Setting Up a Sysplex](#)

### **Ways to control the resources used by parallel operations**

You have several options for controlling the amount of system resources that are used for processing queries that use Sysplex query parallelism.

#### **Priority**

You can use z/OS workload management to control the priority of parallel query work within a member. Pay special attention to the task of classifying work that runs on parallelism assistants.

#### **Buffer pool space**

You can control how much of the total buffer pool is used for parallel processing in general, and for Sysplex query parallelism specifically.

- The parallel threshold (VPPSEQT) determines the amount of space that is used for all parallel processing.
- The assisting parallel sequential threshold (VPXPSEQT) determines what subset of the parallel space is used to assist queries originating from another member of the data sharing group.

Response time can degrade if these thresholds are set too low.

#### **Resource limits**

You can use the DB2 resource limit facility (governor) to help control dynamic queries that use Sysplex query parallelism. Resource limits are local in scope, so DB2 can ensure that a specific dynamic query does not overrun a member. Members of a data sharing group can share the same resource limit specification table (RLST), or each member can have its own RLST. In either case, DB2 honors the limits that are specified on that member, no matter how many tasks are included in the statement.

The following figure illustrates how dynamic queries that use Sysplex query parallelism are governed by the DB2 resource limit facility.

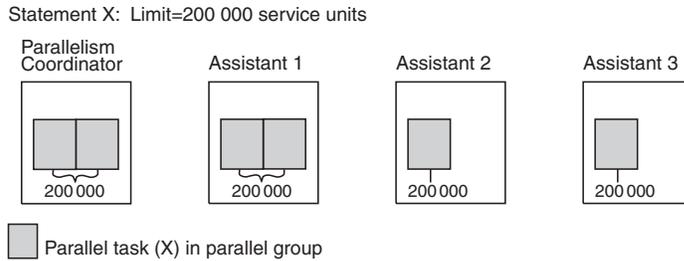


Figure 39. Governing a dynamic query that runs on four members. This figure assumes that all members share the same RLST. Statement X is not allowed to consume more than 200 000 service units on any member.

**Governing statements with more than one parallel group:** If multiple parallel groups process queries on a member, **each** parallel group can consume up to the service unit limit that is set for that member. In contrast, on the coordinator, **all** parallel groups, including the originating task, are governed by the limit on that coordinating member. The following figure illustrates how a query with two parallel groups runs at the same time.

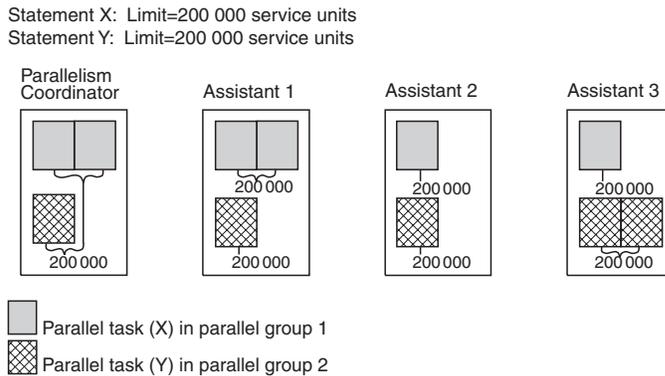


Figure 40. A query with two parallel groups running at the same time. This figure assumes that members are sharing the same RLST. No parallel group that is participating in the processing of statement Y can consume more than 200 000 service units. On the parallelism coordinator, all tasks in all parallel groups are constrained by the limit specified for the statement.

**Queries that use *INSTALL SYSADM* or *SYSOPR* authorities:** If a query is submitted by an authorization ID with *INSTALL* authority, none of the parallel tasks is governed, regardless of where the parallel tasks run.

**A statement that is executed more than once:** If the same statement executes more than once in an application, the service unit limit is applied differently to the parallelism coordinator and assistants. The service unit limit applies to all executions of the statement on the parallelism coordinator such that the cumulative number of service units consumed cannot exceed the limit. Conversely, on the parallelism assistants, each execution of the statement is subject to the service unit limit. The number of service units consumed each time the statement executes cannot exceed the limit.

#### Related tasks

“Setting workload management goals” on page 143

## Disabling Sysplex query parallelism

In addition to controls that enable or disable parallelism, you can control Sysplex query parallelism specifically.

You can disable Sysplex query parallelism on a system-wide basis by:

- Specifying COORDINATOR=NO and ASSISTANT=NO on installation panel DSNTIPK.
- Using buffer pool threshold controls for parallelism assistants.

You can also disable Sysplex query parallelism for a single dynamic query by specifying a value of '5' for the resource limit facility (governor).

**Related concepts**

“Buffer pool threshold for parallelism assistants” on page 146

**Controls for disabling Sysplex query parallelism**

Some DB2 controls, such as subsystem parameters, enable you to disable Sysplex query parallelism.

The following table summarizes the controls for disabling Sysplex query parallelism.

*Table 25. Controls for disabling Sysplex query parallelism*

Control	Checked at...	Use
COORDINATOR subsystem parameter	Bind and run time	Set the value of COORDINATOR to NO to restrict parallelism to this single member.  (See the QXCOORNO field of IFCID 0002 for cases in which the parameter was changed between bind and run time.)  Changing the value of this parameter from NO to YES requires that plans and packages be rebound before they are considered for Sysplex query parallelism.
ASSISTANT subsystem parameter	Bind and run time	Set the value of ASSISTANT to NO to prevent this member from being considered as a parallelism assistant.  (At run time, the assistant's buffer pool must be defined to allow parallelism; otherwise, the coordinator does not send work there.)  Changes from NO to YES require that plans or packages be rebound for this assistant's processing capability to affect the planned parallel degree.
Parallel buffer pool threshold (VPPSEQT) of the coordinator	Run time	Set this value to 0 to disallow query processing on this member.
Assisting parallel sequential threshold (VPXPSEQT)	Run time	Set this value to 0 to prevent this member from being considered as a parallelism assistant for any query that is using that buffer pool.  To disallow the entire subsystem from assisting with Sysplex query parallelism, all buffer pools must have VPXPSEQT=0 or VPPSEQT=0.
Governor (RLFFUNC='5')	Run time	Set this value to 5 to disable Sysplex query parallelism for a single dynamic query.

Table 25. Controls for disabling Sysplex query parallelism (continued)

Control	Checked at...	Use
BIND option DEGREE (1)	Bind time	Specify 1 for this BIND option to disable query parallelism for static queries.
Special register SET CURRENT DEGREE='1' or a 1 in the CURRENT DEGREE parameter of installation panel DSNTIPF	Run time	Set this value to 1 to disable query parallelism for dynamic queries.

## Improving concurrency

You can improve concurrency by understanding how transaction locking works in a data sharing environment and completing some actions to reduce locking overhead.

Transaction locks are often called *logical locks* (L-locks) in data sharing.

Data sharing also makes use of *physical locks* (P-locks). But, P-locks are related to caching, not concurrency, and they use different mechanisms than the transaction locks you are familiar with in DB2.

### Related concepts

“Physical locks in data sharing” on page 186

## Global transaction locking

With data sharing, concurrency control exists both within a specific member and among all members of a data sharing group.

This means that locks used in data sharing are global in scope. Many global locks are processed not only by the local IRLM but also by the cross-system extended services (XES) of z/OS and by the lock structure in the coupling facility.

### Locking optimizations

Four types of DB2 locking optimizations reduce the necessity of processing locks beyond the local IRLM whenever possible.

DB2 has the following optimizations:

- Explicit hierarchical locking makes certain optimizations possible. When no inter-DB2 read/write interest exists in an object, it is possible to avoid processing certain locks beyond the local IRLM.
- If a single member with update interest and multiple members with read-only interest exist, DB2 propagates fewer locks than when all members have update interest in the same page set.
- All locks (except L-locks) that are held beyond the local IRLM are owned by a member, not by an individual work unit. This reduces lock propagation by requiring that only the most restrictive lock mode for an object on a given member be propagated to XES and the coupling facility. A new lock that is equal to, or less restrictive than, the lock currently being held is not propagated.
- When the lock structure is allocated in a coupling facility of CFLEVEL=2 or higher, IRLM can release many locks with just one request to XES. This can occur, for example, after a transaction commits and has two or more locks that

need to be unlocked in XES. It also can occur at DB2 shutdown or abnormal termination when the member has two or more locks that need to be unlocked.

**Related concepts**

“A locking scenario” on page 160

“Explicit hierarchical locking”

**Explicit hierarchical locking**

When sharing data, DB2 uses explicit hierarchical locking to determine whether propagating L-locks beyond the local IRLM to XES and to the coupling facility is necessary.

Explicit hierarchical locking allows IRLM to grant child locks locally when no inter-DB2 read/write interest exists on the parent. Granting lock requests locally, versus globally, improves performance.

The top object in the hierarchy is a *parent*; all objects below the parent are *children*, with the caveat that a child can be the parent of another child. While a lock is held, the first lock on the top parent is always propagated to XES and the lock structure. Thereafter, only more restrictive locks are propagated. When the lock is released, the process begins again. For partitioned table spaces, each locked partition is a parent of the child locks that are held for that partition. Explicit hierarchical locking is based on the lock hierarchy that is shown in the table below.

*Table 26. Lock hierarchy.* Indexes are not included in the hierarchy because index pages are protected by locks on the corresponding data.

Parent	Children or child
Simple table space	Data pages and rows
Partitioned table space	Data pages and rows
Segmented table space	Data pages and rows
Tables in a segmented table space	N/A
LOB table space	LOB

When you use locking protocol 1, locks on child objects are based on contention on the parent L-lock. When you use locking protocol 2, locks on child objects are propagated depending on the compatibility of the page set P-lock with the page set P-locks that are held by other members for the table space.

The following table shows the conditions that cause the child locks to be propagated when locking protocol 1 is used.

*Table 27. Determining when child locks are propagated to XES when locking protocol 1 is used*

Maximum page set L-lock mode of this member is ...	And the maximum page set L-lock mode of other members is...	Are X, L-child locks propagated by this member?	Are S, L-child locks propagated by this member?	Are U, L-child locks propagated by this member?
IS, S	None, IS, S	N/A	No	N/A
X	None	N/A	N/A	N/A
IS	IX, SIX	N/A	Yes	N/A
IX, SIX	IS	Yes	No	Yes

Table 27. Determining when child locks are propagated to XES when locking protocol 1 is used (continued)

Maximum page set L-lock mode of this member is ...	And the maximum page set L-lock mode of other members is...	Are X, L-child locks propagated by this member?	Are S, L-child locks propagated by this member?	Are U, L-child locks propagated by this member?
IX	IX	Yes	Yes	Yes
IX, SIX	None	No	No	No

The following table shows the conditions that cause the child locks to be propagated when locking protocol 2 is used.

Table 28. Determining when child locks are propagated to XES when locking protocol 2 is used

Maximum page set P-lock mode of this member is ...	And the maximum page set P-lock mode of the other members is...	Are X, L-child locks propagated by this member?	Are S, L-child locks propagated by this member?	Are U, L-child locks propagated by this member?
IS, S	None, IS, S	N/A	No	Yes
X	None	No	No	No
IS	IX, SIX	N/A	Yes	Yes
IX, SIX	IS	Yes	No	Yes
IX	IX	Yes	Yes	Yes
SIX	None	Yes	No	No
IX	None	No	No	No

**Notes:**

- Some child L-locks might be acquired before the page set P-lock is obtained. When this happens, child L-locks are automatically propagated.
- When a page set switches from inter-system read/write interest to no inter-system read/write interest, a short period of time exists when the page set remains GBP-dependent, before the P-lock reverts to X state. During this time, child L-locks continue to be propagated.

**Related concepts**

“A locking scenario”

**Relationship between L-locks and P-locks:**

L-locks and P-locks are managed independently of one another.

Inter-DB2 interest is possible on an L-lock but not on a P-lock that is held on the same object. Inter-DB2 interest is also possible on the P-lock but not on the L-lock. The inter-DB2 interest on the page set P-lock, in locking protocol 2, controls both GBP-dependency and child lock propagation.

**A locking scenario**

Looking at an example of locking activity between two members of a data sharing group can help you to understand locking in a data sharing environment.

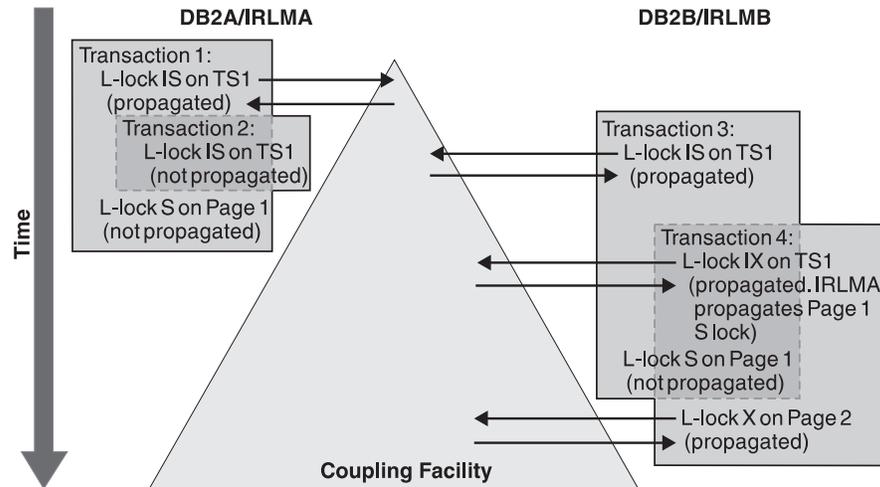


Figure 41. A lock propagation scenario

In the figure above:

1. The L-lock on table space 1 (TS1), which is associated with transaction 2, is not propagated because DB2A already holds a lock of an equal restriction on that object. (The L-lock on TS1, which is associated with transaction 3, is propagated because that lock is from another member.)
2. The child L-lock on page 1, which is associated with transaction 1, is not propagated at the time that it is requested because its parent lock is IS on both members: no inter-DB2 read/write interest exists on the parent.
3. When Transaction 4 upgrades its lock to IX on TS1, its X lock on Page 2 must be propagated because inter-DB2 read/write interest now exists on the parent. Also, the child lock on Page 1, which is associated with transaction 1, must be propagated.

The child lock is propagated under an IRLM SRB, not under the transaction's TCB. This propagation is counted in the statistics report as an asynchronous propagation, as shown in **B** in the DB2 OMEGAMON statistics trace.

#### Related reference

“Using the DB2 statistics trace” on page 172

#### Traces that indicate whether locks have been propagated

The statistics and accounting traces indicate the number of global locks that have been propagated to XES.

The ratio of this number to the total number of global locks requested reflects the effects of explicit hierarchical locking and other locking optimizations.

#### Related tasks

“Measuring transaction locking optimizations” on page 175

## Tuning your use of locks

Most recommendations for reducing lock contention and locking costs in a single system hold true when sharing data.

**Recommendations:** To reduce locking contention, use the tuning actions that are in place for single-DB2 processing. Here is a summary:

- Use partitioned table spaces.

- Use page locking.  
Although row locking can increase concurrency in some cases, you must weigh the benefits of increased concurrency against the increase in locking overhead that row locking might incur. (The amount of overhead depends on how well your application can avoid taking locks.)

One way to achieve the concurrency of row locking and avoid the additional data sharing lock overhead is to define the table space with MAXROWS 1.

- Use an ISOLATION level of uncommitted read (UR), if your applications can tolerate reading uncommitted data.
- Take advantage of lock avoidance whenever possible, if you cannot use ISOLATION(UR). You do this by binding with an isolation level of cursor stability (CS) and CURRENTDATA(NO). These are **not** the defaults.
- Reduce the scope of BIND operations by using packages. This reduces DB2 catalog and directory contention.
- Design for thread reuse and choose the RELEASE option carefully.

Using locking protocol 2 can reduce false lock contention and XES lock contention for parent L-locks when you request lock modes IS or IX.

If you use the BIND option RELEASE(DEALLOCATE) for objects that do not have a lot of concurrent activity within a member, you can avoid the cost of releasing and reacquiring the same parent locks again and again. You can also reduce the amount of false contention for those transactions that use the thread.

To achieve a good balance between storage and processor usage, use the bind option RELEASE(DEALLOCATE) for plans or packages that are frequently used. To avoid increasing the EDM pool storage too much, use RELEASE(COMMIT) for plans or packages that are not used as frequently.

#### Related tasks

 [Choosing ACQUIRE and RELEASE options \(DB2 Performance\)](#)

 [Programming your applications for concurrency \(DB2 Performance\)](#)

### Avoiding false contention

The DB2 coupling facility lock structure has two parts: a lock table, which is used to determine whether inter-DB2 read/write interest exists on a particular resource, and a list of the update locks that are currently held.

When considering false contention, you must be concerned with the size of the lock table. The total size of the lock structure determines the size of the lock table. Assuming that you specify an INITSIZE value on the CFRM policy that is a power of 2, the lock table is allocated to one-half the total size of the lock structure. The value that you specify for the lock table entries (LTE) parameter in the IRLMPROC or with the MODIFY irlmproc,SET,LTE command can control how the lock structure is partitioned.

The number of members in the group determines the size of each entry in that lock table. IRLM uses the value that you specify in the LOCK ENTRY SIZE field of installation panel DSNTIPJ to determine the initial size of the lock table entries. IRLM also uses the value that you specify in the NUMBER OF LOCK ENTRIES field of installation panel DSNTIPJ to determine how the lock structure is initially partitioned.

IRLM assigns locked resources to an entry value in the lock table. This is how it can quickly check to see if a resource is already locked. If the lock structure is too small (thereby making the lock table portion too small), many locks can be

represented by a single value. Thus, “false” lock contention can exist. *False lock contention* is where two different locks on different resources hash to the same lock entry. The second lock requester is suspended until it is determined that no real lock contention exists on the resource.

False contention can be a problem for work loads that have heavy inter-DB2 read/write interest.

One way to reduce false contention is to increase the size of the lock structure or to increase the proportional size of the lock table.

#### **Related reference**

 LOCK ENTRY SIZE field (DB2 Installation and Migration)

#### **Monitoring for false contention:**

You can determine the amount of false contention by using the RMF Coupling Facility Activity reports.

DB2 also provides necessary information in its accounting and statistics trace classes. More detailed information can be found in the performance trace.

#### **Related tasks**

“Lock monitoring with the DB2 performance trace” on page 176

#### **Related reference**

“Lock monitoring with the coupling facility structure activity report” on page 171

“Lock monitoring with the DB2 statistics trace” on page 174

“Using the DB2 statistics trace” on page 172

#### **How much contention is acceptable:**

For the best performance, you want to achieve the least possible amount of global lock contention, both real and false.

*Global lock contention* includes false contention, XES contention, and IRLM contention. Aim for total global lock contention of less than 5%. When the global lock contention is less than 5%, the individual breakdown of false contention, XES contention, and IRLM contention is not significant.

Use the following guidelines to monitor your global lock contention.

- If the global lock contention is less than 3%, no action is necessary.
- If the global lock contention is 3 - 5%, no immediate action is necessary, but you should monitor your system.
- If the global lock contention is greater than 5%, identify the greatest contributor among false contention, XES contention, and IRLM contention, and tune your system accordingly.
- Ignore individual intervals with very low locking rates but very high false contention. This situation is not significant, and it often occurs on systems that use a one-minute statistics interval.

## Related concepts

“Ways to monitor DB2 locking activity” on page 170

### How to reduce false contention:

Several tips and recommendations can help you reduce false contention.

The following tips can help you reduce false contention:

- As much as possible, reduce the amount of real lock contention in your applications.
- Specify a larger size for the lock structure and manually rebuild it.
- Ensure that the value for LOCK ENTRY SIZE is not too large for the number of members in your group.

The LOCK ENTRY SIZE parameter for the first IRLM to join the group determines the size of each lock entry in the lock table.

A lock entry size of two allows twice as many lock entries as a lock entry size of four, as illustrated in the figure below.

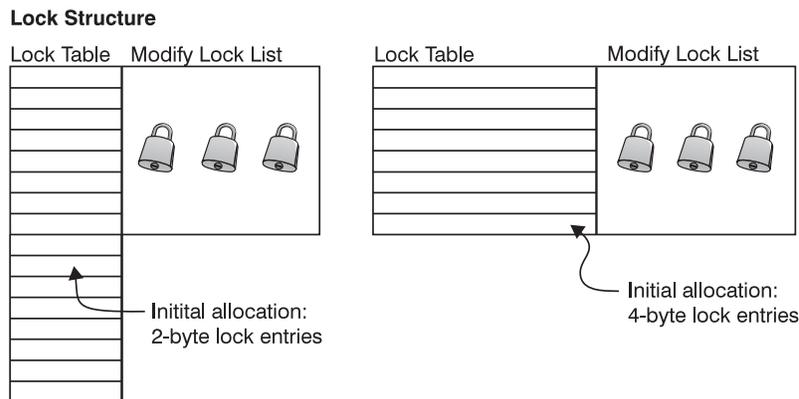


Figure 42. Initial lock entry size

IRLM automatically rebuilds the structure when it needs to. For example, if MAXUSRS=7, when the seventh member joins the group, IRLM automatically rebuilds the lock structure to create the 4-byte lock entries. This prepares the lock structure to handle an eighth member joining the group.

For this reason, even if you anticipate your group growing beyond seven members, you can start out with a lock entry size of 2 make the most efficient use of lock structure storage.

**Recommendation:** Set MAXUSRS for all IRLM instances to an appropriate value for the number of members that you expect to have in a data sharing group. For example, if you plan to have between 7 and 22 members in your data sharing group on a regular basis, you should set MAXUSRS=23 in the IRLMPROC procedure for the IRLM instance that is associated with each member of the data sharing group. Alternatively, you can set the LOCK ENTRY SIZE field value to 4 in installation panel DSNTIPJ when you install each member of the data sharing group. Doing this causes the DB2 installation process to set the MAXUSRS value to 23 in the associated IRLMPROC procedures. Lock structure allocation occurs when the first member joins the data sharing group after an IPL of the operating system, or after any situation that causes the lock structure to be deallocated.

If you initially set MAXUSRS=23 for all IRLM instances, the lock structure size is already adequate for up to 22 members, no matter which IRLM is the first to connect. A lock structure rebuild is not necessary when the seventh member joins the group.

If you increase the lock entry size (and thereby increase MAXUSRS), increase the lock structure size, to maintain the number of lock table entries and record list entries. If you do not increase the lock structure size, IRLM obtains the storage that it needs for the increased lock entry size from storage for lock table entries or record list entries.

- Using locking protocol 2 can reduce false lock contention for parent L-locks when you request lock modes IS or IX. Locking protocol 2 reduces overhead from data sharing processing but might increase child lock propagation to the lock structure in the coupling facility. Propagation is based on the page set P-lock instead of the parent L-lock.

If a data sharing group uses locking protocol 1, and you want to enable locking protocol 2, you must first quiesce that data sharing group. To activate locking protocol 2, you must initiate a group-wide shutdown. The restart of the first member after a group-wide shutdown changes the group protocol level from 1 to 2.

To determine whether locking protocol 2 is enabled, issue the DISPLAY GROUP command.

#### Related tasks

“Changing the size of the lock structure by rebuilding” on page 177

#### Decreasing lock entry size:

IRLM does not automatically rebuild if the number of members decreases.

To decrease the lock entry size, you must:

1. Quiesce all members of the group, using the following command:  
`-DB1A STOP DB2 MODE(QUIESCE)`
2. If IRLM is not automatically stopped along with DB2, enter the z/OS command:  
`STOP irlmproc`
3. Force all connections to the lock structure to disconnect by issuing the following z/OS command:  
`SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL`
4. Force the deallocation of the lock structure by issuing the following z/OS command:  
`SETXCF FORCE,STRUCTURE,STRNAME=strname`
5. Change the lock entry size for at least one IRLM. (You should change the value for all of them.)  

If you change the IRLM startup procedure directly, the parameter you change is called MAXUSRS. The value of LOCK ENTRY SIZE is translated during the DB2 installation or migration process. The value you enter on the parameter directly is not the same as the value you put in the LOCK ENTRY SIZE parameter of DSNTIPJ.
6. Start the member and IRLM that have the updated value. (You must start the updated member first.)
7. Start all other members.

A group restart occurs when you restart the members. Because you quiesced work before changing the lock entry size, the group restart should be relatively quick. Nonetheless, decreasing the lock entry size is a disruptive procedure. Consider doing this only in situations when the lock entry size is set too high for the number of members in the group, and you cannot alleviate the false contention within the storage boundaries you have.

**Related reference**

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

**How z/OS resolves contention problems**

When contention exists on a hash class, z/OS uses XCF messages to resolve the conflict.

This is how it determines which specific resources are involved in the contention, or if the contention is false. For speedy resolution of contention situations, ensure no queuing of messages exists for XCF message buffers. You can use the XCF Activity Report of RMF to detect this queuing.

**Related information**

➡ Tuning a Sysplex (z/OS MVS Setting Up a Sysplex)

**How partition locks affect concurrency**

In a partitioned table space, locks are obtained at the partition level. Individual partitions are locked as they are accessed. This locking behavior enables greater concurrency.

**PSPI** Partition locks are always acquired, even if the table space was defined in a version of DB2 prior to Version 8 or Version 9 with the LOCKPART NO clause. For table spaces that are defined with LOCKPART NO, DB2 no longer locks the entire table space with one lock when any partition of the table space is accessed.

Each locked partition is a separate parent lock. Therefore, DB2 and IRLM can detect when no inter-DB2 read/write interest exists on that partition and thus do not propagate child L-locks unnecessarily.

**Restrictions:** If any of the following conditions are true, DB2 cannot selectively lock the partitions. It must lock **all** the partitions:

- The table space is defined with LOCKSIZE TABLESPACE.
- LOCK TABLE IN EXCLUSIVE MODE is used (without the PART option).

The following figure shows partition locks. A partition lock is taken only when the partition is accessed.

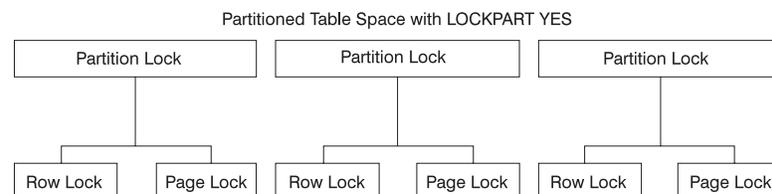


Figure 43. Partition locks

**The duration of a partition lock:** Partition locks follow the same rules as table space locks, and **all** partitions are held for the same duration. Thus, if one package

is using `RELEASE(COMMIT)` and another is using `RELEASE(DEALLOCATE)`, all partitions use `RELEASE(DEALLOCATE)`. A partition lock can be held past commit point if it uses `CURSOR WITH HOLD`.

**The mode of a partition lock:** Partition locks have the same possible states as table space locks (IS, IX, S, SIX, and X).

**Lock escalation:** Lock escalation occurs when the number of locks per table space exceeds the threshold specified at installation or on the `LOCKMAX` clause of `CREATE` or `ALTER TABLESPACE`. Lock counts are not kept on a partition basis. When the maximum number of locks for a table space is reached, the locks on all partitions are escalated to S or X. Partitions that have not yet been locked are not affected by lock escalation; they remain unlocked. Any partitions already holding S or X locks remain unchanged.

After lock escalation occurs, any unlocked partitions that are subsequently accessed use a gross lock.

**Monitoring selective partition locking:** The performance trace and the `DISPLAY DATABASE` command give information about selective partition locking.

Use performance trace class 6 (IFCID 0020) to determine whether you have taken advantage of selective partition locking for your partitioned table spaces. If you are using `OMEGAMON`, you can see this in the locking summary portion of the SQL

Activity Report. 

#### Related tasks

 Locking a table explicitly (DB2 Performance)

## Deadlock detection and resource timeouts in data sharing environments

The deadlock detection intervals and resource timeout value of your data sharing environment should be based on how deadlock detection and resource timeouts work in data sharing environments.

**Recommendations:** Quick detection of deadlocks and timeouts is necessary in a data sharing environment to prevent a large number of waiters on each system. A large numbers of waiters can cause much longer wait times for timeouts and deadlocks. The following are two recommendations to help prevent a large number of waiters from developing on each system:

- If your non-data-sharing DB2 subsystem has a problem with deadlocks, consider reducing the deadlock time to prevent a long lists of waiters from developing. (If you do not have a problem with deadlocks, you most likely will not have to change any parameters for data sharing.)
- If you have stringent timeout limits that must be honored by DB2, consider decreasing the deadlock time before moving to data sharing, as illustrated in this example:

Assume that you have set the timeout period for your non-data-sharing DB2 subsystem to 55 seconds because you want the wait time for timeout to be at or before 60 seconds. (This assumes that your deadlock time value is five.) In a data sharing environment, reduce the timeout period to 40 seconds. This makes it more likely that your actual wait time for timeouts is at or before 60 seconds.

## Global deadlock processing

In a data sharing environment, deadlocks can occur between transactions on different members.

The term *global deadlock* refers to the situation where two or more members are involved in the deadlock. *Local deadlock* refers to the situation where all of the deadlocked transactions reside on a single member.

### Controlling deadlock detection:

Use the DEADLOCK parameter in the IRLM startup procedure to control how often IRLM does its deadlock detection processing.

Specify the parameter as follows:

DEADLOCK='x,y'

- x        The number of seconds between two successive scans for a local deadlock (DEADLOCK TIME value on installation panel DSNTIPJ). The default is 1 second. Values can range from 1 to 5 seconds, or 100 to 5000 milliseconds.
- y        The number of local scans that occur before a scan for global deadlock starts (DEADLOCK CYCLE value on install panel DSNTIPJ). IRLM always uses a value of 1.

Global deadlock detection requires the participation of all IRLM members in the data sharing group. Each IRLM member has detailed information about the locks that are held and are being waited for by the transactions on its associated DB2 member. However, to provide global detection and timeout services, each IRLM is informed of all requests that are waiting globally so that the IRLM can provide information about its own blockers. That IRLM also provides information about its own waiters. The IRLM members use XCF messages to exchange information so that each member has this global information.

### The global deadlock manager:

To coordinate the exchange of information, one IRLM member assumes the role of the global deadlock manager.

As IRLM members join or leave the group, the global deadlock manager might change.

### The local deadlock detector:

Each IRLM member in the group must participate in the global deadlock detection process.

Each IRLM member (including the one designated as the global deadlock manager) has the role of local deadlock detector.

### Relationship between local and global deadlock detection:

The four XCF messages represent one global detection cycle, which usually takes two to four *x*-second intervals to complete (where *x* is the number of local cycles).

Four XCF messages are required to gather and communicate the latest information from the local deadlock detectors:

1. The local deadlock detector sends its information about lock waiters to the global deadlock manager.
2. The global deadlock manager takes that information from all local deadlock detectors and sends messages to each of the IRLMs in the group. (Because the global deadlock manager is also a local deadlock detector, it receives the same information, although somewhat quicker than the rest of the IRLMs.)
3. Each local deadlock detector checks the global view of resources and determines if it has blockers for other waiters. It passes that information along to the global deadlock manager with its list of waiters.
4. The global deadlock manager, from the information it receives from the local deadlock detectors, determines if a global deadlock or timeout situation exists. If a global deadlock situation exists, DB2 chooses a candidate for the deadlock. The global deadlock manager also determines if any timeout candidate is blocked by an incompatible waiter or holder and, if so, presents that candidate to the owning IRLM, along with any deadlock candidates belonging to that IRLM. When DB2 receives this information, it determines if it should request that IRLM reject any given timeout candidate waiter.

The following figure illustrates an example in which the deadlock time value is set to 5 seconds.

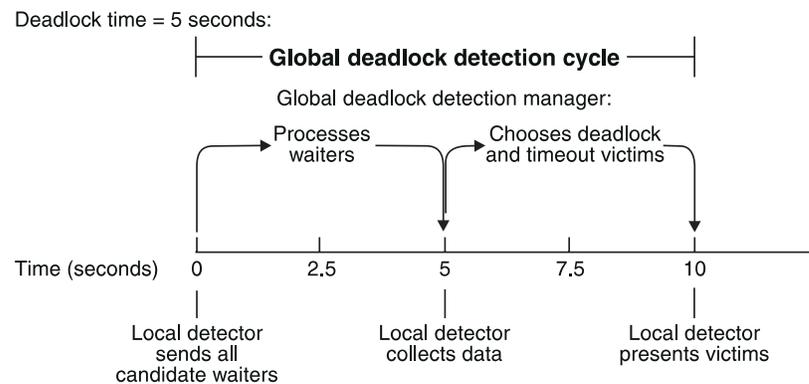


Figure 44. Global deadlock detection cycle

Deadlock detection might be delayed if any of the IRLMs in the group encounter any of the following conditions:

- XCF signaling delays
- IRLM latch contention (can be encountered in systems with extremely high IRLM locking activity)
- A large number of global waiters

### Global timeout processing

Just as in a non-data-sharing environment, DB2 calculates the timeout period based on the RESOURCE TIMEOUT and DEADLOCK TIME installation parameter values.

DB2 calculates the timeout period as follows:

1. Divide RESOURCE TIMEOUT by DEADLOCK TIME
2. Round to the next largest integer
3. Multiply that integer by DEADLOCK TIME

In non-data-sharing systems, the actual time that a transaction waits on a lock before timing out varies between the timeout period and the timeout period plus one DEADLOCK TIME interval.

For example, if the timeout period for a given transaction is 60 seconds and the DEADLOCK TIME value is 5 seconds, the transaction waits between 60 and 65 seconds before timing out, with the average wait time of 62.5 seconds. This is because timeout is driven by the deadlock detection process, which is activated on a timer interval basis.

#### **Elapsed time until timeout, non-data-sharing:**

The timeout period for a local timeout is typically shorter than that of a global timeout.

The actual time a process waits until timing out usually falls within the following range:

```
MIN LOCAL TIMEOUT = timeout period
MAX LOCAL TIMEOUT = timeout period + DEADLOCK TIME value
AVERAGE LOCAL TIMEOUT = timeout period + DEADLOCK TIME value/2
```

However, the maximum or average values can be larger, depending on the number of waiters in the system or if a heavy IRLM workload exists.

#### **Elapsed time until timeout, data sharing:**

In a data sharing environment, because the deadlock detection process sends inter-system XCF messages, a given transaction typically waits somewhat longer before timing out than in a non-data-sharing environment.

How much longer a transaction waits depends on where in the global deadlock detection cycle that the timeout period actually expired. However, the length of time a process waits until timing out generally falls within the following range:

```
MIN GLOBAL TIMEOUT = timeout period + DEADLOCK TIME value
MAX GLOBAL TIMEOUT = timeout period + 4 * DEADLOCK TIME value
AVERAGE GLOBAL TIMEOUT = timeout period + 2 * DEADLOCK TIME value
```

Again, the maximum or average values might be larger.

## **Ways to monitor DB2 locking activity**

With data sharing, it is essential to control the volume of global lock requests that are propagated to the coupling facility and to control the amount of lock contention, both real and false.

You must monitor both the amount and type of locking that your applications are doing, and you must also make sure that any locking problems are not caused by data sharing resources, such as an undersized lock structure, or the overuse of the coupling facility or coupling facility channels (links).

The z/OS command D XCF,STRNAME can also be used to monitor lock structure activity.

### Related reference

“Displaying information about specific structures” on page 85

### Lock monitoring with the DISPLAY DATABASE command

Use the LOCKS ONLY option on DISPLAY DATABASE to display information about page set, partition, or table locks that are held on resources.

The “lock” column of the display describes the type and duration of locks used by corresponding agents.

#### GUPI

The following figure is an example of output of DISPLAY DATABASE for a table space. The application identified as LSS001 on member DB1A has locked partitions 1 and 2. LSS002 on member DB2A has locked partitions 1 and 3. Partition 4, which has no locks, is not on the display because the ONLY option of DISPLAY DATABASE was used.

NAME	TYPE	PART	STATUS	CONNID	CORRID	LOCKINFO
TSPART	TS	01	RO	LSS001	DSN2SQL	H-IS,P,C
-			MEMBER NAME	DB1A		
TSPART	TS	01	RO			H-S,PP,I
-			MEMBER NAME	DB1A		
TSPART	TS	01	RO	LSS002	DSN2SQL	H-IS,P,C
-			MEMBER NAME	DB2A		
TSPART	TS	01	RO			H-S,PP,I
-			MEMBER NAME	DB2A		
TSPART	TS	02	RW	LSS001	DSN2SQL	H-IS,P,C
-			MEMBER NAME	DB1A		
TSPART	TS	02	RW			H-S,PP,I
-			MEMBER NAME	DB1A		
TSPART	TS	03	RW	LSS002	DSN2SQL	H-IS,P,C
-			MEMBER NAME	DB2A		
TSPART	TS	03	RW			H-S,PP,I
-			MEMBER NAME	DB2A		

Figure 45. Example DISPLAY DATABASE LOCKS for a table space

#### GUPI

### Lock monitoring with the coupling facility structure activity report

The Coupling Facility Activity Report of RMF describes activity to all structures in the coupling facility for a given time period.

The following figure shows a partial report, giving information about:

- **A**: Total number of lock-related requests.
- **B**: Number of requests that were deferred because of contention.
- **C**: The number of deferred requests that were caused by false contention.

## COUPLING FACILITY STRUCTURE ACTIVITY

STRUCTURE NAME = DSNDBOA_LOCK1		TYPE = LOCK		----- DELAYED REQUESTS -----									
SYSTEM	TOTAL	#	% OF	-SERV TIME(MIC)-	REASON	#	% OF	---AVG TIME(MIC)---	STD_DEV	/ALL	EXTERNAL REQUEST		
NAME	AVG/SEC	REQ	ALL	AVG	STD_DEV	REQ	REQ	/DEL	STD_DEV	/ALL	CONTENTIONS		
STLABC2	126K	SYNC	126K	51.0%	61.0	22.5					REQ TOTAL <b>A</b> 162K		
	701.7	ASYN	0	0.0%	0.0	0.0	NO SCH	0	0.0%	0.0	0.0	REQ Deferred 612	
		CHNGD	0	0.0%	INCLUDED IN ASYNC							-CONT <b>B</b> 621	
												-FALSE CONT <b>C</b> 212	

Figure 46. Partial RMF Coupling Facility Activity report for lock structure

### Calculating contention percentages:

You can use a formula to calculate the percentage of global contention and false contention in your data sharing environment.

Use the following calculations:

- Total contention is the number of deferred requests (**B**) divided by the total number of requests (**A**), multiplied by 100. So, for this example:

$$(621 / 162000) \times 100 = .387\%$$

This indicates that the global contention rate is approximately 0.39 % (a good figure).

- False contention is the number of false contentions (**C**) divided by the total number of requests (**A**) multiplied by 100. For this example:

$$(212 / 162000) \times 100 = 0.13\%$$

Thus, the rate of false contention is 0.13 % (a very good figure).

### Using the DB2 statistics trace

The DB2 statistics trace provides counters that track the amount of global locking activity and contention that each member in the data sharing group is encountering.

This trace runs with low overhead. Keep the DB2 statistics trace turned on to allow continuous monitoring of each subsystem.

See the following figure to view the type of information provided by a statistics trace.

DATA SHARING LOCKING	QUANTITY	/SECOND	/THREAD	/COMMIT
GLOBAL CONTENTION RATE (%)	3.24	<b>A</b>		
L-LOCKS XES RATE (%)	46.95			
LOCK REQUESTS (P-LOCKS)	147.8K	246.42	N/C	1.61
UNLOCK REQUESTS (P-LOCKS)	147.8K	246.38	N/C	1.61
CHANGE REQUESTS (P-LOCKS)	0.00	0.00	N/C	0.00
<b>B</b>				
SYNCH.XES - LOCK REQUESTS	779.1K	1298.60	N/C	8.47
SYNCH.XES - CHANGE REQUESTS	253.8K	423.05	N/C	2.76
SYNCH.XES - UNLOCK REQUESTS	922.9K	1538.25	N/C	10.04
ASYNCH.XES - RESOURCES	<b>C</b> 81690.00	136.16	N/C	0.89
SUSPENDS - IRLM GLOBAL CONT	<b>D</b> 1871.00	3.12	N/C	0.02
SUSPENDS - XES GLOBAL CONT.	<b>E</b> 59745.00	99.58	N/C	0.65
SUSPENDS - FALSE CONTENTION	<b>F</b> 3964.00	6.61	N/C	0.04
INCOMPATIBLE RETAINED LOCK	0.00	0.00	N/C	0.00
NOTIFY MESSAGES SENT	1646.00	2.74	N/C	0.02
NOTIFY MESSAGES RECEIVED	2359.00	3.93	N/C	0.03
P-LOCK/NOTIFY EXITS ENGINES	500.00	N/A	N/A	N/A
P-LCK/NFY EX.ENGINE UNAVAIL	0.00	0.00	N/C	0.00
PSET/PART P-LCK NEGOTIATION	0.00	0.00	N/C	0.00
PAGE P-LOCK NEGOTIATION	1137.00	1.90	N/C	0.01
OTHER P-LOCK NEGOTIATION	0.00	0.00	N/C	0.00
P-LOCK CHANGE DURING NEG.	1137.00	1.90	N/C	0.01

Figure 47. Data sharing locking block of OMEGAMON statistics trace

The following table contains an explanation of the fields that are shown in the figure above.

Table 29. Explanation of the fields in a OMEGAMON statistics trace

Field	Explanation
<b>A</b>	The global contention rate.
<b>B</b>	These counters indicate the total number of lock, change, and unlock requests (including L-locks and P-locks) that were propagated to XES synchronously.
<b>C</b>	The number of resources (including L-locks and P-locks) that were propagated to XES asynchronously. DB2 uses the term <i>asynchronous</i> to mean that the request was done under a system execution unit, asynchronous to the allied work unit.  This particular counter can be incremented when, for example, one member has an IS lock on a particular table space and another member requests an IX lock. The S child locks held by the first member must be propagated under a system execution unit to XES and the coupling facility.  It is possible for these asynchronous child lock propagations to encounter false contention. If so, the false contention is counted in RMF statistics, but not in DB2.
<b>D</b>	The number of real contentions, as detected by IRLM.
<b>E</b>	The number of real contentions, as detected by XES, that were not IRLM-level contentions. IRLM has knowledge of more lock types than XES. Thus, IRLM often resolves contention that XES cannot. The most common example of XES-level contention is usually the intent locks (IS and IX) on the parent L-locks. IS and IX are compatible to IRLM but not to XES. Another common example is the U and S page L-locks; U and S are compatible to IRLM, but not to XES.
<b>F</b>	The number of false contentions.

## Calculating global contention percentages:

You can use the DB2 statistics trace to calculate global contention percentages for synchronous lock requests.

Use the statistics trace shown in the following figure to calculate the global contention percentages. Figure 47 on page 173

DATA SHARING LOCKING	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----
GLOBAL CONTENTION RATE (%)	3.24	<b>A</b>		
L-LOCKS XES RATE (%)	46.95			
LOCK REQUESTS (P-LOCKS)	147.8K	246.42	N/C	1.61
UNLOCK REQUESTS (P-LOCKS)	147.8K	246.38	N/C	1.61
CHANGE REQUESTS (P-LOCKS)	0.00	0.00	N/C	0.00
<b>B</b>				
SYNCH.XES - LOCK REQUESTS	779.1K	1298.60	N/C	8.47
SYNCH.XES - CHANGE REQUESTS	253.8K	423.05	N/C	2.76
SYNCH.XES - UNLOCK REQUESTS	922.9K	1538.25	N/C	10.04
ASYNCH.XES - RESOURCES	<b>C</b> 81690.00	136.16	N/C	0.89
SUSPENDS - IRLM GLOBAL CONT	<b>D</b> 1871.00	3.12	N/C	0.02
SUSPENDS - XES GLOBAL CONT.	<b>E</b> 59745.00	99.58	N/C	0.65
SUSPENDS - FALSE CONTENTION	<b>F</b> 3964.00	6.61	N/C	0.04
INCOMPATIBLE RETAINED LOCK	0.00	0.00	N/C	0.00
NOTIFY MESSAGES SENT	1646.00	2.74	N/C	0.02
NOTIFY MESSAGES RECEIVED	2359.00	3.93	N/C	0.03
P-LOCK/NOTIFY EXITS ENGINES	500.00	N/A	N/A	N/A
P-LCK/NFY EX.ENGINE UNAVAIL	0.00	0.00	N/C	0.00
PSET/PART P-LCK NEGOTIATION	0.00	0.00	N/C	0.00
PAGE P-LOCK NEGOTIATION	1137.00	1.90	N/C	0.01
OTHER P-LOCK NEGOTIATION	0.00	0.00	N/C	0.00
P-LOCK CHANGE DURING NEG.	1137.00	1.90	N/C	0.01

Figure 48. Data sharing locking block of OMEGAMON statistics trace

These calculations only account for synchronous lock requests.

- Total contention is the total number of suspends because of contention (**D** + **E** + **F**) divided by the total number of requests that went to XES (excluding asynchronous requests) ((three fields under **B**) + **D** + **E** + **F**) multiplied by 100. For this example:

$$(65580 / 2021380) \times 100 = 3.24\%$$

This indicates that the contention rate is approximately 3.24 percent (**A**).

Because this is such a low rate of contention, it is probably not necessary to determine the amount of false contention. However, knowing how to calculate false contention can be useful.

- False contention is the number of false contentions (**F**) divided by the total number of requests that went to XES (excluding asynchronous requests) ((three fields under **B**) + (**D** + **E** + **F**)) multiplied by 100. For this example:

$$3964 / 2021380 \times 100 = .20\%$$

Thus, the approximate rate of false contention is .2 percent, a very low figure.

## Lock monitoring with the DB2 statistics trace

Use the DB2 accounting trace to determine which users or plans are experiencing global lock contention.

The accounting trace provides a summary of thread resource usage within DB2. DB2 threads experiencing global lock contention are shown in accounting trace class 1, as shown in the figure below. The accumulated elapsed time of the suspensions are shown in accounting trace class 3.

LOCKING	AVERAGE	TOTAL	DATA SHARING	AVERAGE	TOTAL
TIMEOUTS	0.00	0	GLOBAL CONT RATE(%)	13.12	N/A
DEADLOCKS	0.00	0	L-LOCKS XES RATE(%)	77.68	N/A
ESCAL. (SHARED)	0.00	0	LOCK REQ - PLOCKS	0.33	<b>B</b> 9
ESCAL. (EXCLUS)	0.00	0	UNLOCK REQ - PLOCKS	0.33	9
MAX PG/ROW LOCKS HELD	4.00	4	CHANGE REQ - PLOCKS	0.00	0
LOCK REQUEST	12.11	<b>A</b> 327	LOCK REQ - XES	9.74	<b>C</b> 263
UNLOCK REQUEST	5.00	135	UNLOCK REQ - XES	4.33	<b>D</b> 117
QUERY REQUEST	0.00	0	CHANGE REQ - XES	2.85	<b>E</b> 77
CHANGE REQUEST	3.93	106	SUSPENDS - IRLM	0.00	0
OTHER REQUEST	0.00	0	SUSPENDS - XES	2.44	66
LOCK SUSPENSIONS	0.00	0	SUSPENDS - FALSE	0.11	<b>F</b> 3
IRLM LATCH SUSPENSIONS	0.07	2	INCOMPATIBLE LOCKS	0.00	0
OTHER SUSPENSIONS	0.00	0	NOTIFY MSGS SENT	0.00	0
TOTAL SUSPENSIONS	0.07	2			

Figure 49. Portion of OMEGAMON accounting trace showing locking activity

### Calculating false contention percentages:

You can use an accounting trace calculate false contention percentages.

Use the accounting trace as shown in the following figure to calculate the false contention percentages. Figure 49

False contention is the total number of suspends due to false contention (**F**)

LOCKING	AVERAGE	TOTAL	DATA SHARING	AVERAGE	TOTAL
TIMEOUTS	0.00	0	GLOBAL CONT RATE(%)	13.12	N/A
DEADLOCKS	0.00	0	L-LOCKS XES RATE(%)	77.68	N/A
ESCAL. (SHARED)	0.00	0	LOCK REQ - PLOCKS	0.33	<b>B</b> 9
ESCAL. (EXCLUS)	0.00	0	UNLOCK REQ - PLOCKS	0.33	9
MAX PG/ROW LOCKS HELD	4.00	4	CHANGE REQ - PLOCKS	0.00	0
LOCK REQUEST	12.11	<b>A</b> 327	LOCK REQ - XES	9.74	<b>C</b> 263
UNLOCK REQUEST	5.00	135	UNLOCK REQ - XES	4.33	<b>D</b> 117
QUERY REQUEST	0.00	0	CHANGE REQ - XES	2.85	<b>E</b> 77
CHANGE REQUEST	3.93	106	SUSPENDS - IRLM	0.00	0
OTHER REQUEST	0.00	0	SUSPENDS - XES	2.44	66
LOCK SUSPENSIONS	0.00	0	SUSPENDS - FALSE	0.11	<b>F</b> 3
IRLM LATCH SUSPENSIONS	0.07	2	INCOMPATIBLE LOCKS	0.00	0
OTHER SUSPENSIONS	0.00	0	NOTIFY MSGS SENT	0.00	0
TOTAL SUSPENSIONS	0.07	2			

Figure 50. Portion of OMEGAMON accounting trace showing locking activity

divided by the total number of requests that went to XES (**C** + **D** + **E**), multiplied by 100.

So, for this example:

$$(3 / 457) \times 100 = .66\%$$

False contention is approximately 0.66% of the total number of requests that went to XES.

### Measuring transaction locking optimizations:

You can measure transaction locking optimizations to see how well global transaction locking optimizations are working in your application.

To measure transaction locking optimizations:

1. Determine the total number of L-lock LOCK requests propagated to XES (X in the following example) by subtracting P-lock LOCK requests from the total number of LOCK requests propagated to XES:

$$\mathbf{C} - \mathbf{B} = \mathbf{X}$$

In this example:

$$263 - 9 = 254$$

2. Divide the total number of L-lock LOCK requests that were propagated to XES (X) by the total number of LOCK requests (**A**).

In this example:

$$254 / 327 = .77$$

3. Multiply by 100.

In this example:

$$.77 \times 100 = 77\%$$

For this particular work load, with 100% of the data of inter-DB2 read/write interest, 23% of LOCK requests were not propagated to XES and the coupling facility because of locking optimizations.

### **Lock monitoring with the DB2 performance trace**

The DB2 performance trace gives more detailed information about which shared resources are experiencing contention.

Performance traces are generally activated on an as needed basis because of their added overhead. Performance trace class 6 (specifically, IFCID 0045) indicates whether the suspension is because of contention.

This trace causes DB2 to write a trace record every time a lock is suspended and every time it is resumed. Part of the data that is recorded is the resource name that is experiencing the contention. By determining which shared resources are experiencing the lock contention, you might be able to make some design changes or other tuning actions to increase concurrency. Check for contention within IRLM, because IRLM indicates true contention over resources.

## **Changing the size of the lock structure**

You can change the size of a lock structure dynamically or by rebuilding the structure after making a CFRM policy change.

When choosing which way to change the size, consider the level of the coupling facility and when you want the storage of the lock table portion of the lock structure changed.

When you change the size of the lock structure dynamically, only the modify lock list portion of the structure is changed immediately. The size of the lock table portion remains unchanged until the structure is rebuilt. When the structure is rebuilt, the structure is reallocated at the changed size with the storage divided between the lock table and the record table based on your IRLMPROC LTE value or the value that is set using the MODIFY irlmproc,SET,LTE= command, if either of

these values is nonzero. (If the structure was forced with the SETXCF FORCE command, it is reallocated at the INITSIZE that is specified in the CFRM policy and not the changed size.)

### Related concepts

 Coupling facility structure size allocation (DB2 Installation and Migration)

## Changing the lock structure size dynamically

If certain conditions are met, you can dynamically change the size of a lock structure.

If **all** of the following conditions are true, you can use the SETXCF START,ALTER command to change the lock structure size:

- The lock structure is allocated in a coupling facility with CFLEVEL greater than zero.
- The currently allocated size of the structure is less than the maximum size that is defined in the SIZE parameter of the CFRM policy.

Enter the following command to change the lock structure size:

```
SETXCF START,ALTER,STRNAME=DSNDB0A_LOCK1,SIZE=news size
```

This example assumes that *news size* is less than or equal to the maximum size defined in the CFRM policy for the lock structure and that the group name is DSNDB0A. If the maximum size (SIZE in the CFRM policy) is still not big enough, you must increase the lock storage in the CFRM policy and rebuild the lock structure.

**Important:** For a duplexed lock structure, you are changing the size of both the primary and secondary structure with a single command.

Because the dynamic method affects only the record table entries portion of the lock structure, the impact of changing the lock size can have a disproportionately large effect on the record table list portion of the structure. For example, if you halve the size of the lock structure, it can result in all of the available record table entries being taken away—probably not the result you want. For the same reason, if you double the lock structure size, the increased storage space is used entirely by the record table unless a rebuild occurs or the group is shut down, the structure is forced, and the group is restarted. If either of these are done after the size is changed, the split of the new structure is determined by the number of lock table entries requested by the first IRLM to connect to the group.

## Changing the size of the lock structure by rebuilding

One way to change the size of a lock structure is to rebuild the structure.

If **any** of the following conditions are true, you must rebuild the lock structure in order to change its size:

- The lock structure is allocated in a coupling facility at CFLEVEL=0.
- The allocated size of the structure is already at the maximum size defined by the SIZE parameter of the CFRM policy, and you need to increase the maximum limit.
- You want to change the size of the lock table portion of the lock structure.

You must do at least one of the following procedures to rebuild the lock structure with a new size:

1. If you are at maximum size or want to increase your maximum size available, change the CFRM POLICY SIZE to the desired size. If you are satisfied with the number of lock table entries that you have been getting, leave the INITSIZE the same. If you want more lock table entries to decrease your contention rate, change the INITSIZE to accommodate the added number of entries, by doing either of the following:
  - a. If you are allowing IRLM to determine how many LTE entries to request, ensure that when you change INITSIZE, it remains an even power of 2 so that there is a 1:1 split between the lock table storage and the record table storage. For example, if your current INITSIZE is 16 MB, increase it to 32 MB.
  - b. If you are controlling the number of lock table entries by specifying LTE= in the IRLMPROC or by issuing the MODIFY irlmproc,SET,LTE= command, your INITSIZE does not need to be a power of 2, but it is still recommended. You do need to ensure that it is large enough to accommodate the storage required for your LTE= value and still large enough to create sufficient record table entries to handle your update volume.
2. If you want to change the size of the lock table:
  - a. If your contention rates are low and you want **fewer** lock table entries, do the following procedure:
    - 1) If you are letting IRLM control the number of lock entries to request, you must decrease the INITSIZE by a power of 2.

**Important:** This also decreases the record table size by half.

- 2) If you want to control the number of lock table entries to request, issue the MODIFY irlmproc,SET,LTE= command, where the LTE= value reduces the current number of lock entries by half. For example, assume that you currently have 16 MB lock entries, specify:

```
MODIFY irlmproc,SET,LTE=8
```

This method increases the number of record table entries, because the INITSIZE is not altered.

- b. If your contention rates are high and you want **more** lock table entries, do the following procedure:
  - 1) If you are letting IRLM control the number of lock entries to request, you should increase the INITSIZE by a power of 2. This also doubles the size of the record table.
  - 2) If you want to control the number of lock table entries to request, issue the MODIFY irlmproc,SET,LTE= command, where the LTE= value increases the current number of lock entries by powers of 2. For example, assume that you currently have 8 MB of lock entries, specify:

```
MODIFY irlmproc,SET,LTE=16
```

**Important:** This method decreases the number of record table entries if the INITSIZE is not altered.

If the INITSIZE is not large enough to provide the lock table storage and adequate record table storage, you will experience failures when trying to write RLE to the coupling facility. If you do not want the record table size to change, you must increase the INITSIZE by the same amount of storage that will be used by the additional lock table entries. For example, assume that you currently have 8 MB of lock table entries and you issue the following command:

```
MODIFY ir1mproc,SET,LTE=16
```

With an INITSIZE of 32 MB, you will not have any storage left for record table entries. Determine the additional storage needed by using the following formula:

$$\text{lock table entries (new) - lock table entries (old) x 2byte = additional storage for lock entries}$$

In this example:

$$16\text{MB} - 8\text{MB} \times 2\text{byte} = 16\text{MB}$$

So adding 16 MB to the original INITSIZE of 32 MB will result in the new INITSIZE of 48 MB.

**Important:** For a duplexed lock structure, you are changing the size of both the primary and secondary structure with a single command.

---

## Tuning group buffer pools

With DB2 data sharing, group buffer pools are a key component to ensuring that DB2 does not read down-level data in its member pools. Physical locks (P-locks) are also used in that process. Your understanding of these processes is helpful when tuning the data sharing group for best performance.

DB2 does the following in a data sharing environment:

- Ensures that DB2 does not read down-level data that is cached in its member buffer pools (*cache coherency*).
- Enables, as much as possible, a quick refresh of a down-level page without having to go to disk.

With DB2 data sharing, a database page can reside:

- In a local buffer pool
- In a group buffer pool
- On disk

Database pages continue to be cached in each member's buffer pools before they can be referenced or updated. Each sharing member can control its own buffer pool configurations (the size and number of buffer pools). However, if inter-DB2 read/write interest exists in the data, the group buffer pool is also used for caching data (unless the group buffer pool is defined as GBPCACHE (NO) or the page set is defined with GBPCACHE NONE).

The group buffer pool contains information necessary for maintaining cache coherency. Pages of GBP-dependent page sets are registered in the group buffer pool. When a changed page is written to the group buffer pool, all members that have this page cached in their buffer pools are notified that the page has been invalidated (this notification does not cause a processing interruption on those systems). This is called *cross-invalidation*. When a member needs a page of data and finds it in its buffer pool, it tests to see if the buffer contents are still valid. If not, then the page must be refreshed, either from the group buffer pool or disk.

## Assigning page sets to group buffer pools

Any data sharing group can have up to fifty 4 KB page size group buffer pools. Each group can have up to ten each of 8 KB, 16 KB, and 32 KB page size group buffer pools.

Different group buffer pools can reside in different coupling facilities. The strict naming convention you must use ensures that DB2 can map the group buffer pools to the individual member buffer pools. For example, buffer pool BP0 maps to group buffer pool GBP0. Thus, your choice of buffer pool determines which group buffer pool is used. GBP0 is the default unless you have specified a different default for user data and indexes on installation panel DSNTIP1.

For example, to assign table space DSN8S10D to GBP2, you must take the following actions:

1. Stop all access to the table space by issuing the following command:  

```
-DB1A STOP DATABASE(DSN8D10A) SPACENAM(DSN8S10D)
```
2. Change the buffer pool assignment by running the following SQL statement:  

```
ALTER TABLESPACE DSN8D10A.DSN8S10D  
  BUFFERPOOL BP2;
```
3. Allow access to the table space by issuing the following command:  

```
-DB1A START DATABASE(DSN8D10A) SPACENAM(DSN8S10D)
```

The above procedure works only when you are altering a table space to a buffer pool with the same page size.

### **Recommendations for performance**

For best performance, keep GBP-dependent page sets in separate buffer pools from non-GBP-dependent page sets.

For example, keep work file table spaces, which are always non-GBP-dependent, in different buffer pools than those used by GBP-dependent page sets. Assign work file table spaces to a buffer pool other than BP0. This separation helps DB2 more efficiently handle registering pages to, and unregistering pages from, the group buffer pool.

- Create a buffer pool specifically for LOBs to improve the efficiency of read LRSN value.
- Data rows that are too large for a single 4-KB page can use an 8-KB, 16-KB, or 32-KB page to improve the balance for different processing requirements. This allows you to store larger rows more efficiently and improve performance.

### **How to keep data from being shared**

It is possible, although not necessarily recommended, to restrict access to data to a single member.

If you choose to do this, consider the following operational issues:

- You cannot do workload balancing for that data, because the other members of the group are not aware of that data. Thus, the member that has access to the data can become overloaded if access to that data increases over time.
- Availability is compromised, because if the member that owns the data goes down, no other member can access that data.
- You might have to set up special affinities to allow the application access to that data. Work cannot be automatically routed around the group to find the data.

### **How to define private data:**

If you want access to a table space named NOSHARE limited only to DB2C, you could assign NOSHARE to a previously unused buffer pool, such as BP25, using the ALTER TABLESPACE statement.

Do not define a group buffer pool that corresponds to BP25, and assign BP25 a size of zero on any other member of the group. This prevents the other members of the group from attempting to use this buffer pool and therefore prevents the other members from accessing table space NOSHARE.

## Inter-DB2 interest and GBP-dependency

For DB2 data sharing environments, the concepts of inter-DB2 read/write interest and group buffer pool dependency (GBP-dependency) are closely related.

Whenever inter-DB2 read/write interest exists on a page set or partition, that object is GBP-dependent. Conversely, if no inter-DB2 read/write interest exists on a page set or partition, the object is usually not GBP-dependent. Sometimes an object still has pages cached in the group buffer pool, and it can remain GBP-dependent even after the inter-DB2 read/write interest has gone away. The following table shows you how to determine if a page set is GBP-dependent based on the inter-DB2 interest of one member and all other members.

*Table 30. Determining group buffer pool dependency*

One member's interest	Other members' interest	Is page set GBP-dependent?
Read-only	None, Read-only	No
Read-only	Read/Write	Yes
Read/Write	None	No <sup>1</sup>
Read/Write	Read-only	Yes
Read/Write	Read/Write	Yes

**Exception:**

1. The page set remains GBP-dependent for some time before DB2 removes the dependency. DB2 might not be able to remove the GBP-dependency if applications update the page set without issuing periodic commits.

## How DB2 tracks interest

The mechanism that DB2 uses to express interest in an object is a global lock called a *physical lock* (P-lock).

These locks are “physical” in contrast to transaction locks, sometimes called “logical locks” (L-locks). The level of P-lock that tracks DB2 read/write interest is a *page set P-lock*. There are also page P-locks, which serve another role.

Although you do not have as much control over physical locks as over transaction locks, P-locks play an important part in how DB2 tracks inter-DB2 interest.

Page set P-lock operations occur on each member, and reflect that member's level of interest in a page set. Even if only one data sharing member is active, page set P-lock operations still occur. The following table shows when those operations occur on each member.

*Table 31. When page set P-lock operations occur on each member*

Event	Page set P-lock operation
Page set or partition data sets are physically opened.	Page set P-lock is obtained in a read-only state.
Page set or partition is first updated.	Page set P-lock is changed to a read-write state.

Table 31. When page set P-lock operations occur on each member (continued)

Event	Page set P-lock operation
No update was done within an installation-specified time period or number of checkpoints (read-only switching).	Page set P-lock is changed to a read-only state.
Page set or partition data sets are closed.	Page set P-lock is released.

In addition to the events mentioned in the table above, a special case can occur under the following conditions:

- A single member with read/write interest and any number of members with read-only interest exist.
- All members have read-only interest and the page set or partition has been GBP-dependent since the time it was physically opened.

In those conditions, if the read-only members do not reference the page set again in a certain amount of time, DB2 physically closes the page set for the read-only members to remove the inter-DB2 read/write interest.

#### Related concepts

“Physical locks in data sharing” on page 186

“Scenarios of P-Lock operations”

### Scenarios of P-Lock operations

Looking at a scenario can help you understand how P-Lock operations work.

The following figure shows a typical sequence of events for P-locking and P-lock negotiations between two members of a data sharing group.

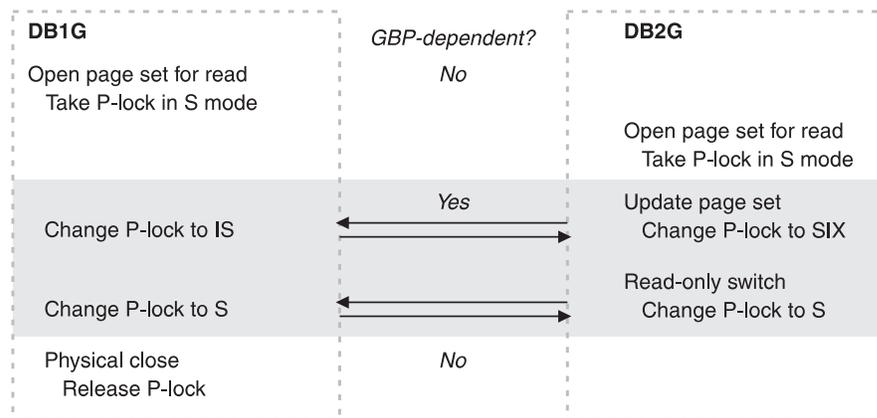


Figure 51. P-lock operations between two members. The arrows indicate that the members are negotiating P-lock modes.

The following figure shows what happens when a single updater remains.

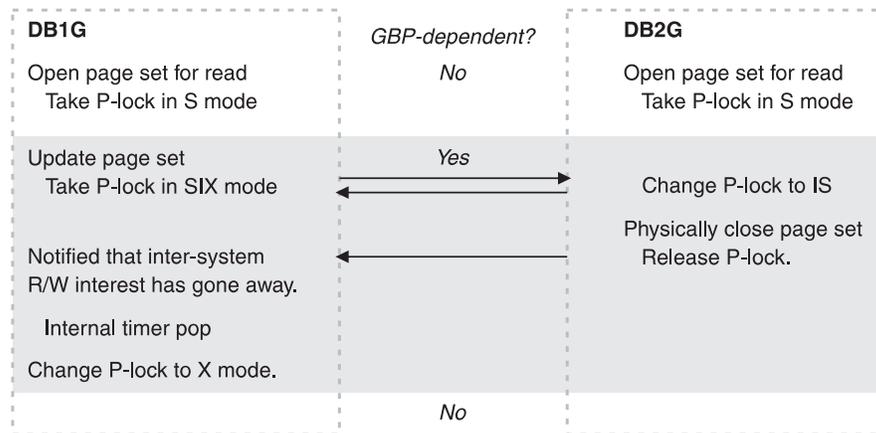


Figure 52. P-lock operations when single updater remains. When the reader physically closes the page set, the updater does not remove the GBP-dependency immediately.

### Tuning recommendation to prevent DB2 from frequently going in and out of GBP-dependency

To avoid the need for DB2 to frequently switch in and out of GBP-dependency, tune the subsystem parameters that affect when data sets are switched to a different state.

These parameters can be set through the following fields on installation panel DSNTIPL1:

- CHECKPOINT TYPE
- RECORDS/CHECKPOINT
- MINUTES/CHECKPOINT
- RO SWITCH CHKPTS
- RO SWITCH TIME

**Note:** NOT LOGGED table spaces change their GBP-dependency more often than LOGGED table spaces, because for NOT LOGGED table spaces, RO SWITCH CHKPTS and RO SWITCH TIME are both set to one. If you run multiple jobs consecutively that modify a NOT LOGGED table space, consider scheduling them to run within an RO SWITCH TIME of one minute, in order to reduce the number of times NOT LOGGED table spaces change their GBP-dependency.

### Related reference

 RECORDS/CHECKPOINT field (CHKFREQ and CHKLOGR subsystem parameters) (DB2 Installation and Migration)

 MINUTES/CHECKPOINT field (CHKFREQ and CHKMINS subsystem parameters) (DB2 Installation and Migration)

 CHECKPOINT TYPE field (CHKTYPE subsystem parameter) (DB2 Installation and Migration)

 RO SWITCH CHKPTS field (PCLOSEN subsystem parameter) (DB2 Installation and Migration)

 RO SWITCH TIME field (PCLOSET subsystem parameter) (DB2 Installation and Migration)

### Determining the amount of inter-system sharing

To determine the amount of inter-system sharing, you can use the DISPLAY BUFFERPOOL command with LIST option to get a snapshot of your objects and how they are being shared across members.

Using this command, you will see, by partition, which members have interest in the object, which level of interest they have (the P-lock state), and which member is the castout owner. You can also use the DBNAME, SPACENAME, GBPDEP, and CASTOWNR options to limit the scope of the report.

Issue the following command:

 GUIP

```
-DISPLAY BUFFERPOOL(BP0) LIST(*)
```

 GUIP

Your output will be similar to the output that is shown in the figure below.

```

DSNB401I : BUFFERPOOL NAME BPO, BUFFERPOOL ID 0, USE COUNT 21
DSNB402I : VIRTUAL BUFFERPOOL SIZE = 500 BUFFERS
          ALLOCATED      =      500  TO BE DELETED   =      0
          IN-USE/UPDATED =      0
DSNB406I : VIRTUAL BUFFERPOOL TYPE -
          CURRENT        = PRIMARY
          PENDING        = PRIMARY
          PAGE STEALING METHOD = LRU
DSNB404I : THRESHOLDS -
          VP SEQUENTIAL  = 80
          DEFERRED WRITE = 85  VERTICAL DEFERRED WRT = 80,0
          PARALLEL SEQUENTIAL = 50  ASSISTING PARALLEL SEQT= 0
DSNB460I :
-----PAGE SET/PARTITION LIST INFORMATION-----
-----DATA SHARING INFO-----
          TS GBP  MEMBER  CASTOUT  USE  P-LOCK
DATABASE SPACE NAME PART IX DEP  NAME  OWNER  COUNT STATE
=====
DSNDB01 DSNLLX02      IX  Y  DB1AA      0      0  IS
          DB1AB      Y      0  SIX
DSN8D61A DSN8S61E    001 TS  Y  DB1AA      0      0  IS
          DB1AB      Y      0  SIX
          002 TS  N  DB1AA      0      0  S
          DB1AB      0      0  S
DSNDB01 DSN8SPT01    IX  N  DB1AA      0      0  S
          DB1AB      0      0  S
          SPT01      TS  N  DB1AA      0      0  S
          DB1AB      0      0  S
DSNDB07 DSN4K01      TS  N  DB1AA      0      0  S
...
DSN9022I : DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION

```

Figure 53. Sample DISPLAY BUFFERPOOL output showing the amount of inter-system sharing.

## Displaying GBP-dependent page sets

You can use the DISPLAY DATABASE command to determine if a page set is GBP-dependent.

**GUPI** To determine if a particular page set is GBP-dependent, use the DISPLAY DATABASE command with the LOCKS option:

```
-DB1A DISPLAY DATABASE(DSN8D10A) SPACE(DSN8S10D) LOCKS
```

Your output will be similar to the output that is shown in the following figure:

NAME	TYPE	PART	STATUS	CONNID	CORRID	LOCKINFO
DSN8S10D	TS		RW			H-IX,PP,I
				MEMBER NAME	DB1A (CO)	
DSN8S10D	TS		RW			H-IX,PP,I
				MEMBER NAME	DB2A	

Figure 54. Sample DISPLAY DATABASE output showing GBP-dependent table spaces

Page set P-locks are identified by a member name rather than a correlation ID. They are also identified by 'PP' as the lock unit. If any of the P-locks shown in the output have a lock state of NSU, SIX, or IX, the identified page set is GBP-dependent. Thus, the output in the figure above shows that DSN8S10D is

GBP-dependent. **GUPI**

## Determining GBP-dependency for a particular member:

At times you might need to know what the impact is to bring down a particular member or to disconnect a particular member from the group buffer pool.

You can use the DISPLAY BUFFERPOOL command with the GBPDEP(Y) option to discover whether a particular member has any page sets opened that are GBP-dependent:

```
-DB1A DISPLAY BUFFERPOOL(BP0) GBPDEP(Y)
```

Your output will be similar to the output that is shown in the figure below.

```
@DIS BPOOL(BP0) GBPDEP(Y)
DSNB401I @ BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 21
DSNB402I @ VIRTUAL BUFFERPOOL SIZE = 500 BUFFERS
          ALLOCATED      =      500 TO BE DELETED    =      0
          IN-USE/UPDATED =      0
DSNB406I @ VIRTUAL BUFFERPOOL TYPE -
          CURRENT        = PRIMARY
          PENDING        = PRIMARY
          PAGE STEALING METHOD = LRU
DSNB404I @ THRESHOLDS -
          VP SEQUENTIAL  = 80
          DEFERRED WRITE = 85 VERTICAL DEFERRED WRT = 80,0
          PARALLEL SEQUENTIAL = 50 ASSISTING PARALLEL SEQ= 0
DSNB460I @
-----PAGE SET/PARTITION LIST INFORMATION-----
-----DATA SHARING INFO-----
          TS GBP  MEMBER CASTOUT USE P-LOCK
DATABASE SPACE NAME PART IX DEP  NAME  OWNER  COUNT STATE
=====
DSNDB01  DSNLLX02          IX Y  DB1AA      Y      0  IX
          DB1AB      0  IX
DSN8D71A DSN8S71E      001 TS Y  DB1AA      Y      0  IX
          DB1AB      0  IX
          003 TS Y  DB1AA      Y      0  IX
          DB1AB      0  IX
DSNDB01  DSNLLX01          IX Y  DB1AA      Y      0  IX
          DB1AB      0  IX
          SYSLGRNX      TS Y  DB1AA      Y      0  IX
          DB1AB      0  IX
DSN9022I @ DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION
```

Figure 55. Sample DISPLAY BUFFERPOOL output indicating page sets and partitions are group-buffer-pool-dependent

## Physical locks in data sharing

A physical lock (P-lock) is a type of page set lock or page lock that is specific to data sharing.

### Page set P-Locks

P-locks are used on physical objects stored in buffers (table spaces, index spaces, and partitions).

P-locks have complete partition independence: it is possible to have a P-lock on one partition of a page set and not on another. A P-lock on a page set does not necessarily mean that a P-lock exists on the corresponding index.

P-locks do not control concurrency but they do help DB2 track the level of interest in a particular page set or partition and determine the need for cache coherency controls.

P-locks differ from L-locks (transaction locks) in the following ways:

- P-locks are owned by a member. After a P-lock is obtained for the subsystem, later transactions accessing the locked data do not have to incur the expense of physical locking.
- The mode of a P-lock can be negotiated. If one member requests a P-lock that another member holds in an incompatible mode, the existing P-lock can be made less restrictive. The negotiation process usually involves registering pages or writing pages to the group buffer pool, and then downgrading the P-lock to a mode that is compatible with the new request.

## Retained P-locks

Just as with transaction locks, certain P-locks can be retained because of a system failure.

A retained P-lock means that other members cannot access the data that the P-lock is protecting if the accessing member requests a P-lock in an incompatible state. Thus, if a member fails holding an IX page set P-lock, it is still possible for another member to obtain an IX page set P-lock on the data.

Use the DISPLAY DATABASE command with the LOCKS option to determine if retained locks exist on a table space, index, or partition. An “R” in the LOCKINFO column indicates that a lock is retained.

The following table shows the possible modes of access for a page set and the P-lock state that is retained if the member that is represented in the first column fails.

Table 32. Determining retained P-lock state

One member's interest	Other members' interest	Retained P-lock states of single member
Read-only	None, Read-only	None
Read-only	Read/Write	None
Read/Write	None	X or NSU <sup>1</sup>
Read/Write	Read-only	IX <sup>2</sup>
Read/Write	Read/Write	IX

### Notes:

1. NSU stands for “non-shared update”. It acts like an X lock, but is only used during P-lock negotiation from an X to an SIX.
2. The P-lock is retained in SIX mode if the page set or partition is an index that is not on a DB2 catalog or directory table space.

## Related concepts

“Active and retained locks” on page 120

## Page P-locks

At times, a P-lock must be obtained on a page to preserve physical consistency of the data between members. These locks are known as *page P-locks*.

Page P-locks, are used, for example, when two subsystems attempt to update the same page of data and row locking is in effect. These locks are also used for GBP-dependent space map pages and GBP-dependent leaf pages for indexes, regardless of locking level. Page P-locks can also be retained if a member fails.

If an index page set or partition is GBP-dependent, DB2 does not use page P-locks for that index page set or partition if **all** of the following are true:

- Only one member is updating the index page set or partition
- No repeatable read claimers exist on the read-only members for the index page set or partition
- The index is not on a DB2 catalog or directory table space

Because of the possible increase in P-lock activity with row locking, evaluate row locking carefully before using it in a data sharing environment. If you have an update-intensive application process, the amount of page P-lock activity might increase the overhead of data sharing.

To decrease the possible contention on those page P-locks, consider using page locking and a MAXROWS value of one on the table space to simulate row locking. You can get the benefits of row locking without the data page P-lock contention that comes with it. A new MAXROWS value does not take effect until you run REORG on the table space.

### P-lock monitoring

Monitoring activity of P-locks, especially page P-locks, in a DB2 data sharing environment can help you determine if you need to control inter-DB2 read/write interest.

More overhead exists when inter-DB2 read/write interest exists in a page set. Although DB2 dynamically tracks inter-DB2 read/write interest, which helps avoid data sharing overhead when it is not needed, you will pay some cost for the advantages of data sharing.

If excessive global contention exists that cannot be resolved by any tuning measures, you might need to reduce the locking overhead by keeping some transactions and data together on a single system.

#### How to find information about page set P-locks:

You can use the DISPLAY DATABASE command with the LOCKS option to find information about page set P-locks, including which member is holding or waiting for P-locks, and whether P-locks are being held because of a DB2 failure.

The following figure has sample output obtained from the command. Figure 54 on page 185

NAME	TYPE	PART	STATUS	CONNID	CORRID	LOCKINFO
DSN8S10D	TS		RW			H-IX,PP,I
				MEMBER NAME	DB1A (CO)	
DSN8S10D	TS		RW			H-IX,PP,I
				MEMBER NAME	DB2A	

Figure 56. Sample DISPLAY DATABASE output showing GBP-dependent table spaces

A "PP" in the LOCKINFO field of the output indicates that a particular lock is a page set or partition P-lock.

You can also obtain information about P-locks, along with information about transaction locking, from the statistics and accounting traces. Performance class 20 (IFCID 0251) also contains information about P-lock negotiation requests. IFCID 0251 is mapped by DSNDQW04.

### **How to find information about page P-locks:**

Page P-locking activity is recorded, along with the rest of the data sharing locking information, in the statistics and accounting trace classes.

You can find more detail about those page P-locks in performance trace class 21 (IFCID 0259). IFCID 0259 allows you to monitor page P-locking without having to use a full DB2 lock trace. IFCID 0259 is mapped by DSNDQW04.

### **Options for reducing space map page contention**

When you define table spaces and indexes, you have several options to help reduce space map hot spots. These hot spots can occur when a large amount of update, insert, or delete activity occurs on a page set from multiple members of a group.

The MEMBER CLUSTER option can reduce contention when doing heavy sequential inserts, and the TRACKMOD NO option can reduce contention when any type of insert, update, or delete activity occurs.

Understand the implications of each option before choosing either one, as both options have drawbacks.

#### **Member affinity clustering:**

For applications that do heavy sequential insert processing from multiple members, the contention on the space map or for the data pages at the end of the table can be considerable.

The MEMBER CLUSTER option of CREATE TABLESPACE causes DB2 to manage space for inserts on a member-by-member basis instead of by using one centralized space map. Table spaces defined with MEMBER CLUSTER have the following characteristics:

- Data that is inserted by the SQL INSERT statement is not clustered by the implicit clustering index (the first index) or the explicit clustering index.
- DB2 chooses where to locate the data in such a way that avoids lock and latch contention. In general, it tries to insert data in a place that is covered by the locally cached space map page. If it cannot find space there, it continues to search through space map pages until it can find a place for which the space map page is available. As a result, space in a data set might not be fully used. But when the data set reaches the maximum number of extents, lock contention can increase and DB2 does use the entire space.
- Each space map covers 199 data pages. Because there are more space map pages and some might be partially used, table spaces that are defined with MEMBER CLUSTER can use more disk.
- To reduce the overhead of reacquiring page P-locks, a page P-lock is held longer for MEMBER CLUSTER table spaces.

The downside to using MEMBER CLUSTER is that data is not inserted in clustering order. If you have a query application that performs best when data is in clustering order, you should run REORG on the table space before starting the query application.

#### **Indexes with randomized key columns to reduce hot spots:**

You can randomize the index key columns in database tables where you observe hot spots within the indexes during INSERT, DELETE, or UPDATE processing by using the new RANDOM option on the CREATE INDEX statement.

This option can be used when you want to alter an index to add a key column that is randomized. Note that even though the indexes are stored in random order, it is still possible to get index only access. Also, when the index is in random order it is only usable for equality lookups.

Example: A user defines an index on the EMPNO column of the EMP table in ASCENDING order as follows:

```
CREATE INDEX DSN81010.XEMP3
           ON DSN81010.EMP
           (EMPNO ASC);
```

The user then continually inserts an ascending sequence of values into the EMPNO column (000010, 000020, 000030, ...). These values are always added to the end of the index, creating a hotspot at the end of the index. In this particular index, the user only wishes to look up specific employees by their EMPNO (i.e. only equality predicates are applied to the EMPNO column). To reduce contention, the user changes the definition of the index to the following:

```
CREATE INDEX DSN81010.XEMP3
           ON DSN81010.EMP
           (EMPNO RANDOM);
```

With the RANDOM specification, the EMPNO values will now be spread randomly throughout the index, preventing the contention that results from always inserting values at the end of the index. Since the common use of this index is for looking up specific employees by their EMPNO, and the user wants to avoid a hotspot at the end, the new RANDOM ordering option provides a good solution.

#### **Control of tracking updates to reduce coupling facility overhead:**

The TRACKMOD NO option of CREATE or ALTER TABLESPACE can reduce coupling facility overhead caused by constant updating of the space map pages of the page sets.

With TRACKMOD NO, DB2 does not keep track of changed pages in the space map page of the page set. By choosing TRACKMOD NO and not tracking updates, less coupling facility overhead exists, but the cost of incremental image copies is much higher because DB2 must use a table space scan to read all pages to determine if the page has been changed and thus needs to be copied.

If longer copy times means you must take fewer incremental copies, monitor your active log data sets to ensure that you do not have to go to a tape archive data set to do recovery. You might need to make the active log data sets larger, specify more active log data sets, or archive to disk to avoid this possibility.

**Recommendation:** If you rarely or never use incremental image copies, or if you always use DFSMS concurrent copy with DB2 LOGONLY recovery, use TRACKMOD NO.

## **Read operations**

In a data sharing environment, the process of reading data is different from the process in a non-data sharing environment.

## Where DB2 looks for a page

DB2 searches for pages in buffer pools and on disk in a specific order.

DB2 searches for pages in this order:

1. In the local buffer pool. If the page is invalid, DB2 refreshes the page from the group buffer pool (or disk).
2. In the group buffer pool. DB2 checks the group buffer pool for a page if the page set is defined as GBPCACHE ALL or if the page set or the partition is GBP-dependent, unless one of the following conditions is true:
  - Group buffer pool is defined as GBPCACHE(NO).
  - Page set is defined as GBPCACHE NONE.
  - Page set is defined as GBPCACHE SYSTEM and the page being read is not a space map page.
3. On disk. If the page is not in the group buffer pool, DB2 refreshes the page in the buffer pool from disk.

For duplexed group buffer pools, read activity occurs only against the primary structure.

## Testing the page validity

Part of the process of controlling cache coherency is testing to see if a page that is referenced in the buffer pool must be refreshed from the group buffer pool or disk because it might no longer be the most current version of the data.

This process is known as testing the page *validity*. Because DB2 tracks the level of interest in a page set across the group, it is not always necessary to make this test. The following table indicates when this test is performed.

For duplexed group buffer pools, only the primary structure is used for cross-invalidations.

*Table 33. Determining when page validity must be tested*

One member's interest	Other members' interest	Test page validity?
Read-only	None, Read-only	No
Read-only	Read/Write	Yes
Read/Write	None	No
Read/Write	Read-only	No
Read/Write	Read/Write	Yes

## Prefetch processing

DB2 prefetch processing for GBP-dependent page sets and partitions varies depending on the level of coupling facility (CFLEVEL) in which the group buffer pool is allocated.

If the group buffer pool is allocated in a coupling facility with CFLEVEL=0 or 1, DB2 reads and registers one page at a time in the group buffer pool.

If the group buffer pool is allocated in a coupling facility with CFLEVEL=2 or higher, DB2 can register the entire list of pages that are being prefetched with one request to the coupling facility. This can be used for sequential prefetch (including sequential detection) and list prefetch. On the list, DB2 does not include any valid pages that are found in the local buffer pool.

For those pages that are cached as “changed” in the group buffer pool, or those that are locked for castout, DB2 still retrieves the changed page from the group buffer pool one at a time. For large, sequential queries, changed pages most likely do not exist in the group buffer pool.

For pages that are cached as “clean” in the group buffer pool, DB2 can get the pages from the group buffer pool (one page at a time), or can include the pages in the disk read I/O request, depending on which is most efficient.

### How to determine if DB2 registered a list of pages

If DB2 registers a list of pages during prefetch, there will be a non-zero value in field QBGLAX in IFCID 0002 (see **K** in the OMEGAMON statistics report). You can also use the DISPLAY GROUPBUFFERPOOL command with the MDETAIL option (see message DSNB789I).

#### Related concepts

“What to look for in a OMEGAMON statistics report” on page 217

### Use of caching for group buffer pools

You can cache pages in the group buffer pool as they are read into a member's local buffer pool by specifying GBPCACHE ALL when you create or alter a table space or index.

When you choose ALL, pages are copied to the group buffer pool as they are read in from disk, even if no inter-DB2 read/write interest exists in those pages.

However, when only a single member has exclusive read/write interest in the page set (that is, only one member has the page set open for update), pages are not cached in the group buffer pool when they are read in from disk. As soon as another member in the data sharing group shows interest in the page set or partition, it becomes GBP-dependent. Changed pages are moved into the group buffer pool, and all pages read in from disk are cached in the group buffer pool.

Choosing GBPCACHE ALL does not prevent DB2 from continuing to cache changed pages in the group buffer pool before writing them to disk (the function provided by the default, GBPCACHE CHANGED).

**Note:** For LOB table spaces, use the default, GBPCACHE CHANGED. If you use GBPCACHE CHANGED for a LOB table space that is defined with LOG NO and the coupling facility fails, the LOB table space is placed in GRECP. When group buffer pool recovery occurs, all LOB values that were in the coupling facility at the time of the failure are marked invalid because the log records that are necessary to perform the recovery for those values are missing due to the LOG NO attribute.

**Example:** Use the GBPCACHE clause to cache read-only page sets in the group buffer pool:

#### GUPI

```
ALTER TABLESPACE DSN8D10A.DSN8S10D
  GBPCACHE ALL;
```

#### GUPI

## Why choose GBPCACHE ALL?

By choosing GBPCACHE ALL, you prevent multiple members from reading the same page in from disk. For this reason, page sets or partitions that typically have a high degree of inter-DB2 read interest are good candidates for GBPCACHE ALL. (To prevent the double buffering of clean pages, hiperpools are not used for page sets or partitions that are defined with GBPCACHE ALL.)

### Planning consideration

If you use the GBPCACHE ALL option, it increases the need for coupling facility resources: processing power, storage, and channel utilization. If you cannot afford the additional strain on coupling facility resources, consider using a 3990 Model 6 cache controller that exploits record and track level caching to achieve caching benefits for a read-intensive work load.

## Write operations

With data sharing, DB2 usually writes changed data to the group buffer pool before writing that changed data to disk.

### How the GBPCACHE option affects write operations

You can use several options in a CREATE TABLESPACE statement to tell DB2 how you want data to be handled through the group buffer pool.

### GBPCACHE CHANGED or ALL

GBPCACHE CHANGED for LOB is the default in DB2. For both ALL and CHANGED, the write operations are the same for changed pages: changed pages are written to the group buffer pool before being written to disk.

### GBPCACHE SYSTEM

This option is allowed only for LOBs. For GBPCACHE SYSTEM page sets, the only pages that are written to the group buffer pool are LOB space map pages. All other data pages are written directly to disk, similar to GBPCACHE NONE page sets. GBPCACHE SYSTEM is the default for a LOB table space.

**Recommendation:** In a data sharing environment, if GBPCACHE CHANGED or GBPCACHE ALL is used instead for a LOG NO LOB table space, a coupling facility failure can result in the table space being marked GRECP. Any kind of recovery of the table space marks the attached LOB values as invalid and places the table space in the AUXW state. For LOB table spaces, choose GBPCACHE SYSTEM to avoid having large LOB values overwhelm the group buffer pool. Also, for LOB table spaces with the LOG NO attribute, GBPCACHE SYSTEM ensures that LOB values are written to disk by commit time, thereby avoiding possible recovery problems caused by missing log data.

### GBPCACHE NONE

For GBPCACHE NONE page sets, or for page sets defined in a group buffer pool as GBPCACHE(NO), no pages are written to the group buffer pool. The group buffer pool is used only for the purpose of buffer cross-invalidation. At every COMMIT, any pages that were updated by the transaction and have not yet been written are synchronously written to disk during commit processing. This can have a severe impact on performance for most types of transactions.

One potential advantage of not caching pages in the group buffer pool is that data does not have to be recovered from the log if the coupling facility fails. However, because DB2 still depends on the cross-invalidation information that is stored in the group buffer pool, a coupling facility failure still means some data might not be available. That is why specifying an alternate coupling facility in the CFRM policy is recommended. If you are looking for a high availability option, consider duplexing the group buffer pool rather than suffering the performance hit of writing directly to disk at every COMMIT.

*Advantage of specifying GBPCACHE(NO) for group buffer pools:* You can specify GBPCACHE(NO) on the group buffer pool level instead of on the page set level. Changing the group buffer pool attribute is usually less disruptive than changing the page set attribute. Because the GBPCACHE(NO) attribute takes precedence over the GBPCACHE option on the page set, you can plan to use different types of processing at different times of day. For example, assume that you want to run transactions or queries during the day, but you want to do batch updates at night. Assume also that your batch updates would benefit from no group buffer pool data caching. You can do the following:

1. Define the table space as GBPCACHE CHANGED and put it into its own buffer pool.
2. Define the corresponding group buffer pool as GBPCACHE(YES) for daytime processing.
3. At night, use the ALTER GROUPBUFFERPOOL command to change the group buffer pool to GBPCACHE(NO).
4. Set the SETXCF START,REBUILD,STRNAME=*strname* command to enable the new attribute.
5. In the morning, use the ALTER GROUPBUFFERPOOL command to change the group buffer pool back to GBPCACHE(YES).
6. Issue the SETXCF START,REBUILD,STRNAME=*strname* command to enable the new attribute.

*Reasons not to cache:* A performance benefit is possible for applications in which an updated page is rarely, if ever, referenced again, such as a batch job that sequentially updates a large table. By not caching, you save the costs of transferring data to the group buffer pool and casting out from the group buffer pool. Again, this benefit is at the cost of synchronous disk I/O at COMMIT. To reduce the amount of synchronous disk I/O at COMMIT, you can lower the deferred write thresholds so that DB2 writes more pages asynchronously prior to COMMIT.

To determine if a particular group buffer pool is a candidate for GBPCACHE(NO), look at the group buffer pool statistics. If the ratio of READS, DATA RETURNED / PAGES WRITTEN is less than 1%, this page set might be a good candidate for GBPCACHE(NO), or the page sets using this group buffer pool might be a good candidate for GBPCACHE NONE. You receive the following benefits:

- Reduced coupling facility costs and faster coupling facility response time
- Reduced processor time on the host system
- Better transaction throughput at a small possible cost in higher transaction response time

If you use GBPCACHE NONE, enable DASD Fast Write and set the vertical deferred write threshold (VDWQT) to 0. By setting this threshold to 0, you let deferred writes happen continuously before the COMMIT, thus avoiding a large surge of write activity at the COMMIT.

## Related concepts

“How DB2 writes to the group buffer pool”

“Use of caching for group buffer pools” on page 192

## How DB2 writes to the group buffer pool

With data sharing, DB2 still performs deferred writes for DB2 table spaces, indexes or partitions. However, when an update is to a page set that has inter-DB2 read/write interest, DB2 forces the updated pages to the group buffer pool before or when the transaction commits.

Updated pages can be written to the group buffer pool before the updating transaction is committed when:

- One of the deferred write thresholds is reached.
- The buffer pool lacks reassignable buffers because writes to the group buffer pool cannot keep up with update activity in the buffer pool. The shortage of buffers can occur when the deferred write thresholds are too high, or if the application is not committing frequently enough—in a data sharing environment, the commits make buffers reassignable.
- An updated page has stayed in the buffer pool for a long period of time since it was last referenced or updated (such as with a long-running transaction that does not issue frequent commits). In this case, a system checkpoint can free space in the buffer pool before the commit.
- The same page is required for update by another system because no conflict on transaction locking exists (such as page sets that are using row locking, index pages, space map pages, and so on). This write is part of the page P-lock negotiation process.

When a page of data is written to the group buffer pool, all copies of that page cached in other members' buffer pools are invalidated. This means that the next time that one of those members needs that page, the page must be refreshed from the group buffer pool (or disk).

Before an updated page is written to the group buffer pool or to disk, DB2 also ensures that the last update log record for that page is externalized to the active log. This ensures that updates can be backed out when necessary.

When committing an updating transaction, DB2 synchronously writes to the group buffer pool pages that were updated but not yet written to the group buffer pool. If a group buffer pool is required and unavailable (because of a channel or hardware failure) at the time the transaction commits, DB2 places all the transaction's updated pages on the logical page list (LPL) associated with each page set. After the problem is fixed, DB2 attempts automatic LPL recovery, but you can issue a `START DATABASE` command with the `SPACENAM` option to manually recover the pages on the logical page list.

**Note:** When a table space or partition is placed in the LPL, at either restart time or during rollback processing, because undo processing is needed for a `NOT LOGGED` table space, automatic LPL recovery is not initiated and a `START DATABASE` command identifying this table space will have no effect on its LPL status.

## Writing to a GBPCACHE(NO) group buffer pool

For GBP-dependent page sets, no data is written to a group buffer pool for the following instances:

- The group buffer pool is defined as GBPCACHE(NO)
- The page set is defined as GBPCACHE NONE
- The changed pages are non-system pages of a page set defined with GBPCACHE SYSTEM

In these cases, DB2 writes changed pages for the transaction directly to disk at or before COMMIT. DB2 batches up to 32 pages in a single I/O.

When DB2 writes a changed page to disk, it cross-invalidates the page using the group buffer pool. These cross-invalidations are called *explicit cross-invalidations*. These explicit cross-invalidations are reported in statistics separately from cross-invalidations caused by directory reclaims or by writes of a changed page to a group buffer pool.

### Writing to a duplexed group buffer pool

When a group buffer pool is duplexed, page writes are performed in the following manner:

1. For some fixed number of pages that must be written:
  - a. Each page is written to the secondary group buffer pool asynchronously
  - b. Each page is written to the primary group buffer pool synchronously
2. After all pages have been written to the primary group buffer pool, DB2 checks to see if all pages have been written to the secondary group buffer pool. If some pages still need to be written, DB2 forces the completion of those writes.

#### Related concepts

“Database monitoring options” on page 87

### How DB2 writes from the group buffer pool to disk

The process of writing pages from the group buffer pool to disk is called *castout*.

Because no physical connection exists between the group buffer pool and disk, the castout process involves reading the page from the group buffer pool into a particular member's private buffer (not part of the buffer pool storage), and then writing the page from the private buffer to disk. This member is the owner of the castout process for the page set or partition. The first member with update intent on the page set or partition is assigned ownership of castout. After castout ownership is assigned, subsequent updating members become backup owners. One of the backup owners becomes the castout owner when the original castout owner no longer has read/write interest in the page set.

Other members can write this page to the group buffer pool even as the page is being cast out. Some events explicitly cause pages to be cast out to disk, such as the STOP DATABASE command.

Castout also occurs when:

- The number of changed pages for a castout class queue exceeds a class threshold value.
- The total number of changed pages for a group buffer pool exceeds a group buffer pool threshold value.
- The group buffer pool checkpoint is triggered.
- No more inter-DB2 read/write interest exists in the page set.
- The group buffer pool is being rebuilt, but the alternate group buffer pool is not large enough to contain the pages from the group buffer that is being rebuilt.

Pages that are cast out as a result of meeting a threshold remain cached in the group buffer pool, and the buffers are available for stealing. Pages that are cast out because no more shared interest exists in the page set are purged from the group buffer pool.

### Casting out from a duplexed group buffer pool

DB2 casts out data to disk only from the primary structure. After a set of pages has been cast out, the same set of pages is deleted from the secondary structure. See the DELETE NAME LIST counter in the DISPLAY GROUP BUFFERPOOL MDETAIL report for how many times this event occurs. DB2 ensures that any pages that might have been written to the group buffer pool during castout processing are not deleted from the secondary structure.

#### Related concepts

“Group buffer pool class castout threshold” on page 201

“Group buffer pool checkpoint” on page 198

#### Displaying the castout owner:

You can use the DISPLAY DATABASE command or the DISPLAY BUFFERPOOL command to display the castout owner for a page set or partition.

**GUIP** Use the DISPLAY DATABASE command with the LOCKS option to display the current castout owner for a given page set:

```
-DB1A DISPLAY DATABASE(TESTDB) SPACE(*) LOCKS
```

Display the castout owner for a particular page set or partition with (CO) by the member name, as shown in the following figure.

```

:
TBS43 TS 01 RW MEMBER NAME DB2A (CO) H-SIX,PP,I
-
TBS43 TS 02 RW BATCH SELEC H-IS,P,C
- MEMBER NAME DB1A
:

```

Figure 57. Partial DISPLAY DATABASE output showing castout owner for a partition

Use the DISPLAY BUFFERPOOL command with the CASTOWNR keyword to determine for which page sets or partitions the members hold castout ownership:

```
-DIS BUFFERPOOL(BP0) CASTOWNR(Y)
```

The following figure shows output that lists the members that hold the page set or partition P-lock in "U" state.

```

@DIS BPOOL(BPO) CASTOWNR(Y)
DSNB401I @ BUFFERPOOL NAME BPO, BUFFERPOOL ID 0, USE COUNT 40
DSNB402I @ VIRTUAL BUFFERPOOL SIZE = 500 BUFFERS
          ALLOCATED      =      500  TO BE DELETED    =      0
          IN-USE/UPDATED =      0
DSNB406I @ VIRTUAL BUFFERPOOL TYPE -
          CURRENT        = PRIMARY
          PENDING        = PRIMARY
          PAGE STEALING METHOD = LRU
DSNB404I @ THRESHOLDS -
          VP SEQUENTIAL  = 80
          DEFERRED WRITE = 85  VERTICAL DEFERRED WRT = 80,0
          PARALLEL SEQUENTIAL = 50  ASSISTING PARALLEL SEQT= 0
DSNB460I @

```

```

-----PAGE SET/PARTITION LIST INFORMATION-----
-----DATA SHARING INFO-----
          TS GBP MEMBER CASTOUT USE P-LOCK
DATABASE SPACE NAME PART IX DEP  NAME  OWNER  COUNT STATE
-----
DSNDB01  DSNLLX02          IX Y  V61A      Y      0  IX
          V61B      Y      0  IX
DSN8D71A DSN8S71E      001 TS Y  V61A      Y      1  SIX
          003 TS Y  V61A      Y      1  SIX
DSNDB01  DSNLLX01          IX Y  V61A      Y      0  IX
          V61B      Y      0  IX
          SYSLGRNX      TS Y  V61A      Y      0  IX
          V61B      Y      0  IX
DSN9022I @ DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION

```

Figure 58. Partial DISPLAY DATABASE output showing the list of members that hold the page set or partition P-lock in "U" state



## Group buffer pool checkpoint

When a group buffer pool is damaged, all changed data that belong to GBP-dependent page sets must be recovered to the page sets from the DB2 logs.

The number of log records that need to be applied to the page set is determined by the frequency of the group buffer pool checkpoint. *Group buffer pool checkpoint* is the process of writing all changed pages in the group buffer pool (only the primary one, if duplexed) to the page set. The purpose of the checkpoint is to reduce the amount of time needed to recover data in a group buffer pool. At group buffer pool checkpoint, DB2 records, in the member BSDSs and SCA, the log record sequence number from which group buffer pool recovery would need to take place. Group buffer pool checkpoint does not record anything in the log.

The group buffer pool checkpoint is triggered by the structure owner. The *structure owner* is usually the first member that connects to this group buffer pool, although the ownership can change over time. Message DSNB798I in the DISPLAY GROUPBUFFERPOOL command output shows which member is the current structure owner.

### Default checkpoint frequency:

You can change the default checkpoint frequency by using the ALTER GROUPBUFFERPOOL command.

The default checkpoint frequency is 4 minutes. For group buffer pools defined as GBPCACHE(NO), the checkpoint interval is ignored; no checkpointing occurs for those group buffer pools.

### Related concepts

“Changing the checkpoint frequency” on page 221

### How DB2 gathers checkpoint information:

At the group buffer pool checkpoint, the structure owner records in the SCA and in its own BSDS, the LRSN from which group buffer pool recovery should take place, if necessary.

This LRSN is displayed in the DISPLAY GROUPBUFFERPOOL command output. DB2 has two possible ways of gathering checkpoint information:

- By issuing many “read directory info” requests  
This method is used when the Parallel Sysplex is not at the maintenance level required for the more efficient method described below. These “read directory info” requests are reported as an increase in the SRB time of the *ssnmDBM1* address space, especially for the structure owner. This is because of the increased number of times that DB2 has to read the directory entries to compute the recovery LRSN.
- By issuing one “read castout statistics” request  
This method is used when the group buffer pool is allocated in a coupling facility at CFLEVEL=5 or higher.  
When you look at the MDETAIL report from a DISPLAY GROUPBUFFERPOOL command you will see significantly fewer “read directory info” requests when you have the proper maintenance applied for this feature.

**Recommendation:** Because group buffer pool checkpoint consumes processor, coupling facility, and I/O resources and can impact other work in the system, balance the performance impact of frequent group buffer pool checkpoints (the lower the checkpoint interval, the higher the system resource consumption) with the recovery impact of infrequent checkpoints (the lower the checkpoint interval, the faster DB2 can recover from a group buffer pool failure). The default checkpoint interval of 4 minutes is a good balance between the performance and recovery considerations in most cases.

### Related tasks

“Tuning the group buffer pool checkpoint interval”

### Tuning the group buffer pool checkpoint interval:

You can monitor your group buffer pool checkpoint intervals and adjust the intervals if necessary.

If the resource consumption of the group buffer pool checkpoint is higher than you prefer:

- Apply the proper maintenance and allocate the group buffer pool in a CFLEVEL=5 coupling facility to take advantage of the checkpoint performance enhancement.
- Increase the checkpoint interval to have the checkpoint occur less frequently.

If the checkpoint is not moving the recovery LRSN forward fast enough, decrease the checkpoint interval. You can determine the LRSN by periodically issuing the `DISPLAY GROUPBUFFERPOOL` command.

The following instrumentation helps you more effectively monitor and tune the group buffer pool checkpoint:

- The `DISPLAY GROUPBUFFERPOOL` command in the following figure shows which member is the structure owner and also shows the group buffer pool checkpoint recovery LRSN.

```
DSNB798I -DB1A LAST GROUP BUFFER POOL CHECKPOINT 17:23:21 OCT  4, 2002
          GBP CHECKPOINT RECOVERY LRSN           = ACD74C01EE30
          STRUCTURE OWNER                       = DB1A
```

*Figure 59. Partial DISPLAY GROUPBUFFERPOOL output showing which member owns the structure and the group buffer pool checkpoint recovery LRSN*

- The `DISPLAY GROUPBUFFERPOOL` command with `MDETAIL` option in the following figure contains the number of checkpoints that occurred for this group buffer pool. The statistics trace also includes this information.

```
DSNB778I -DB1A CASTOUT THRESHOLDS DETECTED
          FOR CLASSES                               = 3
          FOR GROUP BUFFER POOL                   = 1
          GBP CHECKPOINTS TRIGGERED                = 1
          PARTICIPATION IN REBUILD                 = 0
DSNB796I -DB1A CASTOUTS
          PAGES CASTOUT                             = 18
          UNLOCK CASTOUT                           = 3
          READ CASTOUT CLASS                       = 5
          READ CASTOUT STATISTICS                   = 6
          READ DIRECTORY INFO                       = 0
```

*Figure 60. DISPLAY GROUPBUFFERPOOL MDETAIL report*

If you are experiencing surges of coupling facility use, it could be related to group buffer pool checkpointing. Examine the number of read directory info requests. If there are many requests per checkpoint, you can probably benefit from applying the proper maintenance to use the group buffer pool checkpoint performance enhancement.

- IFCID 0261, which gives summary statistics for each group buffer pool checkpoint. You can use this record to estimate the processor cost and to monitor the coupling facility interactions for each group buffer pool checkpoint.
- IFCID 0263, which gives summary statistics for the castouts. You can use this record to monitor the castout activity that is caused by each group buffer pool checkpoint (or triggered for any other reason).

If the recovery LRSN for group buffer pool checkpoint is not advancing as fast as you want, determine if there have been any disk or coupling facility connectivity problems that are impairing the ability of DB2 to cast out.

## Group buffer pool thresholds

You can control the castout process with two group buffer pool thresholds.

The two group buffer pool thresholds are:

- Group buffer pool castout threshold
- Class castout threshold

These thresholds have no effect for GBPCACHE(NO) group buffer pools.

As the figure below illustrates, the group buffer pool castout threshold is a percentage of changed pages in the group buffer pool. The class castout threshold is the percentage of changed pages in the group buffer pool per *castout queue*.

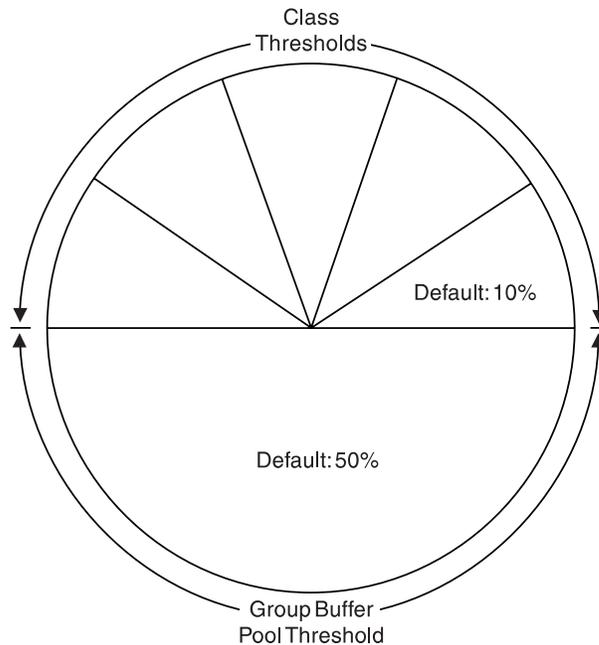


Figure 61. Group buffer pool castout thresholds. Both thresholds are expressed as percentages of the total number of pages in the group buffer pool.

### Group buffer pool class castout threshold

Each group buffer pool contains a fixed number of castout class queues. This number is an internal value that is set by DB2.

DB2 internally maps updated pages that belong to the same page sets or partitions to the same castout class queues. Because of a limited number of castout class queues, it is possible that more than one page set or partition gets mapped into the same castout class queue. This internal mapping scheme is the same across all sharing subsystems.

When DB2 writes changed pages to the group buffer pool, it determines how many changed pages are on a particular class castout queue. If the number of changed pages on a specified castout class queue exceeds the threshold, DB2 casts out a number of pages from that queue.

**How DB2 determines the castout threshold for a duplexed group buffer pool:** For duplexed group buffer pools, DB2 uses the smaller of the number of data entries in the primary and secondary structures. For example, if the primary structure contains 5000 data entries and the secondary structure contains 1000 data entries, and CLASST is 5%, DB2 sets CLASST to 50 pages (5% of 1000 pages).

### Default group buffer pool class castout threshold

The default for the class castout is 5, which means that castout is initiated for a particular page set or partition when 5% of the group buffer pool contains changed pages for the class.

## Group buffer pool castout threshold

The group buffer pool castout threshold determines the total number of changed pages that can exist in the group buffer pool before castout occurs. DB2 casts out enough class castout queues to bring the number of changed pages below the threshold. DB2 periodically determines whether the threshold is exceeded.

*How DB2 determines the group buffer pool castout threshold for a duplexed group buffer pool:* For duplexed group buffer pools, DB2 uses the smaller of the number of data entries in the primary and secondary group buffer pools. For example, if the primary structure contains 5000 data entries and the secondary structure contains 1000 data entries, and GBPOOLT is 30%, DB2 sets GBPOOLT to 300 pages (30% of 1000 pages).

## Default group buffer pool castout threshold

The default value for the group buffer pool castout threshold is 30, which means that when the group buffer pool is 30% full of changed pages, castout is initiated.

## Recommended castout threshold for LOBs

You should set the group buffer pool castout threshold to zero, or a low value, to reduce the need to have a large group buffer pool for LOBs.

## Guidelines for group buffer pool thresholds

In most cases, you should use the CLASST and GBPOOLT thresholds to be the corresponding VDWQT and DWQT thresholds for the local buffer pools if these values perform well locally.

Otherwise, you can use the default values (5% for the class threshold and 30% for the group buffer pool threshold). Depending on your work load, these values help reduce disk contention during castout.

If you find that some writes to the group buffer pool cannot occur because of a lack of storage in the group buffer pool, increase the group buffer pool size, or decrease the group buffer pool castout thresholds. One way to tell if this is happening is to see the detail report of the DISPLAY GROUPBUFFERPOOL command. The following figure shows an example report that indicates the lack of storage problem. Figure 67 on page 211

```

DSNB783I -DB1A CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 May 17, 2002
DSNB784I -DB1A GROUP DETAIL STATISTICS
      READS
      DATA RETURNED A = 3845
DSNB785I -DB1A DATA NOT RETURNED
      DIRECTORY ENTRY EXISTED B = 27
      DIRECTORY ENTRY CREATED C = 28336
      DIRECTORY ENTRY NOT CREATED D = 332, 0

DSNB786I -DB1A WRITES
      CHANGED PAGES = 20909
      CLEAN PAGES = 0
      FAILED DUE TO LACK OF STORAGE E = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB787I -DB1A RECLAIMS
      FOR DIRECTORY ENTRIES F = 18281
      FOR DATA ENTRIES = 47
      CASTOUTS = 16073
DSNB788I -DB1A CROSS INVALIDATIONS
      DUE TO DIRECTORY RECLAIMS G = 4489
      DUE TO WRITES = 3624
      EXPLICIT = 0
DSNB762I -DB1A DUPLEXING STATISTICS FOR GBP0-SEC
      WRITES
      CHANGED PAGES = 20909
      FAILED DUE TO LACK OF STORAGE = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB790I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1A DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION

```

Figure 62. Example output of group detail statistics

The field indicated by **E** indicates this type of problem.

### Effect of GBPCACHE ALL on guidelines

If you are using a group buffer pool to cache pages as they are read in from disk (GBPCACHE ALL page sets), consider lowering the threshold values to allow more space for caching those clean pages.

#### Ways to tune the castout thresholds:

Several facilities can help you more efficiently monitor group buffer pool castout thresholds.

Use the following facilities to help you monitor the castout thresholds:

- The DISPLAY GROUPBUFFERPOOL command with the MDETAIL option.
- The DB2 statistics trace.
- **PSPI** IFCID 0262, which gives summary statistics for each time that the GBPOOLT threshold is reached. You can use this record to monitor how efficiently the GBPOOLT threshold is handling the castout work.
- IFCID 0263, which gives summary statistics for the castouts done by the page set and partition castout owners. All castout work for a given page set or partition is done by the castout owner. You can use this record to monitor the efficiency with which the page set or partition castout owners are doing their work.

**PSPI**

#### Example from MDETAIL report:

Using the MDETAIL option of the DISPLAY GROUPBUFFERPOOL command is one option for monitoring the group buffer pool castout thresholds.

#### GUIP

The following example is a partial output from the command DISPLAY GROUPBUFFERPOOL (GBP0) MDETAIL

```
⋮
DSNB796I -DB1A CASTOUTS
          PAGES CASTOUT           = 217
          UNLOCK CASTOUT          = 35
          READ CASTOUT CLASS      = 47
          READ CASTOUT STATISTICS = 47
          READ DIRECTORY INFO     = 290
⋮
```

#### GUIP

The UNLOCK CASTOUT counter should always be significantly less than the PAGES CASTOUT counter. If, at the very least, it is not less than half, the castout write I/O is not performed efficiently. (The number of pages written per I/O is normally close to the number that is obtained by dividing PAGES CASTOUT by UNLOCK CASTOUT). This is probably because you have random update patterns on the DB2 data.

## Ways to monitor group buffer pools

You can monitor group buffer pool activity by using a z/OS or DB2 command, by using a coupling facility activity report, or by using a statistics trace.

### Group buffer pool monitoring with the z/OS DISPLAY XCF,STR command

You can monitor group buffer pools with the DISPLAY XCF,STR command.

You can use z/OS command D XCF,STR to get information about coupling facility structures:

- CFRM policy definition
- Preference list
- Coupling facility name
- Connections
- Duplexing status

The following command displays information about GBP1 in group DSNDB0A:

```
D XCF,STR,STRNAME=DSNDB0A_GBP1
```

This particular group buffer pool is duplexed, so you see information about both allocations of the structure (the old structure is the primary structure, and the new structure is the secondary one). The output is similar to the output that is shown in the following figure

:

```

DISPLAY XCF
STRNAME: DSNDB0A_GBP1
STATUS: REASON SPECIFIED WITH REBUILD START:
        OPERATOR INITIATED
        DUPLEXING REBUILD
        REBUILD PHASE: DUPLEX ESTABLISHED
POLICY SIZE      : 204800 K
POLICY INITSIZE : 102400 K
REBUILD PERCENT : 1
DUPLEX          : ALLOWED
PREFERENCE LIST : CACHE01 LF01
EXCLUSION LIST IS EMPTY

```

#### DUPLEXING REBUILD NEW STRUCTURE

```

-----
ALLOCATION TIME: 10/14/2002 17:01:48
CFNAME       : LF01
COUPLING FACILITY: ND01...
                PARTITION: 0   CPCID: 00
ACTUAL SIZE  : 102400 K
STORAGE INCREMENT SIZE: 256 K
VERSION      : AF6935AA 78004403
DISPOSITION  : DELETE
ACCESS TIME  : 0
MAX CONNECTIONS: 32
# CONNECTIONS : 2

```

#### DUPLEXING REBUILD OLD STRUCTURE

```

-----
ALLOCATION TIME: 10/14/2002 17:00:38
CFNAME       : CACHE01
COUPLING FACILITY: ND02...
                PARTITION: 0   CPCID: 00
ACTUAL SIZE  : 102400 K
STORAGE INCREMENT SIZE: 256 K
VERSION      : AF693567 9B48B802
ACCESS TIME  : 0
MAX CONNECTIONS: 32
# CONNECTIONS : 2

```

CONNECTION NAME	ID	VERSION	SYSNAME	JOBNAME	ASID	STATE
DB2_DB1A	01	00010001	UTEC469	DB1ADBM1	002E	ACTIVE NEW,OLD
DB2_DB2A	02	00020001	UTEC469	DB2ADBM1	0031	ACTIVE NEW,OLD

Figure 63. z/OS Command D XCF showing group buffer pool information

### Related information

 MVS System Commands

## Group buffer pool monitoring with the coupling facility activity report of RMF

You can use RMF coupling facility structure reports to monitor group buffer pools.

The figure below shows a portion of an example report.

A value for CHNGD ( **B** ) is the percentage of all accesses that were supposed to be done synchronously that had to be done asynchronously. The NO SCH field ( **A** ) indicates the amount of time that requests were queued because of a lack of subchannel resources. If the value in **B** is over 10% or so, and **A** contains a non-zero value, your configuration might not have enough subchannels to handle your systems' workload.

STRUCTURE NAME = DSNDB0A_GBP8		TYPE = CACHE					DELAYED REQUESTS					
SYSTEM	# REQ	REQUESTS			-SERV TIME(MIC)-		REASON	#	% OF	AVG TIME(MIC)		
NAME	AVG/SEC	# REQ	% OF ALL	AVG	STD_DEV		REQ	REQ	/DEL	STD_DEV	/ALL	
STLABC2	66662	SYNC	61K	43.2%	107.7	32.5	<b>A</b>					
	370.3	ASYNC	5677	4.0%	602.8	419.8	NO SCH	0	0.0%	0.0	0.0	0.0
		<b>B</b> CHNGD	0	0.0%	INCLUDED IN ASYNC							
							DUMP	0	0.0%	0.0	0.0	0.0

Figure 64. Portion of RMF Coupling Facility Structure Activity report

## Group buffer pool monitoring with the DISPLAY GROUPBUFFERPOOL command

Use the DISPLAY GROUPBUFFERPOOL command to display information about group buffer pools.

Assume that you want a summary report about group buffer pool zero (GBP0), including all connections to that group buffer pool. Enter the following command:

**GUPI**

-DB1A DISPLAY GROUPBUFFERPOOL(GBP0) CONNLIST(YES)

The following figure shows what the display might look like, assuming that the group buffer pool is duplexed.

```

DSNB750I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 FOLLOWS
DSNB755I -DB1A DB2 GROUP BUFFER POOL STATUS
                CONNECTED = YES
                CURRENT DIRECTORY TO DATA RATIO = 5.4
                PENDING DIRECTORY TO DATA RATIO = 6.0
                CURRENT GBPCACHE ATTRIBUTE = YES
                PENDING GBPCACHE ATTRIBUTE = YES
DSNB756I -DB1A CLASS CASTOUT THRESHOLD = 5%
                GROUP BUFFER POOL CASTOUT THRESHOLD = 30%
                GROUP BUFFER POOL CHECKPOINT INTERVAL = 4 MINUTES
                RECOVERY STATUS = NORMAL
                AUTOMATIC RECOVERY = Y
DSNB757I -DB1A MVS CFRM POLICY STATUS FOR DSNDB0A_GBPO = NORMAL
                MAX SIZE INDICATED IN MVS POLICY = 61440 KB
                DUPLEX INDICATOR IN POLICY = ENABLED
                CURRENT DUPLEXING MODE = DUPLEX
                ALLOCATED = YES
DSNB758I -DB1A ALLOCATED SIZE = 61440 KB
                VOLATILITY STATUS = NON-VOLATILE
                REBUILD STATUS = DUPLEXED
                CFNAME = LF01
                OPERATIONAL CFLEVEL = 5
                ACTUAL CFLEVEL = 7
DSNB759I -DB1A NUMBER OF DIRECTORY ENTRIES = 61394
                NUMBER OF DATA PAGES = 11370
                NUMBER OF CONNECTIONS = 3
DSNB798I -DB1A LAST GROUP BUFFER POOL CHECKPOINT 17:31:23 MAY 9, 2002
                GBP CHECKPOINT RECOVERY LRSN = ACD74C77388C
                STRUCTURE OWNER = DB1A
DSNB799I -DB1A SECONDARY GBP IS ALLOCATED
                ALLOCATED SIZE = 61440 KB
                VOLATILITY STATUS = NON-VOLATILE
                CFNAME = LF01
                OPERATIONAL CFLEVEL = 5
                ACTUAL CFLEVEL = 7
                NUMBER OF DIRECTORY ENTRIES = 61394
                NUMBER OF DATA PAGES = 11370
DSNB766I -DB1A THE CONNLIST REPORT FOLLOWS
DSNB767I -DB1A CONNECTION NAME = DB2_DB1A , CONNECTION STATUS = A
                CONNECTOR'S RELEASE = 6100
DSNB767I -DB1A CONNECTION NAME = DB2_DB2A , CONNECTION STATUS = A
                CONNECTOR'S RELEASE = 6100
DSNB767I -DB1A CONNECTION NAME = DB2_DB3A , CONNECTION STATUS = F
                CONNECTOR'S RELEASE = 6100
DSNB769I -DB1A THE CONNLIST REPORT IS COMPLETE
DSNB790I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSNB9022I -DB1A DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION

```

### GUIP

Figure 65. Summary report with connection list included

You can display detailed statistics using the GDETAIL and MDETAIL keywords.

**Monitoring delete name requests:** Use the DISPLAY GROUPBUFFERPOOL command with the MDETAIL option to determine how many times delete-name requests occur. The following figure shows partial output of the DISPLAY GROUPBUFFERPOOL command which includes delete-name information:

### GUIP

```
-DISPLAY GROUPBUFFERPOOL(GBP29) MDETAIL(*)
```

```

DSNB750I - DISPLAY FOR GROUP BUFFER POOL GBP29 FOLLOWS
      .
      .
      .
DSNB797I - OTHER INTERACTIONS
          REGISTER PAGE                      = 0
          UNREGISTER PAGE                    = 0
          DELETE NAME                        = 0
          READ STORAGE STATISTICS            = 0
          EXPLICIT CROSS INVALIDATIONS       = 0
          ASYNCHRONOUS GBP REQUESTS         = 0
      .
      .
      .
DSNB793I - DELETE NAME LIST                  = 0
          READ CASTOUT STATISTICS            = 0
          DELETE NAME                        = 0
          OTHER ASYNCHRONOUS GBP REQUESTS   = 0
DSNB790I - DISPLAY FOR GROUP BUFFER POOL GBP29 IS COMPLETE
DSN9022I - DSNB1CMD '-DISPLAY GBPOOL' NORMAL COMPLETION

```

### GUIP

Figure 66. DISPLAY GROUPBUFFERPOOL command with MDETAIL option showing delete name information

You can also monitor delete-name requests by using the OMEGAMON Statistics Detail report or IFCID 263 (Page set castout detail).

### Recommendation:

For an easy way to collect interval statistics for performance analysis, create a batch job that issues the following command periodically:

### GUIP

```
-DB1A DISPLAY GROUPBUFFERPOOL(*) GDETAIL(INTERVAL)
```

### GUIP

The first time you run the batch job is the base which purges existing statistics and resets the interval. If you run the job the second time 5 minutes after the first, you can continue running the job every 5 minutes to gather meaningful statistical data on group buffer pool activity.

### Related concepts

“Group buffer pool size is too small” on page 210

“What to look for in a OMEGAMON statistics report” on page 217

### Related reference

 -DISPLAY GROUPBUFFERPOOL (DB2) (DB2 Commands)

### When to use DB2 statistics trace

Use DB2 statistics class 1 to do high-level monitoring of DB2 subsystem activity.

A data section mapped by DSNDQBGL records statistics for a member's use of group buffer pools. The counters are cumulative since the time the member connected to a particular group buffer pool.

Statistics reporting intervals are not synchronized across the members of the data sharing group. For counters that are pertinent to the entire data sharing group, like group buffer statistics, OMEGAMON group-scope statistics reports combine the data of the individual members and present it for the entire group. The member data is apportioned to the same user-specified interval. OMEGAMON presents the synchronized statistics intervals for each member, adds the counters across all members and presents them as statistics on a per-group basis.

Consider also using OMEGAMON to do group-scope exception reporting when a particular counter exceeds a user-specified value.

## Determining the correct size and ratio of group buffer pools

One of the critical tuning factors in a DB2 data sharing configuration is the size of the group buffer pools.

Three aspects of group buffer pool (cache structure) size need to be considered:

- Total structure size

The total structure size of a group buffer pool is specified in the coupling facility policy definition for the cache structure.

- Number of directory entries

A directory entry is used by the coupling facility to determine where to send cross-invalidation signals when a page of data is changed or when that directory entry must be reused. A directory entry contains control information for one database page, no matter in how many places that page is cached. For example, if page P1 is cached in the group buffer pool and in the buffer pools of three members, that page still has only one directory entry.

- Number of data entries

Data entries are the actual places where the data page resides. These are 4 KB, 8 KB, 16 KB, or 32 KB in size (the same size as the data page).

For GBPCACHE NO group buffer pools, no data entries exist.

The number of directory entries and data entries in the coupling facility structure is determined by the size specified in the coupling facility policy and the ratio of directory entries to data pages. The ratio is automatically defined for each group buffer pool at the time that the first member of the group is installed. The default value used is 5 directory entries per data page.

For secondary group buffer pools, the ratio is the same as the ratio used for the primary group buffer pools.

After installation, you can change the ratio with the ALTER GROUPBUFFERPOOL command. However, the change does not take effect until the next time the group buffer pool is allocated.

The following sections describe the symptoms and values that are **not** ideal for best performance, and discuss how you can fix the problems.

### Related concepts

 Storage estimates for data sharing environments (DB2 Installation and Migration)

 Group buffer pool sizes (DB2 Installation and Migration)

### Related information

 IBM eServer zSeries Processor Resource/Systems Manager Planning Guide

### Group buffer pool size is too small

Pages in the group buffer pool need to be refreshed from disk more often because they are not in the group buffer pool.

When the group buffer pool is too small, the following problems can occur:

- The thresholds for changed pages are reached more frequently, causing data to be cast out to disk more often.

If castout cannot keep up with the writes to the group buffer pool, a more serious problem occurs: pages are instead written to the logical page list and are unavailable until they are recovered.

- Many cross-invalidations caused by reusing existing directory entries, which might require refreshing a page from disk later when the page is referenced again.

You can use the GDETAIL option of the DISPLAY GROUPBUFFERPOOL command to gather detailed statistical information about how often data is returned on a read request to the group buffer pool:

#### **GUPI**

```
-DB1A DISPLAY GROUPBUFFERPOOL(GBP0) GDETAIL(*)
```

The following figure shows output similar to the detail portion of the report output.

```

DSNB783I -DB1A CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 May 17, 2002
DSNB784I -DB1A GROUP DETAIL STATISTICS
      READS
      DATA RETURNED A = 3845
DSNB785I -DB1A DATA NOT RETURNED
      DIRECTORY ENTRY EXISTED B = 27
      DIRECTORY ENTRY CREATED C = 28336
      DIRECTORY ENTRY NOT CREATED D = 332, 0

DSNB786I -DB1A WRITES
      CHANGED PAGES = 20909
      CLEAN PAGES = 0
      FAILED DUE TO LACK OF STORAGE E = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB787I -DB1A RECLAIMS
      FOR DIRECTORY ENTRIES F = 18281
      FOR DATA ENTRIES = 47
      CASTOUTS = 16073
DSNB788I -DB1A CROSS INVALIDATIONS
      DUE TO DIRECTORY RECLAIMS G = 4489
      DUE TO WRITES = 3624
      EXPLICIT = 0
DSNB762I -DB1A DUPLEXING STATISTICS FOR GBP0-SEC
      WRITES
      CHANGED PAGES = 20909
      FAILED DUE TO LACK OF STORAGE = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB790I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1A DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION

```

Figure 67. Example output of group detail statistics

#### GUPI

What you need to determine is the *read-hit* percentage. To calculate this value, you need to determine how many of the total number of reads were successful in returning data. Use the following formula:

$$\left( \frac{\mathbf{A}}{\mathbf{A} + \mathbf{B} + \mathbf{C} + \mathbf{D}} \text{ (first number)} \right) \times 100$$

In this example, the calculation is:

$$(3845 / 32540) \times 100 = 11.81\%$$

Data was returned in approximately 12% of the read requests to the group buffer pool. This low percentage of read hits might indicate that the average residency time for a cached page in group buffer pool is too short. You might benefit from altering the group buffer pool to increase the total size.

However, a low percentage of read hits could be caused by other factors:

- A high read-to-write ratio.  
If you are caching only changed pages, not many pages you need will be resident in the group buffer pool.
- Random reference patterns.  
Pages that are frequently referenced are most likely to be resident in the group buffer pool. If the application keeps requesting new pages, any given page is unlikely to be found in the group buffer pool.

To determine whether the low read-hit percentage is a problem, see the field indicated by **B** in the statistics report, shown in the following figure. Figure 72 on page 218

(The same counter also exists in the accounting report.) Ideally, that field contains

GROUP BP4		QUANTITY	/MINUTE	/THREAD	/COMMIT
GROUP BP R/W RATION (%)		50.63	N/A	N/A	N/A
GBP-DEPENDENT GETPAGES		64146.00	99.89	1309.10	0.65
SYN.READS(XI)-DATA RETURNED	<b>A</b>	21079.00	32.83	430.18	0.21
SYN.READS(XI)-NO DATA RETURN	<b>B</b>	0.00	0.00	0.00	0.00
SYN.READS(NF)-DATA RETURNED	<b>C</b>	448.00	0.70	9.14	0.00
SYN.READS(NF)-NO DATA RETURN		238.00	0.37	4.86	0.00
UNREGISTER PAGE	<b>D</b>	3.00	0.00	0.06	0.00
CLEAN PAGES SYNC.WRITTEN		0.00	0.00	0.00	0.00
REG.PAGE LIST (RPL) REQUEST		0.00	0.00	0.00	0.00
NUMBER OF PAGES RETR.FROM GBP		0.00	0.00	0.00	0.00
PGS READ FRM DASD AFTER RPL		N/A	N/A	N/A	N/A
ASYNCH.READ-DATA RETURNED		N/A	N/A	N/A	N/A
PAGES CASTOUT	<b>E</b>	24795.00	38.61	506.02	0.25
UNLOCK CASTOUT	<b>F</b>	3193.00	4.97	65.16	0.03
READ CASTOUT CLASS	<b>G</b>	0.00	0.00	0.00	0.00
READ DIRECTORY INFO	<b>H</b>	0.00	0.00	0.00	0.00
READ STORAGE STATISTICS	<b>I</b>	4.00	0.01	0.08	0.00
REGISTER PAGE	<b>J</b>	0.00	0.00	0.00	0.00
DELETE NAME	<b>K</b>	0.00	0.00	0.00	0.00
ASYNCH GBP REQUESTS	<b>L</b>	87420.00	136.13	1784.08	0.88
EXPLICIT X-INVALIDATIONS	<b>M</b>	0.00	0.00	0.00	0.00
CASTOUT CLASS THRESHOLD	<b>N</b>	0.00	0.02	0.00	0.00
GROUP BP CASTOUT THRESHOLD	<b>O</b>	0.00	0.00	0.00	0.00
GBP CHECKPOINTS TRIGGERED	<b>P</b>	0.00	0.00	0.00	0.00
CASTOUT ENGINE NOT AVAIL.	<b>Q</b>	N/A	N/A	N/A	N/A
WRITE ENGINE NOT AVAILABLE	<b>R</b>	N/A	N/A	N/A	N/A
READ FAILED-NO STORAGE	<b>S</b>	N/A	N/A	N/A	N/A
WRITE FAILED-NO STORAGE	<b>T</b>	0.00	0.00	0.00	0.00
WRITE TO SEC-GBP FAILED		0.00	0.00	0.00	0.00
DELETE NAME LIST SEC-GBP		0.00	0.00	0.00	0.00
DELETE NAME FROM SEC-GBP		0.00	0.00	0.00	0.00
UNLOCK CASTOUT STATS SEC-GBP	<b>U</b>	0.00	0.00	0.00	0.00
ASYNCH SEC-GBP REQUESTS		0.00	0.00	0.00	0.00
WRITE AND REGISTER		42521.00	66.22	867.78	0.43
WRITE AND REGISTER MULT		0.00	0.00	0.00	0.00
CHANGED PGS SYNC.WRTN	<b>V</b>	42522.00	66.22	867.80	0.43
CHANGED PGS ASYNCH.WRTN	<b>W</b>	0.00	0.00	0.00	0.00
PAGES WRITE & REG MULT		0.00	0.00	0.00	0.00
READ FOR CASTOUT		0.00	0.00	0.00	0.00
READ FOR CASTOUT MULT		0.00	0.00	0.00	0.00
WRITE TO SEC-GBP	<b>X</b>	N/A	N/A	N/A	N/A
CLEAN PAGES ASYNCH.WRTN	<b>Y</b>	N/A	N/A	N/A	N/A
CLEAN PGS READ AFT.RPL		N/A	N/A	N/A	N/A
PARTICIPAT.GBP REBUILD	<b>Z</b>				

Figure 68. Portion of OMEGAMON statistics detail report showing GBP activity

0. A non-zero value in the field, in conjunction with a low read hit percentage, can indicate that your group buffer pool is too small.

## Related concepts

“Changing the size of the group buffer pool” on page 221

“Problem: storage shortage in the group buffer pool” on page 114

“Too few data entries” on page 214

“Too few directory entries”

## How to monitor storage of the group buffer pool:

By monitoring the storage use of the group buffer pool, you can avoid data outages caused by a serious lack of storage in the group buffer pool.

### GUIP

**Recommendation:** Issue periodic DISPLAY GROUPBUFFERPOOL commands with the GDETAIL option. The GDETAIL statistics show a snapshot value of the number of changed pages in the group buffer pool. Ensure that this snapshot value ( **C** ) does not rise significantly above the group buffer pool castout threshold ( **B** × **A** ). The following figure highlights the key fields from the report.

```

< GUIP
DSNB756I -DB1A CLASS CASTOUT THRESHOLD = 5%
           A GROUP BUFFER POOL CASTOUT THRESHOLD = 30%
           GROUP BUFFER POOL CHECKPOINT INTERVAL = 4 MINUTES
           RECOVERY STATUS = NORMAL
           AUTOMATIC RECOVERY = Y
DSNB759I -DB1A NUMBER OF DIRECTORY ENTRIES = 61394
           B NUMBER OF DATA PAGES = 11370
           NUMBER OF CONNECTIONS = 3
DSNB783I -DB1A CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 Mar 17, 2002
DSNB784I -DB1A GROUP DETAIL STATISTICS
:
:
DSNB786I -DB1A WRITES
           CHANGED PAGES = 1576
           CLEAN PAGES = 0
           FAILED DUE TO LACK OF STORAGE = 0
           C CHANGED PAGES SNAPSHOT VALUE = 311
:
:
```

Figure 69. Partial output of DISPLAY GROUPBUFFERPOOL command. Ensure that the SNAPSHOT value does not rise significantly above the group buffer pool castout threshold.

## Too few directory entries

When existing directory entries are being reclaimed to handle new work, cross-invalidation must occur for all of the members that have the particular data pages in their buffer pools, even when the data has not actually changed.

### GUIP

For example, in the following figure, **F** indicates that there have been 18 281 directory reclaims. Figure 67 on page 211

```

DSNB783I -DB1A CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 May 17, 2002
DSNB784I -DB1A GROUP DETAIL STATISTICS
      READS
      DATA RETURNED A = 3845
DSNB785I -DB1A DATA NOT RETURNED
      DIRECTORY ENTRY EXISTED B = 27
      DIRECTORY ENTRY CREATED C = 28336
      DIRECTORY ENTRY NOT CREATED D = 332, 0

DSNB786I -DB1A WRITES
      CHANGED PAGES = 20909
      CLEAN PAGES = 0
      FAILED DUE TO LACK OF STORAGE E = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB787I -DB1A RECLAIMS
      FOR DIRECTORY ENTRIES F = 18281
      FOR DATA ENTRIES = 47
      CASTOUTS = 16073
DSNB788I -DB1A CROSS INVALIDATIONS
      DUE TO DIRECTORY RECLAIMS G = 4489
      DUE TO WRITES = 3624
      EXPLICIT = 0
DSNB762I -DB1A DUPLEXING STATISTICS FOR GBP0-SEC
      WRITES
      CHANGED PAGES = 20909
      FAILED DUE TO LACK OF STORAGE = 8
      CHANGED PAGES SNAPSHOT VALUE = 974
DSNB790I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1A DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION

```

Figure 70. Example output of group detail statistics

**G** indicates that, because of those reclaims, 4489 cross-invalidations occurred. The pages in those members' buffer pools need to be refreshed when next needed, probably from disk, which can degrade system performance.

If there is a high value in **G**, check the group buffer pool hit percentage to see if the lack of directory entries might be causing an excessive number of reads from disk.

You can also check the “SYNCHRONOUS READS DUE TO BUFFER INVALIDATION” counters shown in the member detail report.

You can increase the number of directory entries in the group buffer pool in one of two ways:

- Increase the total size of the group buffer pool.
- Use the ALTER GROUPBUFFERPOOL command to adjust the ratio in favor of directory entries.

#### **GUPI**

#### **Related concepts**

“Changing the size of the group buffer pool” on page 221

“Group buffer pool size is too small” on page 210

#### **Related tasks**

“Changing the ratio of directory to data entries” on page 223

#### **Too few data entries**

If a group buffer pool does not have enough data entries, castout to disk occurs more frequently.

**GUPI** You can see the number of pages cast out by using the GDETAIL option of the DISPLAY GROUPBUFFERPOOL command.

A more serious data entry shortage is indicated by field **E** in the DISPLAY GROUPBUFFERPOOL GDETAIL report shown in the following figure. Figure 67 on page 211

```
DSNB783I -DB1A CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 May 17, 2002
DSNB784I -DB1A GROUP DETAIL STATISTICS
          READS
          DATA RETURNED A                = 3845
DSNB785I -DB1A  DATA NOT RETURNED
          DIRECTORY ENTRY EXISTED B        = 27
          DIRECTORY ENTRY CREATED C       = 28336
          DIRECTORY ENTRY NOT CREATED D    = 332, 0

DSNB786I -DB1A  WRITES
          CHANGED PAGES                = 20909
          CLEAN PAGES                   = 0
          FAILED DUE TO LACK OF STORAGE E = 8
          CHANGED PAGES SNAPSHOT VALUE = 974
DSNB787I -DB1A  RECLAIMS
          FOR DIRECTORY ENTRIES F         = 18281
          FOR DATA ENTRIES            = 47
          CASTOUTS                     = 16073
DSNB788I -DB1A  CROSS INVALIDATIONS
          DUE TO DIRECTORY RECLAIMS G     = 4489
          DUE TO WRITES                 = 3624
          EXPLICIT                      = 0
DSNB762I -DB1A  DUPLEXING STATISTICS FOR GBP0-SEC
          WRITES
          CHANGED PAGES                = 20909
          FAILED DUE TO LACK OF STORAGE = 8
          CHANGED PAGES SNAPSHOT VALUE = 974
DSNB790I -DB1A DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1A DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION
```

Figure 71. Example output of group detail statistics

A value in this field indicates that the data page resources of the coupling facility are being consumed faster than the DB2 castout processes can free them.

You can increase the number of data entries in the group buffer pool in one of two ways:

- Increase the total size of the group buffer pool.
- Use the ALTER GROUPBUFFERPOOL command to adjust the ratio in favor of data entries.

**GUPI**

### Related concepts

“Changing the size of the group buffer pool” on page 221

### Related tasks

“Changing the ratio of directory to data entries” on page 223

### Auto Alter capabilities

You can avoid cross invalidations due to directory reclaims and writes failed due to lack of storage by using a z/OS capability called Auto Alter.

This capability is initiated by the z/OS structure full monitoring that occurs regularly for each structure. Structure full monitoring adds support for the monitoring of objects within a coupling facility structure. This type of monitoring determines the level of usage for objects within a coupling facility, and issues a warning message if a structure full condition is imminent. Doing this allows tuning actions to avoid a structure full condition. Structure full monitoring occurs every few seconds.

Auto Alter has algorithms that can request the coupling facility to dynamically increase or decrease the number of entries and/or data page elements to avoid structure full conditions. It can increase or decrease the size of the structure if necessary.

**Note:** The design point for Auto Alter is for gradual growth, not to handle spikes in the workload.

Auto Alter applies to all coupling facility structures. For the lock structure, it can only dynamically increase the RLE portion of the structure. It can increase the size of the SCA. Its main value in DB2; however, is for the group buffer pools. Guessing at a proper directory to data ratio is almost impossible. A group buffer pool usually needs more than a 5:1 ratio. Static values might or might not be accurate and, over time, they could change as the workload changes. Auto Alter can pinpoint precisely the ratio needed at any given time and change the ratios dynamically. By comparison, if the ratio is changed by operator command, the structure must be manually rebuilt, which causes a slight disruption.

When either the directory entries or data pages exceed the FULLTHRESHOLD, XES will increase the size of the structure and will increase the component in short supply while decreasing the other one. For DB2 group buffer pools, the shortages usually occur for the data elements, not directory entries.

Note that if there is general storage stress on the coupling facility (less than 10% free storage), XES can decrease those structures with ALLOWAUTOALTER(YES). XES will never decrease them below MINSIZE (defaulted to 75% of INITSIZE).

Even though XES is directing the coupling facility to make the changes, the DB2 -DISPLAY GROUPBUFFERPOOL command reflects the current directory to data ratios. The total numbers of directory entries and data elements are found in the coupling facility usage summary section of the RMF coupling facility activity report.

These are the main Auto Alter capabilities:

- Auto Alter supports autonomic tuning, because you set the sizes and Auto Alter does the rest.
- Auto Alter alters ratios in the coupling facility without rebuilding the structure (versus the DB2 -ALTER GROUPBUFFERPOOL command that requires a rebuild).
- Auto Alter builds a better directory to data ratio than manual tuning (up to 40:1). Reclaim avoidance adjusts dynamically to changes with workload.
- Auto Alter can avoid the error: Writes failed due to lack of storage.
- Auto Alter allows database administrators to change local buffer pool sizes or add a member without needing a CFRM policy change. It adjusts to gradual growth in the workload.

The way you should have sized your group buffer pools is for proactive tuning, where there is enough storage in each structure to allow for growth, either through increases in local buffer pools or by adding another DB2 member. The goal is to allow group buffer pool tuning to be ignored for a certain amount of time, perhaps six months to a year. This approach is safe when there is sufficient storage available in the coupling facilities.

**Important:** Your z/OS systems teams must allow enough white space in each coupling facility to allocate all of the structures in both coupling facilities. In the rare event that a coupling facility fails, for whatever reason, the structures should rebuild quickly in the remaining coupling facility. This is specified in the STRUCTURE statement via the REBUILDPERCENT value of 1, and the PREF keyword, where at least one other coupling facility is referenced.

Once you have sized your structures, it is easy to start using Auto Alter. You can enter the parameter ALLOWAUTOALT(YES) with each STRUCTURE statement for which you want to allow structure alteration. This example shows the GBP0 definition:

```
STRUCTURE NAME=groupname_GBP0
INITSIZE=16000
SIZE=32000
ALLOWAUTOALT(YES)
FULLTHRESHOLD=80
MINSIZE=16000
PREFLIST=(CF2,CF1)
DUPLEX(ENABLED)
REBUILDPERCENT(1)
```

You do not want the group buffer pool to fall below the INITSIZE you have specified, so you can code MINSIZE with the same value as INITSIZE. You also set the FULLTHRESHOLD to 80% (also the default).

#### Related tasks

 [Allowing a Structure to Be Altered Automatically \(z/OS MVS Setting Up a Sysplex\)](#)

### What to look for in a OMEGAMON statistics report

You can use OMEGAMON statistics reports to understand group buffer pool activity.

Refer to the OMEGAMON statistics detail report shown in the following figure. The fields from that report are used to explain some of the activity that takes place for cross-invalidation and refresh of buffers. Some of the information is the same as that described by the DISPLAY GROUPBUFFERPOOL output, which was covered earlier in this section.

GROUP BP4		QUANTITY	/MINUTE	/THREAD	/COMMIT
GROUP BP R/W RATION (%)		50.63	N/A	N/A	N/A
GBP-DEPENDENT GETPAGES		64146.00	99.89	1309.10	0.65
SYN.READS(XI)-DATA RETURNED	<b>A</b>	21079.00	32.83	430.18	0.21
SYN.READS(XI)-NO DATA RETURN	<b>B</b>	0.00	0.00	0.00	0.00
SYN.READS(NF)-DATA RETURNED	<b>C</b>	448.00	0.70	9.14	0.00
SYN.READS(NF)-NO DATA RETURN		238.00	0.37	4.86	0.00
UNREGISTER PAGE	<b>D</b>	3.00	0.00	0.06	0.00
CLEAN PAGES SYNC.WRITTEN		0.00	0.00	0.00	0.00
REG.PAGE LIST (RPL) REQUEST		0.00	0.00	0.00	0.00
NUMBER OF PAGES RETR.FROM GBP		0.00	0.00	0.00	0.00
PGS READ FRM DASD AFTER RPL		N/A	N/A	N/A	N/A
ASYNCH.READ-DATA RETURNED		N/A	N/A	N/A	N/A
PAGES CASTOUT	<b>E</b>	24795.00	38.61	506.02	0.25
UNLOCK CASTOUT	<b>F</b>	3193.00	4.97	65.16	0.03
READ CASTOUT CLASS	<b>G</b>	0.00	0.00	0.00	0.00
READ DIRECTORY INFO	<b>H</b>	0.00	0.00	0.00	0.00
READ STORAGE STATISTICS	<b>I</b>	4.00	0.01	0.08	0.00
REGISTER PAGE	<b>J</b>	0.00	0.00	0.00	0.00
DELETE NAME	<b>K</b>	0.00	0.00	0.00	0.00
ASYNCH GBP REQUESTS	<b>L</b>	87420.00	136.13	1784.08	0.88
EXPLICIT X-INVALIDATIONS	<b>M</b>	0.00	0.00	0.00	0.00
CASTOUT CLASS THRESHOLD	<b>N</b>	0.00	0.02	0.00	0.00
GROUP BP CASTOUT THRESHOLD	<b>O</b>	0.00	0.00	0.00	0.00
GBP CHECKPOINTS TRIGGERED	<b>P</b>	0.00	0.00	0.00	0.00
CASTOUT ENGINE NOT AVAIL.	<b>Q</b>	N/A	N/A	N/A	N/A
WRITE ENGINE NOT AVAILABLE	<b>R</b>	N/A	N/A	N/A	N/A
READ FAILED-NO STORAGE	<b>S</b>	N/A	N/A	N/A	N/A
WRITE FAILED-NO STORAGE	<b>T</b>	0.00	0.00	0.00	0.00
WRITE TO SEC-GBP FAILED		0.00	0.00	0.00	0.00
DELETE NAME LIST SEC-GBP		0.00	0.00	0.00	0.00
DELETE NAME FROM SEC-GBP		0.00	0.00	0.00	0.00
UNLOCK CASTOUT STATS SEC-GBP	<b>U</b>	0.00	0.00	0.00	0.00
ASYNCH SEC-GBP REQUESTS		0.00	0.00	0.00	0.00
WRITE AND REGISTER		42521.00	66.22	867.78	0.43
WRITE AND REGISTER MULT		0.00	0.00	0.00	0.00
CHANGED PGS SYNC.WRTN	<b>V</b>	42522.00	66.22	867.80	0.43
CHANGED PGS ASYNCH.WRTN	<b>W</b>	0.00	0.00	0.00	0.00
PAGES WRITE & REG MULT		0.00	0.00	0.00	0.00
READ FOR CASTOUT		0.00	0.00	0.00	0.00
READ FOR CASTOUT MULT		0.00	0.00	0.00	0.00
WRITE TO SEC-GBP	<b>X</b>	N/A	N/A	N/A	N/A
CLEAN PAGES ASYNCH.WRTN	<b>Y</b>	N/A	N/A	N/A	N/A
CLEAN PGS READ AFT.RPL		N/A	N/A	N/A	N/A
PARTICIPAT.GBP REBUILD	<b>Z</b>				

Figure 72. Portion of OMEGAMON statistics detail report showing GBP activity

### Explanation of fields

- A** The number of requests made to read a page from the group buffer pool because the page was invalidated in the member's buffer pool. The member found the required page in the group buffer pool.
- B** The number of requests to read a page from the group buffer pool that were required because the page was invalidated in the member's buffer pool. The member did not find the data in the group buffer pool and had to retrieve the page from DASD.

- C** The number of requests made to read a page from the group buffer pool because the page was not in the member's buffer pool. The member found the page in the group buffer pool.
- D** The number of times DB2 unregistered interest for a single page. This happens when DB2 steals pages from the member's buffer pool that belong to GBP-dependent page sets or partitions.
- E** The number of data pages that were cast out of the member's group buffer pool. Castout to a page set or partition is done by the castout owner of the page set or partition. This is normally the DB2 subsystem that had the first update intent on the page set or partition.
- F** The number of times DB2 issued an unlock request to the coupling facility for completed castout I/Os. When pages are cast out to DASD, they are locked for castout in the coupling facility. This castout lock is not an IRLM lock; its function is to ensure that only one system can cast out a given page at a time.
- G** The number of requests issued by the group buffer pool to determine which pages, from a particular page set or partition, must be cast out because they are cached as changed pages. This request is issued either by the page set or partition castout owner, or, when the group buffer pool castout threshold is reached by the group buffer pool structure owner.
- H** The number of requests issued by the group buffer pool structure owner to read the directory entries of all changed pages in the group buffer pool. This request is issued at group buffer pool checkpoints to record the oldest recovery log record sequence number (LRSN). It is used as a basis for recovery if the group buffer pool fails. Such requests might have to be issued several times for each group buffer pool checkpoint to read the directory entries for all changed pages.
- I** The number of times DB2 requested statistics information from the group buffer pool. It is issued by the group buffer pool structure owner at timed intervals to determine whether the group buffer pool castout threshold (GBPOOLT) has been reached.
- J** The number of times DB2 registered interest in a single page. These are "register-only" requests, which means that DB2 is not requesting any data back from the request. This request is made only to create a directory entry for the page to be used for cross-invalidation when the page set or partition P-Lock is downgraded from S to IS mode, or from SIX to IX mode.
- K** The number of requests made by DB2 to delete directory and data entries associated with a particular page set or partition from the group buffer pool. DB2 issues this request when it changes a page set or partition from GBP-dependent to non GBP-dependent. DB2 also issues this request for objects that are defined with GBPCACHE ALL when those objects are first opened.
- L** The number of IXLCACHE invocations for the primary group buffer pool.
- M** The number of times an explicit coupling facility cross-invalidation request was issued.
- N** The number of times group buffer pool castout was initiated because the group buffer pool class castout threshold was detected.
- O** The number of times a group buffer pool castout was initiated because the group buffer pool castout threshold was detected.

- P** The number of group buffer pool checkpoints triggered by this member.
- Q** The number of times a castout engine was not available.
- R** The number of times a coupling facility write engine was not available for coupling facility writes. This field is not applicable to versions higher than DB2 V7.
- S** The number of coupling facility read requests that did not complete due to a lack of coupling facility storage resources. This field is not applicable for versions higher than DB2 V7.
- T** The number of coupling facility write requests that could not complete due to a lack of coupling facility storage resources.
- U** The number of coupling facility requests to read the castout statistics for the secondary group buffer pool. These requests are issued by the group buffer pool structure owner to check for orphaned data entries in the secondary group buffer pool.
- V** The number of changed pages written synchronously to the group buffer pool. Pages are written with Write and Register (WAR) requests or Write and Register Multiple (WARM) requests. At commit time changed pages are forced from the member's virtual buffer pool to the coupling facility.
- W** The number of changed pages written asynchronously to the group buffer pool. Pages are written in response to Write and Register (WAR) and Write and Register Multiple (WARM) requests. Changed pages can be written from the member's virtual buffer pool to the group coupling facility before the application commits. This happens when, for example, a local buffer pool threshold is reached, or when P-Lock negotiation forces the pages on the vertical deferred write queue to be written to the group buffer pool.
- X** The number of coupling facility requests to write changed pages to the secondary group buffer pool for duplexing. This field is not applicable for versions higher than DB2 V7.
- Y** The number of clean pages that were asynchronously written to the group buffer pool from the virtual pool. This field is not applicable for versions higher than DB2 V7.
- Z** The number of times this member participated in a group buffer pool rebuild. This includes normal rebuilds and rebuilds to establish duplexing. This field is applicable for DB2 Versions 6 and 7 only.

## Changing group buffer pools

You can change group buffer pool attributes such as the castout threshold values, checkpoint frequency, and size.

### Related tasks

“Shutting down the coupling facility” on page 135

“Starting duplexing for a structure” on page 133

“Stopping duplexing for a structure” on page 134

### Changing the castout threshold values

Use the ALTER GROUPBUFFERPOOL command to change the group buffer pool castout thresholds.

The following command, for example, changes the class castout threshold to 15% and the group buffer pool threshold to 55%:

**GUIP**

```
-DB1A ALTER GROUPBUFFERPOOL(GBP1) CLASST(15) GBPOOLT(55)
```

**GUIP**

These changes take effect immediately.

**Changing the checkpoint frequency**

Use the ALTER GROUPBUFFERPOOL command to change the checkpoint frequency.

For example, to indicate that you want group buffer pool checkpoints to occur every three minutes, enter the following command:

**GUIP**

```
-DB1A ALTER GROUPBUFFERPOOL(GBP1) GBPCHKPT(3)
```

**GUIP**

This change takes effect immediately.

**Changing the size of the group buffer pool**

You can use two methods to change the size of the group buffer pool.

The method you choose depends on what level of coupling facility the group buffer pool is allocated and whether the group buffer pool is already allocated at the maximum size. For a duplexed group buffer pool, you are changing the size of both the primary and secondary structure with a single command.

**Dynamic method**

If all of the following conditions are true:

- The group buffer pool is allocated in a coupling facility with CFLEVEL greater than zero.
- The currently allocated size of the structure is less than the maximum size as defined in the SIZE parameter of the CFRM policy.

Then you can enter the following command (this example assumes the group name is DSNDB0A):

```
SETXCF START,ALTER,STRNAME=DSNDB0A_GBPn,SIZE=new size
```

This example assumes that *newsize* is less than or equal to the maximum size that is defined by the CFRM policy for the group buffer pool.

If the maximum size (SIZE in the CFRM policy) is still not big enough, you must use the method described below in Static Method.

Assume a DISPLAY GROUPBUFFERPOOL command produced the following output:

**GUIP**

```

:
DSNB757I -DB1A MVS CFRM POLICY STATUS FOR DSNDB0A_GBP1 = NORMAL
          MAX SIZE INDICATED IN POLICY                 = 4096 KB
          ALLOCATED                                     = YES
DSNB758I -DB1A ALLOCATED SIZE                           = 1024 KB
          VOLATILITY STATUS                             = VOLATILE
          REBUILD STATUS                               = NONE
          DUPLEXING STATUS                             = SIMPLEXED
          CFNAME                                        = LF01
          OPERATIONAL CFLEVEL                          = 5
          ACTUAL CFLEVEL                               = 7
DSNB759I -DB1A NUMBER OF DIRECTORY ENTRIES             = 924
          NUMBER OF DATA PAGES                       = 180
          NUMBER OF CONNECTIONS                       = 2
:

```

#### GUPI

And then you enter the following z/OS command to increase the size:

```
SETXCF START,ALTER,STRNM=DSNDB0A_GBP1,SIZE=1536
```

The DISPLAY GROUPBUFFERPOOL command output might look similar to the following after you alter the size:

#### GUPI

```

:
DSNB757I -DB1A MVS CFRM POLICY STATUS FOR DSNDB0A_GBP1 = NORMAL
          MAX SIZE INDICATED IN POLICY                 = 4096 KB
          ALLOCATED                                     = YES
DSNB758I -DB1A ALLOCATED SIZE                           = 1536 KB
          VOLATILITY STATUS                             = VOLATILE
          REBUILD STATUS                               = NONE
          DUPLEXING STATUS                             = SIMPLEXED
          CFNAME                                        = LF01
          OPERATIONAL CFLEVEL                          = 5
          ACTUAL CFLEVEL                               = 7
DSNB759I -DB1A NUMBER OF DIRECTORY ENTRIES             = 1426
          NUMBER OF DATA PAGES                       = 284
          NUMBER OF CONNECTIONS                       = 2
:

```

#### GUPI

Notice that the allocated size, the numbers of directory entries, and the number of data pages has increased. The existing ratio is maintained.

### Static method

If **any** of the following conditions are true:

- The group buffer pool is allocated in a coupling facility at CFLEVEL=0.
- The allocated size of the structure is already at the maximum size defined by the SIZE parameter of the CFRM policy.

You must use the following procedure. Because the group buffer pool must be rebuilt, use this procedure when there is little activity in the group.

1. Increase the storage for the group buffer pool in the CFRM policy, and use DUPLEX(ENABLED).

2. Use the following z/OS command to start the updated policy:  
`SETXCF START,POLICY,TYPE=CFRM,POLNAME=polycname`
3. Use the following command to stop duplexing:  
`SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=OLD`
4. Use the following command to rebuild the group buffer pool:  
`SETXCF START,REBUILD,STRNAME=strname`
5. Use the DISPLAY GROUPBUFFERPOOL command to determine whether the rebuild caused duplexing to restart.
6. If duplexing did not restart on its own, use the following command to restart it:  
`SETXCF START,REBUILD,DUPLEX,STRNAME=strname`

## Changing the ratio of directory to data entries

To change the ratio of directory to data entries, you must use the ALTER GROUPBUFFERPOOL command.

For example, if the current ratio is 5 (that is, there are five directory entries to every one data page), you can use an ALTER GROUPBUFFERPOOL command, similar to the one shown here, to increase the ratio to 7:

**GUIP**

```
-DB1A ALTER GROUPBUFFERPOOL (GBP0) RATIO (7)
```

**GUIP**

For the change to take effect, you must rebuild the group buffer pool by using the SETXCF START, REBUILD command.

For duplexed group buffer pools, you must stop duplexing before you can rebuild the group buffer pool. Use the following procedure to rebuild duplexed group buffer pools:

1. Stop duplexing to revert the group buffer pool to simplex mode.
2. Alter the group buffer pool ratio and issue the SETXCF START,REBUILD,STRNAME=*strname* command.
3. Start duplexing.

### Related tasks

“Starting duplexing for a structure” on page 133

“Stopping duplexing for a structure” on page 134

## Changing the GBPCACHE attribute

To change the GBPCACHE option, use the ALTER GROUPBUFFERPOOL command.

For the change to take effect, you must rebuild the group buffer pool by using the SETXCF START,REBUILD,STRNAME=*strname* command.

DB2 does not let you duplex a GBPCACHE(NO) group buffer pool. If the group buffer pool is currently GBPCACHE(YES) but the NO attribute is pending, DB2 ignores the pending GBPCACHE(NO) attribute if you start a duplexing rebuild.

---

## Access path selection in a data sharing group

The way that a data sharing member is configured affects access path selection. Use the EXPLAIN statement to obtain information about access path selection.

### Effect of member configuration on access path selection

Because plans and packages are bound on individual members in the group, the way a member is configured influences the access path chosen for statements in that plan or package.

For example, it is possible to have different buffer pool sizes and different RID (record identifier) pool sizes on each member. It is also possible that members are on different CPC models.

When you bind your application from one of the members, DB2 chooses the best access path, given the catalog statistics, CPC model and buffer pool sizes, among other things. Suppose, though, that the selected access path is optimal for the one member, but is a relatively poor choice for a different member in the same group. Because the group shares the catalog and directory, the same plan (and hence the same access paths) are used regardless of member, after the application is bound.

*Where to bind in a mixed data sharing configuration:* If your data sharing group consists of mixed CPC models, be aware that the speed of a central processor might change your access path. This effect is more likely to occur with long-running queries than with fast-running transactions.

*Automatic rebind:* The access path can change if automatic rebind occurs while the application is executing on a different member than the one on which the original bind occurred.

### How EXPLAIN works in a data sharing group

EXPLAIN informs you about access paths that DB2 chooses.

PSPI

Because EXPLAIN can be run on one member and a plan can be bound and executed on other members in a data sharing group, it is important to know which member performed the EXPLAIN. The PLAN\_TABLE's GROUP\_MEMBER column contains the name of the member that performed the EXPLAIN. The column is blank if the member was not in a data sharing environment when the EXPLAIN

was performed. PSPI

---

## How DB2 maintains in-memory statistics in data sharing

To enable automation of utility job scheduling, DB2 collects statistics for use by users or automated task schedulers to determine which objects might require REORG, RUNSTATS, or COPY.

These statistics are collected in real time and periodically written to DB2 tables.

Statistics are maintained for each partition if the page set is partitioned. To externalize the statistics from in-memory to the real-time statistics tables, DB2 examines the in-memory statistics, calculates the new totals, updates the real-time

statistics tables, and then resets the in-memory statistics. This process is an asynchronous task, managed by the statistics manager.

In DB2 data sharing, each member updates their statistics in a serial process. Each member reads the target row from the statistics table, and while holding a lock, aggregates their in-memory statistics and then updates the statistics tables with the new totals.

Utilities that reset page sets to empty can invalidate the in-memory statistics of other members. COPY and RUNSTATS must externalize other members' statistics before running. The member that is running the utility notifies the other members, and the in-memory statistics are either invalidated or externalized. If the notify process fails, the running utility will not fail. The appropriate timestamp (ReorgLastTime, StatsLastTime, or CopyLastTime) is set to NULL to indicate that the table statistics are unknown.

### **Related concepts**

 [Real-time statistics \(DB2 Performance\)](#)



---

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

Many information resources are available to help you use DB2 for z/OS and many related products. A large amount of technical information about IBM products is now available online in information centers or on library websites.

**Disclaimer:** Any web addresses that are included here are accurate at the time this information is being published. However, web addresses sometimes change. If you visit a web address that is listed here but that is no longer valid, you can try to find the current web address for the product information that you are looking for at either of the following sites:

- <http://www.ibm.com/support/publications/us/library/index.shtml>, which lists the IBM information centers that are available for various IBM products
- <http://www.ibm.com/shop/publications/order>, which is the IBM Publications Center, where you can download online PDF books or order printed books for various IBM products

### DB2 for z/OS product information

The primary place to find and use information about DB2 for z/OS is the Information Management Software for z/OS Solutions Information Center (<http://publib.boulder.ibm.com/infocenter/imzic>), which also contains information about IMS, QMF™, and many DB2 and IMS Tools products. This information center is also available as an installable information center that can run on a local system or on an intranet server. You can order the Information Management for z/OS Solutions Information Center DVD (SK5T-7377) for a low cost from the IBM Publications Center (<http://www.ibm.com/shop/publications/order>).

The majority of the DB2 for z/OS information in this information center is also available in the books that are identified in the following table. You can access these books at the DB2 for z/OS library website (<http://www.ibm.com/software/data/db2/zos/library.html>) or at the IBM Publications Center (<http://www.ibm.com/shop/publications/order>).

Table 34. DB2 10 for z/OS book titles

Title	Publication number	Available in information center	Available in PDF	Available in printed format
<i>DB2 10 for z/OS Administration Guide</i>	SC19-2968	X	X	
<i>DB2 10 for z/OS Application Programming &amp; SQL Guide</i>	SC19-2969	X	X	
<i>DB2 10 for z/OS Application Programming Guide and Reference for Java</i>	SC19-2970	X	X	
<i>DB2 10 for z/OS Codes</i>	GC19-2971	X	X	
<i>DB2 10 for z/OS Command Reference</i>	SC19-2972	X	X	
<i>DB2 10 for z/OS Data Sharing: Planning and Administration</i>	SC19-2973	X	X	
<i>DB2 10 for z/OS Diagnosis Guide and Reference</i>	LY37-3220		X	X

Table 34. DB2 10 for z/OS book titles (continued)

Title	Publication number	Available in information center	Available in PDF	Available in printed format
<i>DB2 10 for z/OS Installation and Migration Guide</i>	GC19-2974	X	X	
<i>DB2 10 for z/OS Internationalization Guide</i>	SC19-2975	X	X	
<i>DB2 10 for z/OS Introduction to DB2</i>	SC19-2976	X	X	
<i>DB2 10 for z/OS Licensed Program Specifications</i>	GC19-2977		X	X
<i>DB2 10 for z/OS Managing Performance</i>	SC19-2978	X	X	
<i>DB2 10 for z/OS Messages</i>	GC19-2979	X	X	
<i>DB2 10 for z/OS ODBC Guide and Reference</i>	SC19-2980	X	X	
<i>DB2 10 for z/OS Program Directory</i>	GI10-8829		X	X
<i>DB2 10 for z/OS pureXML Guide</i>	SC19-2981	X	X	
<i>DB2 10 for z/OS RACF Access Control Module Guide</i>	SC19-2982	X	X	
<i>DB2 10 for z/OS SQL Reference</i>	SC19-2983	X	X	
<i>DB2 10 for z/OS Utility Guide and Reference</i>	SC19-2984	X	X	
<i>DB2 10 for z/OS What's New?</i>	GC19-2985	X	X	
<i>IRLM Messages and Codes for IMS and DB2 for z/OS</i>	GC19-2666	X	X	

**Note:**

1. *DB2 10 for z/OS Diagnosis Guide and Reference* is available in PDF format on the DB2 10 for z/OS Licensed Library Collection kit, LK5T-7390. You can order this Licensed Library Collection kit on the IBM Publications Center site (<http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss>). This book is also available in online format in DB2 data set DSNA10.SDSNIVPD(DSNDR).

## Information resources for related products

In the following table, related product names are listed in alphabetic order, and the associated web addresses of product information centers or library web pages are indicated.

Table 35. Related product information resource locations

Related product	Information resources
C/C++ for z/OS	Library website: <a href="http://www.ibm.com/software/awdtools/czos/library/">http://www.ibm.com/software/awdtools/czos/library/</a>  This product is now called z/OS XL C/C++.
CICS Transaction Server for z/OS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp">http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp</a>
COBOL	Information center: <a href="http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp">http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp</a>  This product is now called Enterprise COBOL for z/OS.
DB2 Connect	Information center: <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp">http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp</a>  This resource is for DB2 Connect 9.
DB2 Database for Linux, UNIX, and Windows	Information center: <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp">http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp</a>  This resource is for DB2 9 for Linux, UNIX, and Windows.

Table 35. Related product information resource locations (continued)

Related product	Information resources
DB2 Query Management Facility™	Information center: <a href="http://publib.boulder.ibm.com/infocenter/imzic">http://publib.boulder.ibm.com/infocenter/imzic</a>
DB2 Server for VSE & VM	Product website: <a href="http://www.ibm.com/software/data/db2/vse-vm/">http://www.ibm.com/software/data/db2/vse-vm/</a>
DB2 Tools	<p>One of the following locations:</p> <ul style="list-style-type: none"> <li>• Information center: <a href="http://publib.boulder.ibm.com/infocenter/imzic">http://publib.boulder.ibm.com/infocenter/imzic</a></li> <li>• Library website: <a href="http://www.ibm.com/software/data/db2imstools/library.html">http://www.ibm.com/software/data/db2imstools/library.html</a></li> </ul> <p>These resources include information about the following products and others:</p> <ul style="list-style-type: none"> <li>• DB2 Administration Tool</li> <li>• DB2 Automation Tool</li> <li>• DB2 Log Analysis Tool</li> <li>• DB2 Object Restore Tool</li> <li>• DB2 Query Management Facility</li> <li>• DB2 SQL Performance Analyzer</li> </ul>
DB2 Universal Database™ for iSeries®	Information center: <a href="http://www.ibm.com/systems/i/infocenter/">http://www.ibm.com/systems/i/infocenter/</a>
Debug Tool for z/OS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp">http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp</a>
Enterprise COBOL for z/OS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp">http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp</a>
Enterprise PL/I for z/OS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp">http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp</a>
InfoSphere™ Replication Server for z/OS	<p>Information center: <a href="http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.swg.im.iis.db.prod.repl.nav.doc/dochome/iiryrcnav_dochome.html">http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.swg.im.iis.db.prod.repl.nav.doc/dochome/iiryrcnav_dochome.html</a></p> <p>This product was also known as DB2 DataPropagator, DB2 Information Integrator Replication Edition for z/OS, and WebSphere Replication Server for z/OS.</p>
IMS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/imzic">http://publib.boulder.ibm.com/infocenter/imzic</a>
IMS Tools	<p>One of the following locations:</p> <ul style="list-style-type: none"> <li>• Information center: <a href="http://publib.boulder.ibm.com/infocenter/imzic">http://publib.boulder.ibm.com/infocenter/imzic</a></li> <li>• Library website: <a href="http://www.ibm.com/software/data/db2imstools/library.html">http://www.ibm.com/software/data/db2imstools/library.html</a></li> </ul> <p>These resources have information about the following products and others:</p> <ul style="list-style-type: none"> <li>• IMS Batch Terminal Simulator for z/OS</li> <li>• IMS Connect</li> <li>• IMS HALDB Conversion and Maintenance Aid</li> <li>• IMS High Performance Utility products</li> <li>• IMS DataPropagator</li> <li>• IMS Online Reorganization Facility</li> <li>• IMS Performance Analyzer</li> </ul>
Integrated Data Management products	<p>Information center: <a href="http://publib.boulder.ibm.com/infocenter/idm/v2r2/index.jsp">http://publib.boulder.ibm.com/infocenter/idm/v2r2/index.jsp</a></p> <p>This information center has information about the following products and others:</p> <ul style="list-style-type: none"> <li>• IBM Data Studio</li> <li>• InfoSphere Data Architect</li> <li>• InfoSphere Warehouse</li> <li>• Optim™ Database Administrator</li> <li>• Optim Development Studio</li> <li>• Optim Query Tuner</li> </ul>

Table 35. Related product information resource locations (continued)

Related product	Information resources
PL/I	Information center: <a href="http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp">http://publib.boulder.ibm.com/infocenter/pdthelp/v1r1/index.jsp</a> This product is now called Enterprise PL/I for z/OS.
System z <sup>®</sup>	<a href="http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp">http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp</a>
Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS	Information center: <a href="http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.omegamon.xe_db2.doc/ko2welcome_pe.htm">http://publib.boulder.ibm.com/infocenter/tivihelp/v15r1/topic/com.ibm.omegamon.xe_db2.doc/ko2welcome_pe.htm</a> In earlier releases, this product was called DB2 Performance Expert for z/OS.
WebSphere Application Server	Information center: <a href="http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp">http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp</a>
WebSphere Message Broker with Rules and Formatter Extension	Information center: <a href="http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp">http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp</a> The product is also known as WebSphere MQ Integrator Broker.
WebSphere MQ	Information center: <a href="http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp">http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp</a> The resource includes information about MQSeries <sup>®</sup> .
z/Architecture <sup>®</sup>	Library Center site: <a href="http://www.ibm.com/servers/eserver/zseries/zos/bkserv/">http://www.ibm.com/servers/eserver/zseries/zos/bkserv/</a>

Table 35. Related product information resource locations (continued)

Related product	Information resources
z/OS	<p data-bbox="505 258 1365 281">Library Center site: <a href="http://www.ibm.com/servers/eserver/zseries/zos/bkserv/">http://www.ibm.com/servers/eserver/zseries/zos/bkserv/</a></p> <p data-bbox="505 310 1453 333">This resource includes information about the following z/OS elements and components:</p> <ul data-bbox="505 348 997 1444" style="list-style-type: none"> <li>• Character Data Representation Architecture</li> <li>• Device Support Facilities</li> <li>• DFSORT</li> <li>• Fortran</li> <li>• High Level Assembler</li> <li>• NetView®</li> <li>• SMP/E for z/OS</li> <li>• SNA</li> <li>• TCP/IP</li> <li>• TotalStorage Enterprise Storage Server</li> <li>• VTAM</li> <li>• z/OS C/C++</li> <li>• z/OS Communications Server</li> <li>• z/OS DCE</li> <li>• z/OS DFSMS</li> <li>• z/OS DFSMS Access Method Services</li> <li>• z/OS DFSMSdss</li> <li>• z/OS DFSMSHsm</li> <li>• z/OS DFSMSdfp</li> <li>• z/OS ICSF</li> <li>• z/OS ISPF</li> <li>• z/OS JES3</li> <li>• z/OS Language Environment®</li> <li>• z/OS Managed System Infrastructure</li> <li>• z/OS MVS</li> <li>• z/OS MVS JCL</li> <li>• z/OS Parallel Sysplex</li> <li>• z/OS RMF</li> <li>• z/OS Security Server</li> <li>• z/OS UNIX System Services</li> </ul>
z/OS XL C/C++	<p data-bbox="505 1465 1114 1488"><a href="http://www.ibm.com/software/awdtools/czos/library/">http://www.ibm.com/software/awdtools/czos/library/</a></p>

The following information resources from IBM are not necessarily specific to a single product:

- The DB2 for z/OS Information Roadmap; available at: <http://www.ibm.com/software/data/db2/zos/roadmap.html>
- DB2 Redbooks® and Redbooks about related products; available at: <http://www.ibm.com/redbooks>
- IBM Educational resources:
  - Information about IBM educational offerings is available on the web at: <http://www.ibm.com/software/sw-training/>

- A collection of glossaries of IBM terms in multiple languages is available on the IBM Terminology website at: <http://www.ibm.com/software/globalization/terminology/index.jsp>
- National Language Support information; available at the IBM Publications Center at: <http://www.elink.ibm.com/public/applications/publications/cgi-bin/pbi.cgi>
- *SQL Reference for Cross-Platform Development*; available at the following developerWorks® site: <http://www.ibm.com/developerworks/db2/library/techarticle/0206sqlref/0206sqlref.html>

The following information resources are not published by IBM but can be useful to users of DB2 for z/OS and related products:

- Database design topics:
  - *DB2 for z/OS and OS/390 Development for Performance Volume I*, by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-605-2
  - *DB2 for z/OS and OS/390 Development for Performance Volume II*, by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-606-0
  - *Handbook of Relational Database Design*, by C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8
- Distributed Relational Database Architecture (DRDA) specifications; <http://www.opengroup.org>
- Domain Name System: *DNS and BIND*, Third Edition, Paul Albitz and Cricket Liu, O'Reilly, ISBN 0-59600-158-4
- Microsoft Open Database Connectivity (ODBC) information; <http://msdn.microsoft.com/library/>
- Unicode information; <http://www.unicode.org>

---

## How to obtain DB2 information

You can access the official information about the DB2 product in a number of ways.

- “DB2 on the web”
- “DB2 product information”
- “DB2 education” on page 234
- “How to order the DB2 library” on page 234

### DB2 on the web

Stay current with the latest information about DB2 by visiting the DB2 home page on the web:

<http://www.ibm.com/software/db2zos>

On the DB2 home page, you can find links to a wide variety of information resources about DB2. You can read news items that keep you informed about the latest enhancements to the product. Product announcements, press releases, fact sheets, and technical articles help you plan and implement your database management strategy.

### DB2 product information

The official DB2 for z/OS information is available in various formats and delivery methods. IBM provides mid-version updates to the information in the information center and in softcopy updates that are available on the web and on CD-ROM.

#### Information Management Software for z/OS Solutions Information Center

DB2 product information is viewable in the information center, which is the primary delivery vehicle for information about DB2 for z/OS, IMS, QMF, and related tools. This information center enables you to search across related product information in multiple languages for data management solutions for the z/OS environment and print individual topics or sets of related topics. You can also access, download, and print PDFs of the publications that are associated with the information center topics. Product technical information is provided in a format that offers more options and tools for accessing, integrating, and customizing information resources. The information center is based on Eclipse open source technology.

The Information Management Software for z/OS Solutions Information Center is viewable at the following website:

<http://publib.boulder.ibm.com/infocenter/imzic>

#### CD-ROMs and DVD

Books for DB2 are available on a CD-ROM that is included with your product shipment:

- DB2 10 for z/OS Licensed Library Collection, LK5T-7390, in English

The CD-ROM contains the collection of books for DB2 10 for z/OS in PDF format. Periodically, IBM refreshes the books on subsequent editions of this CD-ROM.

The books for DB2 for z/OS are also available on the following DVD collection kit, which contains online books for many IBM products:

- IBM z/OS Software Products DVD Collection, SK3T-4271, in English

### **PDF format**

Many of the DB2 books are available in PDF (Portable Document Format) for viewing or printing from CD-ROM or the DB2 home page on the web or from the information center. Download the PDF books to your intranet for distribution throughout your enterprise.

### **DB2 education**

IBM Education and Training offers a wide variety of classroom courses to help you quickly and efficiently gain DB2 expertise. IBM schedules classes in cities all over the world. You can find class information, by country, at the IBM Learning Services website:

<http://www.ibm.com/services/learning>

IBM also offers classes at your location, at a time that suits your needs. IBM can customize courses to meet your exact requirements. For more information, including the current local schedule, contact your IBM representative.

### **How to order the DB2 library**

To order books, visit the IBM Publication Center on the web:

<http://www.ibm.com/shop/publications/order>

From the IBM Publication Center, you can go to the Publication Notification System (PNS). PNS users receive electronic notifications of updated publications in their profiles. You have the option of ordering the updates by using the publications direct ordering application or any other IBM publication ordering channel. The PNS application does not send automatic shipments of publications. You will receive updated publications and a bill for them if you respond to the electronic notification.

You can also order DB2 publications and CD-ROMs from your IBM representative or the IBM branch office that serves your locality. If your location is within the United States or Canada, you can place your order by calling one of the toll-free numbers:

- In the U.S., call 1-800-879-2755.
- In Canada, call 1-800-426-4968.

To order additional copies of licensed publications, specify the SOFTWARE option. To order additional publications or CD-ROMs, specify the PUBLICATIONS option. Be prepared to give your customer number, the product number, and either the feature codes or order numbers that you want.

---

## How to use the DB2 library

Titles of books in the library begin with DB2 10 for z/OS. However, references from one book in the library to another are shortened and do not include the product name, version, and release. Instead, they point directly to the section that holds the information. The primary place to find and use information about DB2 for z/OS is the Information Management Software for z/OS Solutions Information Center (<http://publib.boulder.ibm.com/infocenter/imzic>).

If you are new to DB2 for z/OS, *Introduction to DB2 for z/OS* provides a comprehensive introduction to DB2 10 for z/OS. Topics included in this book explain the basic concepts that are associated with relational database management systems in general, and with DB2 for z/OS in particular.

The most rewarding task associated with a database management system is asking questions of it and getting answers, the task called *end use*. Other tasks are also necessary—defining the parameters of the system, putting the data in place, and so on. The tasks that are associated with DB2 are grouped into the following major categories.

### Installation

If you are involved with installing DB2, you will need to use a variety of resources, such as:

- *DB2 Program Directory*
- *DB2 Installation and Migration Guide*
- *DB2 Administration Guide*
- *DB2 Application Programming Guide and Reference for Java*
- *DB2 Codes*
- *DB2 Internationalization Guide*
- *DB2 Messages*
- *DB2 Managing Performance*
- *DB2 RACF Access Control Module Guide*
- *DB2 Utility Guide and Reference*

If you will be using data sharing capabilities you also need *DB2 Data Sharing: Planning and Administration*, which describes installation considerations for data sharing.

If you will be installing and configuring DB2 ODBC, you will need *DB2 ODBC Guide and Reference*.

If you are installing IBM Spatial Support for DB2 for z/OS, you will need *IBM Spatial Support for DB2 for z/OS User's Guide and Reference*.

If you are installing IBM OmniFind® Text Search Server for DB2 for z/OS, you will need *IBM OmniFind Text Search Server for DB2 for z/OS Installation, Administration, and Reference*.

## End use

End users issue SQL statements to retrieve data. They can also insert, update, or delete data, with SQL statements. They might need an introduction to SQL, detailed instructions for using SPUFI, and an alphabetized reference to the types of SQL statements. This information is found in *DB2 Application Programming and SQL Guide*, and *DB2 SQL Reference*.

End users can also issue SQL statements through the DB2 Query Management Facility (QMF) or some other program, and the library for that licensed program might provide all the instruction or reference material they need.

## Application programming

Some users access DB2 without knowing it, using programs that contain SQL statements. DB2 application programmers write those programs. Because they write SQL statements, they need the same resources that end users do.

Application programmers also need instructions for many other topics:

- How to transfer data between DB2 and a host program—written in Java, C, or COBOL, for example
- How to prepare to compile a program that embeds SQL statements
- How to process data from two systems simultaneously, for example, DB2 and IMS or DB2 and CICS
- How to write distributed applications across operating systems
- How to write applications that use Open Database Connectivity (ODBC) to access DB2 servers
- How to write applications that use JDBC and SQLJ with the Java programming language to access DB2 servers
- How to write applications to store XML data on DB2 servers and retrieve XML data from DB2 servers.

The material needed for writing a host program containing SQL is in *DB2 Application Programming and SQL Guide*.

The material needed for writing applications that use JDBC and SQLJ to access DB2 servers is in *DB2 Application Programming Guide and Reference for Java*. The material needed for writing applications that use DB2 CLI or ODBC to access DB2 servers is in *DB2 ODBC Guide and Reference*. The material needed for working with XML data in DB2 is in *DB2 pureXML Guide*. For handling errors, see *DB2 Messages and DB2 Codes*.

Information about writing applications across operating systems can be found in *IBM DB2 SQL Reference for Cross-Platform Development*.

## System and database administration

*Administration* covers almost everything else. *DB2 Administration Guide* divides some of those tasks among the following sections:

- Designing a database: Discusses the decisions that must be made when designing a database and tells how to implement the design by creating and altering DB2 objects, loading data, and adjusting to changes.

- Security and auditing: Describes ways of controlling access to the DB2 system and to data within DB2, to audit aspects of DB2 usage, and to answer other security and auditing concerns.
- Operation and recovery: Describes the steps in normal day-to-day operation and discusses the steps one should take to prepare for recovery in the event of some failure.

*DB2 Managing Performance* explains how to monitor the performance of the DB2 system and its parts. It also lists things that can be done to make some parts run faster.

If you will be using the RACF access control module for DB2 authorization checking, you will need *DB2 RACF Access Control Module Guide*.

If you are involved with DB2 only to design the database, or plan operational procedures, you need *DB2 Administration Guide*. If you also want to carry out your own plans by creating DB2 objects, granting privileges, running utility jobs, and so on, you also need:

- *DB2 SQL Reference*, which describes the SQL statements you use to create, alter, and drop objects and grant and revoke privileges
- *DB2 Utility Guide and Reference*, which explains how to run utilities
- *DB2 Command Reference*, which explains how to run commands

If you will be using data sharing, you need *DB2 Data Sharing: Planning and Administration*, which describes how to plan for and implement data sharing.

Additional information about system and database administration can be found in *DB2 Messages* and *DB2 Codes*, which list messages and codes issued by DB2, with explanations and suggested responses.

## Diagnosis

Diagnostics detect and describe errors in the DB2 program. They might also recommend or apply a remedy. The documentation for this task is in *DB2 Diagnosis Guide and Reference*, *DB2 Messages*, and *DB2 Codes*.



---

## Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
J46A/G4  
555 Bailey Avenue  
San Jose, CA 95141-1003  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

---

## Programming Interface Information

This information is intended to help you to plan for and administer DB2 10 for z/OS. This information also documents General-use Programming Interface and Associated Guidance Information and Product-sensitive Programming Interface and Associated Guidance Information provided by DB2 10 for z/OS.

### General-use Programming Interface and Associated Guidance Information

General-use Programming Interfaces allow the customer to write programs that obtain the services of DB2 10 for z/OS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs by the following markings:

 General-use Programming Interface and Associated Guidance Information...



## 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:

 Product-sensitive Programming Interface and Associated Guidance

Information... 

---

## Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com/)<sup>®</sup> are trademarks or registered marks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at <http://www.ibm.com/legal/copytrade.shtml>.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.



---

## Glossary

**abend** See abnormal end of task.

**abend reason code**

A 4-byte hexadecimal code that uniquely identifies a problem with DB2.

**abnormal end of task (abend)**

Termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve during execution.

**access method services**

The facility that is used to define, alter, delete, print, and reproduce VSAM key-sequenced data sets.

**access path**

The path that is used to locate data that is specified in SQL statements. An access path can be indexed or sequential.

**access path stability**

A characteristic of an access path that defines reliability for dynamic or static queries. Access paths are not regenerated unless there is a schema change or manual intervention.

**active log**

The portion of the DB2 log to which log records are written as they are generated. The active log always contains the most recent log records. See also archive log.

**address space**

A range of virtual storage pages that is identified by a number (ASID) and a collection of segment and page tables that map the virtual pages to real pages of the computer's memory.

**address space connection**

The result of connecting an allied address space to DB2. See also allied address space and task control block.

**address space identifier (ASID)**

A unique system-assigned identifier for an address space.

**AFTER trigger**

A trigger that is specified to be activated after a defined trigger event (an insert, update, or delete operation on the table that is specified in a trigger definition).

Contrast with BEFORE trigger and INSTEAD OF trigger.

**agent** In DB2, the structure that associates all processes that are involved in a DB2 unit of work. See also allied agent and system agent.

**aggregate function**

An operation that derives its result by using values from one or more rows. Contrast with scalar function.

**alias**

An alternative name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 subsystem. An alias can be qualified with a schema qualifier and can thereby be referenced by other users. Contrast with synonym.

**allied address space**

An area of storage that is external to DB2 and that is connected to DB2. An allied address space can request DB2 services. See also address space.

**allied agent**

An agent that represents work requests that originate in allied address spaces. See also system agent.

**allied thread**

A thread that originates at the local DB2 subsystem and that can access data at a remote DB2 subsystem.

**allocated cursor**

A cursor that is defined for a stored procedure result set by using the SQL ALLOCATE CURSOR statement.

**ambiguous cursor**

A database cursor for which DB2 cannot determine whether it is used for update or read-only purposes.

**APAR** See authorized program analysis report.

**APF** See authorized program facility.

**API** See application programming interface.

**APPL** A VTAM network definition statement that is used to define DB2 to VTAM as an application program that uses SNA LU 6.2 protocols.

**application**

A program or set of programs that performs a task; for example, a payroll application.

**application period**

A pair of columns with application-maintained values that indicate the period of time when a row is valid.

**application-period temporal table**

A table that includes an application period.

**application plan**

The control structure that is produced during the bind process. DB2 uses the application plan to process SQL statements that it encounters during statement execution.

**application process**

The unit to which resources and locks are allocated. An application process involves the execution of one or more programs.

**application programming interface (API)**

A functional interface that is supplied by the operating system or by a separately ordered licensed program that allows an application program that is written in a high-level language to use specific data or functions of the operating system or licensed program.

**application requester**

The component on a remote system that generates DRDA requests for data on behalf of an application.

**application server**

The target of a request from a remote application. In the DB2 environment, the application server function is provided by the distributed data facility and is used to access DB2 data from remote applications.

**archive log**

The portion of the DB2 log that contains log records that have been copied from the active log. See also active log.

**ASCII** An encoding scheme that is used to represent strings in many environments, typically on personal computers and workstations. Contrast with EBCDIC and Unicode.

**ASID** See address space identifier.

**attachment facility**

An interface between DB2 and TSO, IMS, CICS, or batch address spaces. An attachment facility allows application programs to access DB2.

**attribute**

A characteristic of an entity. For example, in database design, the phone number of an employee is an attribute of that employee.

**authorization ID**

A string that can be verified for connection to DB2 and to which a set of privileges is allowed. An authorization ID can represent an individual or an organizational group.

**authorized program analysis report (APAR)**

A report of a problem that is caused by a suspected defect in a current release of an IBM supplied program.

**authorized program facility (APF)**

A facility that allows an installation to identify system or user programs that can use sensitive system functions.

**automatic bind**

(More correctly *automatic rebind*.) A process by which SQL statements are bound automatically (without a user issuing a BIND command) when an application process begins execution and the bound application plan or package it requires is not valid.

**automatic query rewrite**

A process that examines an SQL statement that refers to one or more base tables or materialized query tables, and, if appropriate, rewrites the query so that it performs better.

**auxiliary index**

An index on an auxiliary table in which each index entry refers to a LOB or XML document.

**auxiliary table**

A table that contains columns outside the actual table in which they are defined. Auxiliary tables can contain either LOB or XML data.

**backout**

The process of undoing uncommitted changes that an application process made. A backout is often performed in the event

of a failure on the part of an application process, or as a result of a deadlock situation.

### **backward log recovery**

The final phase of restart processing during which DB2 scans the log in a backward direction to apply UNDO log records for all aborted changes.

### **base table**

A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with clone table, materialized query table, result table, temporary table, and transition table.

### **base table space**

A table space that contains base tables.

### **basic row format**

A row format in which values for columns are stored in the row in the order in which the columns are defined by the CREATE TABLE statement. Contrast with reordered row format.

### **basic sequential access method (BSAM)**

An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential-access or a direct-access device.

### **BEFORE trigger**

A trigger that is specified to be activated before a defined trigger event (an insert, an update, or a delete operation on the table that is specified in a trigger definition). Contrast with AFTER trigger and INSTEAD OF trigger.

### **begin column**

In a system period or an application period, the column that indicates the beginning of the period.

### **binary large object (BLOB)**

A binary string data type that contains a sequence of bytes that can range in size from 0 bytes to 2 GB, less 1 byte. This string does not have an associated code page and character set. BLOBs can contain, for example, image, audio, or video data. In general, BLOB values are used whenever a binary string might exceed the limits of the VARBINARY type.

### **binary string**

A sequence of bytes that is not associated

with a CCSID. Binary string data type can be further classified as BINARY, VARBINARY, or BLOB.

### **binary XML format**

A representation of XML data that uses binary values, an approach that facilitates more efficient storage and exchange.

### **bind**

A process by which a usable control structure with SQL statements is generated; the structure is often called an access plan, an application plan, or a package. During this bind process, access paths to the data are selected, and some authorization checking is performed. See also automatic bind.

### **bit data**

- Data with character type CHAR or VARCHAR that is defined with the FOR BIT DATA clause. Note that using BINARY or VARBINARY rather than FOR BIT DATA is highly recommended.
- Data with character type CHAR or VARCHAR that is defined with the FOR BIT DATA clause.
- A form of character data. Binary data is generally more highly recommended than character-for-bit data.

### **bitemporal table**

A table that is both a system-period temporal table and an application-period temporal table.

### **BLOB** See binary large object.

### **block fetch**

A capability in which DB2 can retrieve, or fetch, a large set of rows together. Using block fetch can significantly reduce the number of messages that are being sent across the network. Block fetch applies only to non-rowset cursors that do not update data.

### **bootstrap data set (BSDS)**

A VSAM data set that contains name and status information for DB2 and RBA range specifications for all active and archive log data sets. The BSDS also contains passwords for the DB2 directory and catalog, and lists of conditional restart and checkpoint records.

### **BSAM**

See basic sequential access method.

**BSDS** See bootstrap data set.

**buffer pool**

| An area of memory into which data pages  
| are read, modified, and held during  
| processing.

**built-in data type**

| A data type that IBM supplies. Among  
| the built-in data types for DB2 for z/OS  
| are string, numeric, XML, ROWID, and  
| datetime. Contrast with distinct type.

**built-in function**

| A function that is generated by DB2 and  
| that is in the SYSIBM schema. Contrast  
| with user-defined function. See also  
| function, cast function, external function,  
| sourced function, and SQL function.

**business dimension**

A category of data, such as products or  
time periods, that an organization might  
want to analyze.

**cache structure**

A coupling facility structure that stores  
data that can be available to all members  
of a Sysplex. A DB2 data sharing group  
uses cache structures as group buffer  
pools.

**CAF** See call attachment facility.

**call attachment facility (CAF)**

A DB2 attachment facility for application  
programs that run in TSO or z/OS batch.  
The CAF is an alternative to the DSN  
command processor and provides greater  
control over the execution environment.  
Contrast with Recoverable Resource  
Manager Services attachment facility.

**call-level interface (CLI)**

A callable application programming  
interface (API) for database access, which  
is an alternative to using embedded SQL.

**cascade delete**

A process by which DB2 enforces  
referential constraints by deleting all  
descendent rows of a deleted parent row.

**CASE expression**

An expression that is selected based on  
the evaluation of one or more conditions.

**cast function**

A function that is used to convert  
instances of a (source) data type into  
instances of a different (target) data type.

**castout**

The DB2 process of writing changed  
pages from a group buffer pool to disk.

**castout owner**

The DB2 member that is responsible for  
casting out a particular page set or  
partition.

**catalog**

In DB2, a collection of tables that contains  
descriptions of objects such as tables,  
views, and indexes.

**catalog table**

Any table in the DB2 catalog.

**CCSID**

See coded character set identifier.

**CDB**

See communications database.

**CDRA**

See Character Data Representation  
Architecture.

**CEC**

See central processor complex.

**central electronic complex (CEC)**

See central processor complex.

**central processor complex (CPC)**

A physical collection of hardware that  
consists of main storage, one or more  
central processors, timers, and channels.

**central processor (CP)**

The part of the computer that contains the  
sequencing and processing facilities for  
instruction execution, initial program  
load, and other machine operations.

**CFRM** See coupling facility resource  
management.

**CFRM policy**

The allocation rules for a coupling facility  
structure that are declared by a z/OS  
administrator.

**character conversion**

The process of changing characters from  
one encoding scheme to another.

**Character Data Representation Architecture (CDRA)**

An architecture that is used to achieve  
consistent representation, processing, and  
interchange of string data.

**character large object (CLOB)**

| A character string data type that contains  
| a sequence of bytes that represent

| characters (single-byte, multibyte, or both)  
| that can range in size from 0 bytes to 2  
| GB, less 1 byte. In general, CLOB values  
| are used whenever a character string  
| might exceed the limits of the VARCHAR  
| type.

**character set**

A defined set of characters.

**character string**

| A sequence of bytes that represent bit  
| data, single-byte characters, or a mixture  
| of single-byte and multibyte characters.  
| Character data can be further classified as  
| CHARACTER, VARCHAR, or CLOB.

**check constraint**

A user-defined constraint that specifies the values that specific columns of a base table can contain.

**check integrity**

The condition that exists when each row in a table conforms to the check constraints that are defined on that table.

**check pending**

A state of a table space or partition that prevents its use by some utilities and by some SQL statements because of rows that violate referential constraints, check constraints, or both.

**checkpoint**

A point at which DB2 records status information on the DB2 log; the recovery process uses this information if DB2 abnormally terminates.

**child lock**

For explicit hierarchical locking, a lock that is held on either a table, page, row, or a large object (LOB). Each child lock has a parent lock. See also parent lock.

**CI** See control interval.

**CICS** Represents (in this information): CICS Transaction Server for z/OS: Customer Information Control System Transaction Server for z/OS.

**CICS attachment facility**

| A facility that provides a multithread  
| connection to DB2 to allow applications  
| that run in the CICS environment to  
| execute DB2 statements.

**claim** A notification to DB2 that an object is being accessed. Claims prevent drains

from occurring until the claim is released, which usually occurs at a commit point. Contrast with drain.

**claim class**

A specific type of object access that can be one of the following isolation levels:

- Cursor stability (CS)
- Repeatable read (RR)
- Write

**class of service**

A VTAM term for a list of routes through a network, arranged in an order of preference for their use.

**clause** In SQL, a distinct part of a statement, such as a SELECT clause or a WHERE clause.

**CLI** See call-level interface.

**client** See requester.

**CLOB** See character large object.

| **clone object**

| An object that is associated with a clone  
| table, including the clone table itself and  
| check constraints, indexes, and BEFORE  
| triggers on the clone table.

| **clone table**

| A table that is structurally identical to a  
| base table. The base and clone table each  
| have separate underlying VSAM data  
| sets, which are identified by their data set  
| instance numbers. Contrast with base  
| table.

**closed application**

An application that requires exclusive use of certain statements on certain DB2 objects, so that the objects are managed solely through the external interface of that application.

**clustering index**

An index that determines how rows are physically ordered (*clustered*) in a table space. If a clustering index on a partitioned table is not a partitioning index, the rows are ordered in cluster sequence within each data partition instead of spanning partitions.

| **CM** See conversion mode.

| **CM\*** See conversion mode\*.

- C++ member**  
A data object or function in a structure, union, or class.
- C++ member function**  
An operator or function that is declared as a member of a class. A member function has access to the private and protected data members and to the member functions of objects in its class. Member functions are also called methods.
- C++ object**  
A region of storage. An object is created when a variable is defined or a new function is invoked.  
An instance of a class.
- coded character set**  
A set of unambiguous rules that establish a character set and the one-to-one relationships between the characters of the set and their coded representations.
- coded character set identifier (CCSID)**  
A 16-bit number that uniquely identifies a coded representation of graphic characters. It designates an encoding scheme identifier and one or more pairs that consist of a character set identifier and an associated code page identifier.
- code page**  
A set of assignments of characters to code points. Within a code page, each code point has only one specific meaning. In EBCDIC, for example, the character *A* is assigned code point X'C1', and character *B* is assigned code point X'C2'.
- code point**  
In CDRA, a unique bit pattern that represents a character in a code page.
- code unit**  
The fundamental binary width in a computer architecture that is used for representing character data, such as 7 bits, 8 bits, 16 bits, or 32 bits. Depending on the character encoding form that is used, each code point in a coded character set can be represented by one or more code units.
- coexistence**  
During migration, the period of time in which two releases exist in the same data sharing group.
- cold start**  
A process by which DB2 restarts without processing any log records. Contrast with warm start.
- collection**  
A group of packages that have the same qualifier.
- column**  
The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer).
- column function**  
See aggregate function.
- "come from" checking**  
An LU 6.2 security option that defines a list of authorization IDs that are allowed to connect to DB2 from a partner LU.
- command**  
A DB2 operator command or a DSN subcommand. A command is distinct from an SQL statement.
- command prefix**  
A 1- to 8-character command identifier. The command prefix distinguishes the command as belonging to an application or subsystem rather than to z/OS.
- command recognition character (CRC)**  
A character that permits a z/OS console operator or an IMS subsystem user to route DB2 commands to specific DB2 subsystems.
- command scope**  
The scope of command operation in a data sharing group.
- commit**  
The operation that ends a unit of work by releasing locks so that the database changes that are made by that unit of work can be perceived by other processes. Contrast with rollback.
- commit point**  
A point in time when data is considered consistent.
- common service area (CSA)**  
In z/OS, a part of the common area that contains data areas that are addressable by all address spaces. Most DB2 use is in the extended CSA, which is above the 16-MB line.

**communications database (CDB)**

A set of tables in the DB2 catalog that are used to establish conversations with remote database management systems.

**comparison operator**

A token (such as =, >, or <) that is used to specify a relationship between two values.

**compatibility mode**

See conversion mode.

**compatibility mode\* (CM\*)**

See conversion mode\*.

**composite key**

An ordered set of key columns or expressions of the same table.

**compression dictionary**

The dictionary that controls the process of compression and decompression. This dictionary is created from the data in the table space or table space partition.

**concurrency**

The shared use of resources by more than one application process at the same time.

**conditional restart**

A DB2 restart that is directed by a user-defined conditional restart control record (CRCR).

**connection**

In SNA, the existence of a communication path between two partner LUs that allows information to be exchanged (for example, two DB2 subsystems that are connected and communicating by way of a conversation).

**connection context**

In SQLJ, a Java object that represents a connection to a data source.

**connection declaration clause**

In SQLJ, a statement that declares a connection to a data source.

**connection handle**

The data object containing information that is associated with a connection that DB2 ODBC manages. This includes general status information, transaction status, and diagnostic information.

**connection ID**

An identifier that is supplied by the attachment facility and that is associated with a specific address space connection.

**consistency token**

A timestamp that is used to generate the version identifier for an application. See also version.

**constant**

A language element that specifies an unchanging value. Constants are classified as string constants or numeric constants. Contrast with variable.

**constraint**

A rule that limits the values that can be inserted, deleted, or updated in a table. See referential constraint, check constraint, and unique constraint.

**context**

An application's logical connection to the data source and associated DB2 ODBC connection information that allows the application to direct its operations to a data source. A DB2 ODBC context represents a DB2 thread.

**contracting conversion**

A process that occurs when the length of a converted string is smaller than that of the source string. For example, this process occurs when an EBCDIC mixed-data string that contains DBCS characters is converted to ASCII mixed data; the converted string is shorter because the shift codes are removed.

**control interval (CI)**

- A unit of information that VSAM transfers between virtual and auxiliary storage.
- In a key-sequenced data set or file, the set of records that an entry in the sequence-set index record points to.

**conversation**

Communication, which is based on LU 6.2 or Advanced Program-to-Program Communication (APPC), between an application and a remote transaction program over an SNA logical unit-to-logical unit (LU-LU) session that allows communication while processing a transaction.

**conversion mode\* (CM\*)**

A stage of the version-to-version migration process that applies to a DB2 subsystem or data sharing group that was in enabling-new-function mode (ENFM), enabling-new-function mode\* (ENFM\*), or

new-function mode (NFM) at one time. Fallback to a prior version is not supported. When in conversion mode\*, a DB2 data sharing group cannot coexist with members that are still at the prior version level. Contrast with conversion mode, enabling-new-function mode, enabling-new-function mode\*, and new-function mode.

Previously known as compatibility mode\* (CM\*).

#### **conversion mode (CM)**

The first stage of the version-to-version migration process. In a DB2 data sharing group, members in conversion mode can coexist with members that are still at the prior version level. Fallback to the prior version is also supported. When in conversion mode, the DB2 subsystem cannot use most new functions of the new version. Contrast with conversion mode\*, enabling-new-function mode, enabling-new-function mode\*, and new-function mode.

Previously known as compatibility mode (CM).

#### **coordinator**

The system component that coordinates the commit or rollback of a unit of work that includes work that is done on one or more other systems.

#### **coprocessor**

See SQL statement coprocessor.

#### **copy pool**

A collection of names of storage groups that are processed collectively for fast replication operations.

#### **copy target**

A named set of SMS storage groups that are to be used as containers for copy pool volume copies. A copy target is an SMS construct that lets you define which storage groups are to be used as containers for volumes that are copied by using FlashCopy functions.

#### **copy version**

A point-in-time FlashCopy copy that is managed by HSM. Each copy pool has a version parameter that specifies the number of copy versions to be maintained on disk.

#### **correlated columns**

A relationship between the value of one column and the value of another column.

#### **correlated subquery**

A subquery (part of a WHERE or HAVING clause) that is applied to a row or group of rows of a table or view that is named in an outer subselect statement.

#### **correlation ID**

An identifier that is associated with a specific thread. In TSO, it is either an authorization ID or the job name.

#### **correlation name**

An identifier that is specified and used within a single SQL statement as the exposed name for objects such as a table, view, table function reference, nested table expression, or result of a data change statement. Correlation names are useful in an SQL statement to allow two distinct references to the same base table and to allow an alternative name to be used to represent an object.

#### **cost category**

A category into which DB2 places cost estimates for SQL statements at the time the statement is bound. The cost category is externalized in the COST\_CATEGORY column of the DSN\_STATEMENT\_TABLE when a statement is explained.

#### **coupling facility**

A special PR/SM logical partition (LPAR) that runs the coupling facility control program and provides high-speed caching, list processing, and locking functions in a Parallel Sysplex.

#### **coupling facility resource management (CFRM)**

A component of z/OS that provides the services to manage coupling facility resources in a Parallel Sysplex. This management includes the enforcement of CFRM policies to ensure that the coupling facility and structure requirements are satisfied.

**CP** See central processor.

**CPC** See central processor complex.

**CRC** See command recognition character.

#### **created temporary table**

A persistent table that holds temporary data and is defined with the SQL statement CREATE GLOBAL

TEMPORARY TABLE. Information about created temporary tables is stored in the DB2 catalog and can be shared across application processes. Contrast with declared temporary table. See also temporary table.

**cross-system coupling facility (XCF)**

A component of z/OS that provides functions to support cooperation between authorized programs that run within a Sysplex.

**cross-system extended services (XES)**

A set of z/OS services that allow multiple instances of an application or subsystem, running on different systems in a Sysplex environment, to implement high-performance, high-availability data sharing by using a coupling facility.

**CS** See cursor stability.

**CSA** See common service area.

**CT** See cursor table.

**current data**

Data within a host structure that is current with (identical to) the data within the base table.

**current status rebuild**

The second phase of restart processing during which the status of the subsystem is reconstructed from information on the log.

**cursor** A control structure that an application program uses to point to a single row or multiple rows within some ordered set of rows of a result table. A cursor can be used to retrieve, update, or delete rows from a result table.

**cursor sensitivity**

The degree to which database updates are visible to the subsequent FETCH statements in a cursor.

**cursor stability (CS)**

The isolation level that provides maximum concurrency without the ability to read uncommitted data. With cursor stability, a unit of work holds locks only on its uncommitted changes and on the current row of each of its cursors. See also read stability, repeatable read, and uncommitted read.

**cursor table (CT)**

The internal representation of a cursor.

**cycle** A set of tables that can be ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table. A self-referencing table is a cycle with a single member. See also referential cycle.

**database**

A collection of tables, or a collection of table spaces and index spaces.

**database access thread (DBAT)**

A thread that accesses data at the local subsystem on behalf of a remote subsystem.

**database administrator (DBA)**

An individual who is responsible for designing, developing, operating, safeguarding, maintaining, and using a database.

**database alias**

The name of the target server if it is different from the location name. The database alias is used to provide the name of the database server as it is known to the network.

**database descriptor (DBD)**

An internal representation of a DB2 database definition, which reflects the data definition that is in the DB2 catalog. The objects that are defined in a database descriptor are table spaces, tables, indexes, index spaces, relationships, check constraints, and triggers. A DBD also contains information about accessing tables in the database.

**database exception status**

In a data sharing environment, an indication that something is wrong with a database.

**database identifier (DBID)**

An internal identifier of the database.

**database management system (DBMS)**

A software system that controls the creation, organization, and modification of a database and the access to the data that is stored within it.

**database request module (DBRM)**

A data set member that is created by the DB2 precompiler and that contains

information about SQL statements. DBRMs are used in the bind process.

**database server**

The target of a request from a local application or a remote intermediate database server.

**data currency**

The state in which the data that is retrieved into a host variable in a program is a copy of the data in the base table.

| **data-dependent pagination**

| The process used when applications need  
| to access part of a DB2 result set that is  
| based on a logical key value.

**data dictionary**

A repository of information about an organization's application programs, databases, logical data models, users, and authorizations.

**data partition**

A VSAM data set that is contained within a partitioned table space.

**data-partitioned secondary index (DPSI)**

A secondary index that is partitioned according to the underlying data. Contrast with nonpartitioned secondary index.

| **data set instance number**

| A number that indicates the data set that  
| contains the data for an object.

**data sharing**

A function of DB2 for z/OS that enables applications on different DB2 subsystems to read from and write to the same data concurrently.

**data sharing group**

A collection of one or more DB2 subsystems that directly access and change the same data while maintaining data integrity.

**data sharing member**

A DB2 subsystem that is assigned by XCF services to a data sharing group.

**data source**

A local or remote relational or non-relational data manager that is capable of supporting data access via an ODBC driver that supports the ODBC

APIs. In the case of DB2 for z/OS, the data sources are always relational database managers.

**data type**

An attribute of columns, constants, variables, parameters, special registers, and the results of functions and expressions.

**data warehouse**

A system that provides critical business information to an organization. The data warehouse system cleanses the data for accuracy and currency, and then presents the data to decision makers so that they can interpret and use it effectively and efficiently.

**DBA** See database administrator.

**DBAT** See database access thread.

**DB2 catalog**

A collection of tables that are maintained by DB2 and contain descriptions of DB2 objects, such as tables, views, and indexes.

**DBCLOB**

See double-byte character large object.

**DB2 command**

An instruction to the DB2 subsystem that a user enters to start or stop DB2, to display information on current users, to start or stop databases, to display information on the status of databases, and so on.

**DBCS** See double-byte character set.

**DBD** See database descriptor.

**DB2I** See DB2 Interactive.

**DBID** See database identifier.

**DB2 Interactive (DB2I)**

An interactive service within DB2 that facilitates the execution of SQL statements, DB2 (operator) commands, and programmer commands, and the invocation of utilities.

**DBMS**

See database management system.

**DBRM**

See database request module.

**DB2 thread**

| The database manager structure that

describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to the database manager resources and services. Most DB2 for z/OS functions execute under a thread structure.

#### **DCLGEN**

See declarations generator.

**DDF** See distributed data facility.

#### **deadlock**

Unresolved contention for the use of a resource, such as a table or an index.

#### **declarations generator (DCLGEN)**

A subcomponent of DB2 that generates SQL table declarations and COBOL, C, or PL/I data structure declarations that conform to the table. The declarations are generated from DB2 system catalog information.

#### **declared temporary table**

A non-persistent table that holds temporary data and is defined with the SQL statement DECLARE GLOBAL TEMPORARY TABLE. Information about declared temporary tables is not stored in the DB2 catalog and can be used only by the application process that issued the DECLARE statement. Contrast with created temporary table. See also temporary table.

#### **default value**

A predetermined value, attribute, or option that is assumed when no other value is specified. A default value can be defined for column data in DB2 tables by specifying the DEFAULT keyword in an SQL statement that changes data (such as INSERT, UPDATE, and MERGE).

#### **deferred embedded SQL**

SQL statements that are neither fully static nor fully dynamic. These statements are embedded within an application and are prepared during the execution of the application.

#### **deferred write**

The process of asynchronously writing changed data pages to disk.

#### **degree of parallelism**

The number of concurrently executed operations that are initiated to process a query.

#### **delete hole**

The location on which a cursor is positioned when a row in a result table is refetched and the row no longer exists on the base table. See also update hole.

#### **delete rule**

The rule that tells DB2 what to do to a dependent row when a parent row is deleted. Delete rules include CASCADE, RESTRICT, SET NULL, or NO ACTION.

#### **delete trigger**

A trigger that is defined with the triggering delete SQL operation.

#### **delimited identifier**

A sequence of one or more characters enclosed by escape characters, such as quotation marks ("").

#### **delimiter token**

A string constant, a delimited identifier, an operator symbol, or any of the special characters that are shown in DB2 syntax diagrams.

#### **denormalization**

The intentional duplication of columns in multiple tables to increase data redundancy. Denormalization is sometimes necessary to minimize performance problems. Contrast with normalization.

#### **dependent**

An object (row, table, or table space) that has at least one parent. The object is also said to be a dependent (row, table, or table space) of its parent. See also parent row, parent table, and parent table space.

#### **dependent row**

A row that contains a foreign key that matches the value of a primary key in the parent row.

#### **dependent table**

A table that is a dependent in at least one referential constraint.

#### **descendent**

An object that is a dependent of an object or is the dependent of a descendent of an object.

#### **descendent row**

A row that is dependent on another row, or a row that is a descendent of a dependent row.

**descendent table**

A table that is a dependent of another table, or a table that is a descendent of a dependent table.

**deterministic function**

A user-defined function whose result is dependent on the values of the input arguments. That is, successive invocations with the same input values produce the same answer. Sometimes referred to as a *not-variant* function. Contrast with nondeterministic function (sometimes called a *variant function*).

**dimension**

A data category such as time, products, or markets. The elements of a dimension are referred to as members. See also dimension table.

**dimension table**

The representation of a dimension in a star schema. Each row in a dimension table represents all of the attributes for a particular member of the dimension. See also dimension, star schema, and star join.

**directory**

The DB2 system database that contains internal objects such as database descriptors and skeleton cursor tables.

**disk** A direct-access storage device that records data magnetically.

**distinct type**

A user-defined data type that is represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

**distributed data**

Data that resides on a DBMS other than the local system.

**distributed data facility (DDF)**

A set of DB2 components through which DB2 communicates with another relational database management system.

**Distributed Relational Database Architecture (DRDA)**

A connection protocol for distributed relational database processing that is used by IBM relational database products. DRDA includes protocols for communication between an application and a remote relational database

management system, and for communication between relational database management systems. See also DRDA access.

**DNS** See domain name server.

**DOCID**

See document ID.

**document ID**

A value that uniquely identifies a row that contains an XML column. This value is stored with the row and never changes.

**domain**

The set of valid values for an attribute.

**domain name**

The name by which TCP/IP applications refer to a TCP/IP host within a TCP/IP network.

**domain name server (DNS)**

A special TCP/IP network server that manages a distributed directory that is used to map TCP/IP host names to IP addresses.

**double-byte character large object (DBCLOB)**

A graphic string data type in which a sequence of bytes represent double-byte characters that range in size from 0 bytes to 2 GB, less 1 byte. In general, DBCLOB values are used whenever a double-byte character string might exceed the limits of the VARCHAR type.

**double-byte character set (DBCS)**

A set of characters, which are used by national languages such as Japanese and Chinese, that have more symbols than can be represented by a single byte. Each character is 2 bytes in length. Contrast with single-byte character set and multibyte character set.

**double-precision floating point number**

A 64-bit approximate representation of a real number.

**DPSI** See data-partitioned secondary index.

**drain** The act of acquiring a locked resource by quiescing access to that object. Contrast with claim.

**drain lock**

A lock on a claim class that prevents a claim from occurring.

**DRDA**  
See Distributed Relational Database Architecture.

**DRDA access**  
An open method of accessing distributed data that you can use to connect to another database server to execute packages that were previously bound at the server location.

**DSN**

- The default DB2 subsystem name.
- The name of the TSO command processor of DB2.
- The first three characters of DB2 module and macro names.

**dynamic cursor**  
A named control structure that an application program uses to change the size of the result table and the order of its rows after the cursor is opened. Contrast with static cursor.

**dynamic dump**  
A dump that is issued during the execution of a program, usually under the control of that program.

**dynamic SQL**  
SQL statements that are prepared and executed at run time. In dynamic SQL, the SQL statement is contained as a character string in a host variable or as a constant, and it is not precompiled.

**EA-enabled table space**  
A table space or index space that is enabled for extended addressability and that contains individual partitions (or pieces, for LOB table spaces) that are greater than 4 GB.

**EB** See exabyte.

**EBCDIC**  
Extended binary coded decimal interchange code. An encoding scheme that is used to represent character data in the z/OS, VM, VSE, and iSeries environments. Contrast with ASCII and Unicode.

**embedded SQL**  
SQL statements that are coded within an application program. See static SQL.

**enabling-new-function mode\* (ENFM\*)**  
A transitional stage of the

version-to-version migration process that applies to a DB2 subsystem or data sharing group that was in new-function mode (NFM) at one time. When in enabling-new-function mode\*, a DB2 subsystem or data sharing group is preparing to use the new functions of the new version but cannot yet use them. A data sharing group that is in enabling-new-function mode\* cannot coexist with members that are still at the prior version level. Fallback to a prior version is not supported. Contrast with conversion mode, conversion mode\*, enabling-new-function mode, and new-function mode.

**enabling-new-function mode (ENFM)**

A transitional stage of the version-to-version migration process during which the DB2 subsystem or data sharing group is preparing to use the new functions of the new version. When in enabling-new-function mode, a DB2 data sharing group cannot coexist with members that are still at the prior version level. Fallback to a prior version is not supported, and most new functions of the new version are not available for use in enabling-new-function mode. Contrast with conversion mode, conversion mode\*, enabling-new-function mode\*, and new-function mode.

**enclave**

In Language Environment, an independent collection of routines, one of which is designated as the main routine. An enclave is similar to a program or run unit. See also WLM enclave.

**encoding scheme**

A set of rules to represent character data (ASCII, EBCDIC, or Unicode).

**end column**

In a system period or an application period, the column that indicates the end of the period.

**ENFM** See enabling-new-function mode.

**ENFM\***

See enabling-new-function mode\*.

**entity** A person, object, or concept about which information is stored. In a relational database, entities are represented as tables. A database includes information

| about the entities in an organization or  
| business, and their relationships to each  
| other.

**enumerated list**

A set of DB2 objects that are defined with a LISTDEF utility control statement in which pattern-matching characters (\*, %;, \_ or ?) are not used.

**environment**

A collection of names of logical and physical resources that are used to support the performance of a function.

**environment handle**

| A handle that identifies the global context  
| for database access. All data that is  
| pertinent to all objects in the environment  
| is associated with this handle.

**equijoin**

A join operation in which the join-condition has the form *expression = expression*. See also join, full outer join, inner join, left outer join, outer join, and right outer join.

**error page range**

A range of pages that are considered to be physically damaged. DB2 does not allow users to access any pages that fall within this range.

**escape character**

The symbol, a double quotation (") for example, that is used to enclose an SQL delimited identifier.

**exabyte**

A unit of measure for processor, real and virtual storage capacities, and channel volume that has a value of 1 152 921 504 606 846 976 bytes or 2<sup>60</sup>.

**exception**

| An SQL operation that involves the  
| EXCEPT set operator, which combines  
| two result tables. The result of an  
| exception operation consists of all of the  
| rows that are in only one of the result  
| tables.

**exception table**

A table that holds rows that violate referential constraints or check constraints that the CHECK DATA utility finds.

**exclusive lock**

A lock that prevents concurrently

executing application processes from reading or changing data. Contrast with share lock.

**executable statement**

An SQL statement that can be embedded in an application program, dynamically prepared and executed, or issued interactively.

**execution context**

In SQLJ, a Java object that can be used to control the execution of SQL statements.

**exit routine**

A user-written (or IBM-provided default) program that receives control from DB2 to perform specific functions. Exit routines run as extensions of DB2.

**expanding conversion**

A process that occurs when the length of a converted string is greater than that of the source string. For example, this process occurs when an ASCII mixed-data string that contains DBCS characters is converted to an EBCDIC mixed-data string; the converted string is longer because shift codes are added.

**explicit hierarchical locking**

Locking that is used to make the parent-child relationship between resources known to IRLM. This kind of locking avoids global locking overhead when no inter-DB2 interest exists on a resource.

**explicit privilege**

| A privilege that has a name and is held as  
| the result of an SQL GRANT statement  
| and revoked as the result of an SQL  
| REVOKE statement. For example, the  
| SELECT privilege.

**exposed name**

A correlation name or a table or view name for which a correlation name is not specified.

**expression**

An operand or a collection of operators and operands that yields a single value.

**Extended Recovery Facility (XRF)**

A facility that minimizes the effect of failures in z/OS, VTAM, the host processor, or high-availability applications during sessions between high-availability applications and designated terminals.

This facility provides an alternative subsystem to take over sessions from the failing subsystem.

**Extensible Markup Language (XML)**

A standard metalanguage for defining markup languages that is a subset of Standardized General Markup Language (SGML).

**external function**

A function that has its functional logic implemented in a programming language application that resides outside the database, in the file system of the database server. The association of the function with the external code application is specified by the EXTERNAL clause in the CREATE FUNCTION statement. External functions can be classified as external scalar functions and external table functions. Contrast with sourced function, built-in function, and SQL function.

**external procedure**

A procedure that has its procedural logic implemented in an external programming language application. The association of the procedure with the external application is specified by a CREATE PROCEDURE statement with a LANGUAGE clause that has a value other than SQL and an EXTERNAL clause that implicitly or explicitly specifies the name of the external application. Contrast with external SQL procedure and native SQL procedure.

**external routine**

A user-defined function or stored procedure that is based on code that is written in an external programming language.

**external SQL procedure**

An SQL procedure that is processed using a generated C program that is a representation of the procedure. When an external SQL procedure is called, the C program representation of the procedure is executed in a stored procedures address space. Contrast with external procedure and native SQL procedure.

**failed member state**

A state of a member of a data sharing group in which the member's task,

address space, or z/OS system terminates before the state changes from active to quiesced.

**fallback**

The process of returning to a previous release of DB2 after attempting or completing migration to a current release. Fallback is supported only from a subsystem that is in conversion mode.

**false global lock contention**

A contention indication from the coupling facility that occurs when multiple lock names are hashed to the same indicator and when no real contention exists.

**fan set**

A direct physical access path to data, which is provided by an index, hash, or link; a fan set is the means by which DB2 supports the ordering of data.

**federated database**

The combination of a DB2 server (in Linux, UNIX, and Windows environments) and multiple data sources to which the server sends queries. In a federated database system, a client application can use a single SQL statement to join data that is distributed across multiple database management systems and can view the data as if it were local.

**fetch orientation**

The specification of the desired placement of the cursor as part of a FETCH statement. The specification can be before or after the rows of the result table (with BEFORE or AFTER). The specification can also have either a single-row fetch orientation (for example, NEXT, LAST, or ABSOLUTE *n*) or a rowset fetch orientation (for example, NEXT ROWSET, LAST ROWSET, or ROWSET STARTING AT ABSOLUTE *n*).

**field procedure**

A user-written exit routine that is designed to receive a single value and transform (encode or decode) it in any way the user can specify.

**file reference variable**

A host variable that is declared with one of the derived data types (BLOB\_FILE, CLOB\_FILE, DBCLOB\_FILE); file

reference variables direct the reading of a LOB from a file or the writing of a LOB into a file.

**filter factor**

A number between zero and one that estimates the proportion of rows in a table for which a predicate is true.

**fixed-length string**

A character, graphic, or binary string whose length is specified and cannot be changed. Contrast with varying-length string.

**FlashCopy**

A function on the IBM Enterprise Storage Server that can, in conjunction with the BACKUP SYSTEM utility, create a point-in-time copy of data while an application is running.

**foreign key**

A column or set of columns in a dependent table of a constraint relationship. The key must have the same number of columns, with the same descriptions, as the primary key of the parent table. Each foreign key value must either match a parent key value in the related parent table or be null.

**forest** An ordered set of subtrees of XML nodes.

**forward log recovery**

The third phase of restart processing during which DB2 processes the log in a forward direction to apply all REDO log records.

**free space**

The total amount of unused space in a page; that is, the space that is not used to store records or control information is free space.

**full outer join**

The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of both tables. See also join, equijoin, inner join, left outer join, outer join, and right outer join.

**fullselect**

A subselect, a fullselect in parentheses, or a number of both that are combined by set operators. Fullselect specifies a result table. If a set operator is not used, the

result of the fullselect is the result of the specified subselect or fullselect.

**fully escaped mapping**

A mapping from an SQL identifier to an XML name when the SQL identifier is a column name.

**function**

A mapping, which is embodied as a program (the function body) that can be invoked by means of zero or more input values (arguments) to a single value (the result). See also aggregate function and scalar function.

Functions can be user-defined, built-in, or generated by DB2. (See also built-in function, cast function, external function, sourced function, SQL function, and user-defined function.)

**function definer**

The authorization ID of the owner of the schema of the function that is specified in the CREATE FUNCTION statement.

**function package**

A package that results from binding the DBRM for a function program.

**function package owner**

The authorization ID of the user who binds the function program's DBRM into a function package.

**function signature**

The logical concatenation of a fully qualified function name with the data types of all of its parameters.

**GB** Gigabyte. A value of (1 073 741 824 bytes).

**GBP** See group buffer pool.

**GBP-dependent**

The status of a page set or page set partition that is dependent on the group buffer pool. Either read/write interest is active among DB2 subsystems for this page set, or the page set has changed pages in the group buffer pool that have not yet been cast out to disk.

**generalized trace facility (GTF)**

A z/OS service program that records significant system events such as I/O interrupts, SVC interrupts, program interrupts, or external interrupts.

**generic resource name**

A name that VTAM uses to represent

several application programs that provide the same function in order to handle session distribution and balancing in a Sysplex environment.

**getpage**

An operation in which DB2 accesses a data page.

**global lock**

A lock that provides concurrency control within and among DB2 subsystems. The scope of the lock is across all DB2 subsystems of a data sharing group.

**global lock contention**

Conflicts on locking requests between different DB2 members of a data sharing group when those members are trying to serialize shared resources.

**governor**

See resource limit facility.

**graphic string**

A sequence of DBCS characters. Graphic data can be further classified as GRAPHIC, VARGRAPHIC, or DBCLOB.

**GRECP**

See group buffer pool recovery pending.

**gross lock**

The *shared*, *update*, or *exclusive* mode locks on a table, partition, or table space.

**group buffer pool duplexing**

The ability to write data to two instances of a group buffer pool structure: a primary group buffer pool and a secondary group buffer pool. z/OS publications refer to these instances as the "old" (for primary) and "new" (for secondary) structures.

**group buffer pool (GBP)**

A coupling facility cache structure that is used by a data sharing group to cache data and to ensure that the data is consistent for all members.

**group buffer pool recovery pending (GRECP)**

The state that exists after the buffer pool for a data sharing group is lost. When a page set is in this state, changes that are recorded in the log must be applied to the affected page set before the page set can be used.

**group level**

The release level of a data sharing group,

which is established when the first member migrates to a new release.

**group name**

The z/OS XCF identifier for a data sharing group.

**group restart**

A restart of at least one member of a data sharing group after the loss of either locks or the shared communications area.

**GTF** See generalized trace facility.

**handle**

In DB2 ODBC, a variable that refers to a data structure and associated resources. See also statement handle, connection handle, and environment handle.

**hash access**

Access to a table using the hash value of a key that is defined by the *organization-clause* of a CREATE TABLE statement or ALTER TABLE statement.

**hash overflow index**

A DB2 index used to track data rows that do not fit into the fixed hash space, and therefore, reside in the hash overflow space. DB2 accesses the hash overflow index to fetch rows from the hash overflow area.

**help panel**

A screen of information that presents tutorial text to assist a user at the workstation or terminal.

**heuristic damage**

The inconsistency in data between one or more participants that results when a heuristic decision to resolve an indoubt LUW at one or more participants differs from the decision that is recorded at the coordinator.

**heuristic decision**

A decision that forces indoubt resolution at a participant by means other than automatic resynchronization between coordinator and participant.

**histogram statistics**

A way of summarizing data distribution. This technique divides up the range of possible values in a data set into intervals, such that each interval contains approximately the same percentage of the values. A set of statistics are collected for each interval.

**history table**  
A base table that is associated with a system-period temporal table. A history table is used by DB2 to store the historical versions of the rows from the associated system-period temporal table.

**hole** A row of the result table that cannot be accessed because of a delete or an update that has been performed on the row. See also delete hole and update hole.

**home address space**  
The area of storage that z/OS currently recognizes as *dispatched*.

**host** The set of programs and resources that are available on a given TCP/IP instance.

**host expression**  
A Java variable or expression that is referenced by SQL clauses in an SQLJ application program.

**host identifier**  
A name that is declared in the host program.

**host language**  
A programming language in which you can embed SQL statements.

**host program**  
An application program that is written in a host language and that contains embedded SQL statements.

**host structure**  
In an application program, a structure that is referenced by embedded SQL statements.

**host variable**  
In an application program written in a host language, an application variable that is referenced by embedded SQL statements.

**host variable array**  
An array of elements, each of which corresponds to a value for a column. The dimension of the array determines the maximum number of rows for which the array can be used.

**IBM System z9 Integrated Processor (zIIP)**  
A specialized processor that can be used for some DB2 functions.

**IDCAMS**  
An IBM program that is used to process access method services commands. It can

be invoked as a job or jobstep, from a TSO terminal, or from within a user's application program.

#### **IDCAMS LISTCAT**

A facility for obtaining information that is contained in the access method services catalog.

#### **identity column**

A column that provides a way for DB2 to automatically generate a numeric value for each row. Identity columns are defined with the AS IDENTITY clause. Uniqueness of values can be ensured by defining a unique index that contains only the identity column. A table can have no more than one identity column.

**IFCID** See instrumentation facility component identifier.

**IFI** See instrumentation facility interface.

**IFI call**  
An invocation of the instrumentation facility interface (IFI) by means of one of its defined functions.

**image copy**  
An exact reproduction of all or part of a table space. DB2 provides utility programs to make full image copies (to copy the entire table space) or incremental image copies (to copy only those pages that have been modified since the last image copy).

**IMS attachment facility**  
A DB2 subcomponent that uses z/OS subsystem interface (SSI) protocols and cross-memory linkage to process requests from IMS to DB2 and to coordinate resource commitment.

**in-abort**  
A status of a unit of recovery. If DB2 fails after a unit of recovery begins to be rolled back, but before the process is completed, DB2 continues to back out the changes during restart.

**in-commit**  
A status of a unit of recovery. If DB2 fails after beginning its phase 2 commit processing, it "knows," when restarted, that changes made to data are consistent. Such units of recovery are termed *in-commit*.

**independent**

An object (row, table, or table space) that is neither a parent nor a dependent of another object.

**index** A set of pointers that are logically ordered by the values of a key. Indexes can provide faster access to data and can enforce uniqueness on the rows in a table.

**index-controlled partitioning**

A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are specified on the CREATE INDEX statement. Partition limits are saved in the LIMITKEY column of the SYSIBM.SYSINDEXPART catalog table.

**index key**

The set of columns in a table that is used to determine the order of index entries.

**index partition**

A VSAM data set that is contained within a partitioning index space.

**index space**

A page set that is used to store the entries of one index.

**indicator column**

A 4-byte value that is stored in a base table in place of a LOB column.

**indicator variable**

A variable that is used to represent the null value in an application program. If the value for the selected column is null, a negative value is placed in the indicator variable.

**indoubt**

A status of a unit of recovery. If DB2 fails after it has finished its phase 1 commit processing and before it has started phase 2, only the commit coordinator knows if an individual unit of recovery is to be committed or rolled back. At restart, if DB2 lacks the information it needs to make this decision, the status of the unit of recovery is *indoubt* until DB2 obtains this information from the coordinator. More than one unit of recovery can be *indoubt* at restart.

**indoubt resolution**

The process of resolving the status of an *indoubt* logical unit of work to either the committed or the rollback state.

**inflight**

A status of a unit of recovery. If DB2 fails before its unit of recovery completes phase 1 of the commit process, it merely backs out the updates of its unit of recovery at restart. These units of recovery are termed *inflight*.

**inheritance**

The passing downstream of class resources or attributes from a parent class in the class hierarchy to a child class.

**initialization file**

For DB2 ODBC applications, a file containing values that can be set to adjust the performance of the database manager.

**inline copy**

A copy that is produced by the LOAD or REORG utility. The data set that the inline copy produces is logically equivalent to a full image copy that is produced by running the COPY utility with read-only access (SHRLEVEL REFERENCE).

**inline SQL PL**

A subset of SQL procedural language that can be used in SQL functions, triggers, and dynamic compound statements. See also SQL procedural language.

**inner join**

The result of a join operation that includes only the matched rows of both tables that are being joined. See also join, equijoin, full outer join, left outer join, outer join, and right outer join.

**inoperative package**

In DB2 Version 9.1 for z/OS and earlier, a package that cannot be used because one or more user-defined functions or procedures that the package depends on were dropped. Such a package must be explicitly rebound. Contrast with invalid package.

**insensitive cursor**

A cursor that is not sensitive to inserts, updates, or deletes that are made to the underlying rows of a result table after the result table has been materialized.

**insert trigger**

A trigger that is defined with the triggering SQL operation, an insert.

**install** The process of preparing a DB2 subsystem to operate as a z/OS subsystem.

**INSTEAD OF trigger**

A trigger that is associated with a single view and is activated by an insert, update, or delete operation on the view and that can define how to propagate the insert, update, or delete operation on the view to the underlying tables of the view. Contrast with BEFORE trigger and AFTER trigger.

**instrumentation facility component identifier (IFCID)**

A value that names and identifies a trace record of an event that can be traced. As a parameter on the START TRACE and MODIFY TRACE commands, it specifies that the corresponding event is to be traced.

**instrumentation facility interface (IFI)**

A programming interface that enables programs to obtain online trace data about DB2, to submit DB2 commands, and to pass data to DB2.

**Interactive System Productivity Facility (ISPF)**

An IBM licensed program that provides interactive dialog services in a z/OS environment.

**inter-DB2 R/W interest**

A property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members.

**intermediate database server**

The target of a request from a local application or a remote application requester that is forwarded to another database server.

**internal resource lock manager (IRLM)**

A z/OS subsystem that DB2 uses to control communication and database locking.

**internationalization**

The support for an encoding scheme that is able to represent the code points of characters from many different geographies and languages. To support all geographies, the Unicode standard requires more than 1 byte to represent a single character. See also Unicode.

**intersection**

An SQL operation that involves the INTERSECT set operator, which combines two result tables. The result of an intersection operation consists of all of the rows that are in both result tables.

**invalid package**

In DB2 Version 9.1 for z/OS and earlier, a package that depends on an object (other than a user-defined function) that is dropped. Such a package is implicitly rebound on invocation. Contrast with inoperative package.

**IP address**

A value that uniquely identifies a TCP/IP host.

**IRLM** See internal resource lock manager.

**isolation level**

The degree to which a unit of work is isolated from the updating operations of other units of work. See also cursor stability, read stability, repeatable read, and uncommitted read.

**ISPF** See Interactive System Productivity Facility.

**iterator**

In SQLJ, an object that contains the result set of a query. An iterator is equivalent to a cursor in other host languages.

**iterator declaration clause**

In SQLJ, a statement that generates an iterator declaration class. An iterator is an object of an iterator declaration class.

**JAR** See Java Archive.

**Java Archive (JAR)**

A file format that is used for aggregating many files into a single file.

**JDBC** A Sun Microsystems database application programming interface (API) for Java that allows programs to access database management systems by using callable SQL.

**join** A relational operation that allows retrieval of data from two or more tables based on matching column values. See also equijoin, full outer join, inner join, left outer join, outer join, and right outer join.

**KB** Kilobyte. A value of 1024 bytes.

**Kerberos**

A network authentication protocol that is designed to provide strong authentication for client/server applications by using secret-key cryptography.

**Kerberos ticket**

A transparent application mechanism that transmits the identity of an initiating principal to its target. A simple ticket contains the principal's identity, a session key, a timestamp, and other information, which is sealed using the target's secret key.

**key** A column, an ordered collection of columns, or an expression that is identified in the description of a table, index, or referential constraint. The same column or expression can be part of more than one key.

**key-sequenced data set (KSDS)**

A VSAM file or data set whose records are loaded in key sequence and controlled by an index.

**KSDS** See key-sequenced data set.

**large object (LOB)**

A sequence of bytes representing bit data, single-byte characters, double-byte characters, or a mixture of single- and double-byte characters. A LOB can be up to 2 GB minus 1 byte in length. See also binary large object, character large object, and double-byte character large object.

**last agent optimization**

An optimized commit flow for either presumed-nothing or presumed-abort protocols in which the last agent, or final participant, becomes the commit coordinator. This flow saves at least one message.

**latch** A DB2 mechanism for controlling concurrent events or the use of system resources.

**LCID** See log control interval definition.

**LDS** See linear data set.

**leaf page**

An index page that contains pairs of keys and RIDs and that points to actual data. Contrast with nonleaf page.

**left outer join**

The result of a join operation that

includes the matched rows of both tables that are being joined, and that preserves the unmatched rows of the first table. See also join, equijoin, full outer join, inner join, outer join, and right outer join.

**limit key**

The highest value of the index key for a partition.

**linear data set (LDS)**

A VSAM data set that contains data but no control information. A linear data set can be accessed as a byte-addressable string in virtual storage.

**linkage editor**

A computer program for creating load modules from one or more object modules or load modules by resolving cross references among the modules and, if necessary, adjusting addresses.

**link-edit**

The action of creating a loadable computer program using a linkage editor.

**list** A type of object, which DB2 utilities can process, that identifies multiple table spaces, multiple index spaces, or both. A list is defined with the LISTDEF utility control statement.

**list structure**

A coupling facility structure that lets data be shared and manipulated as elements of a queue.

**L-lock** See logical lock.

**load module**

A program unit that is suitable for loading into main storage for execution. The output of a linkage editor.

**LOB** See large object.

**LOB locator**

A mechanism that allows an application program to manipulate a large object value in the database system. A LOB locator is a fullword integer value that represents a single LOB value. An application program retrieves a LOB locator into a host variable and can then apply SQL operations to the associated LOB value using the locator.

**LOB lock**

A lock on a LOB value.

**LOB table space**

A table space that contains all the data for a particular LOB column in the related base table.

**local** A way of referring to any object that the local DB2 subsystem maintains. A *local table*, for example, is a table that is maintained by the local DB2 subsystem. Contrast with remote.

**locale** The definition of a subset of a user's environment that combines a CCSID and characters that are defined for a specific language and country.

**local lock**

A lock that provides intra-DB2 concurrency control, but not inter-DB2 concurrency control; that is, its scope is a single DB2.

**local subsystem**

The unique relational DBMS to which the user or application program is directly connected (in the case of DB2, by one of the DB2 attachment facilities).

**location**

The unique name of a database server. An application uses the location name to access a DB2 database server. A database alias can be used to override the location name when accessing a remote server.

**location alias**

Another name by which a database server identifies itself in the network. Applications can use this name to access a DB2 database server.

**lock** A means of controlling concurrent events or access to data. DB2 locking is performed by the IRLM.

**lock duration**

The interval over which a DB2 lock is held.

**lock escalation**

The promotion of a lock from a row, page, or LOB lock to a table space lock because the number of page locks that are concurrently held on a given resource exceeds a preset limit.

**locking**

The process by which the integrity of data is ensured. Locking prevents concurrent users from accessing inconsistent data. See also claim, drain, and latch.

**lock mode**

A representation for the type of access that concurrently running programs can have to a resource that a DB2 lock is holding.

**lock object**

The resource that is controlled by a DB2 lock.

**lock promotion**

The process of changing the size or mode of a DB2 lock to a higher, more restrictive level.

**lock size**

The amount of data that is controlled by a DB2 lock on table data; the value can be a row, a page, a LOB, a partition, a table, or a table space.

**lock structure**

A coupling facility data structure that is composed of a series of lock entries to support shared and exclusive locking for logical resources.

**log**

A collection of records that describe the events that occur during DB2 execution and that indicate their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

**log control interval definition**

A suffix of the physical log record that tells how record segments are placed in the physical control interval.

**logical claim**

A claim on a logical partition of a nonpartitioning index.

**logical index partition**

The set of all keys that reference the same data partition.

**logical lock (L-lock)**

The lock type that transactions use to control intra- and inter-DB2 data concurrency between transactions. Contrast with physical lock (P-lock).

**logically complete**

A state in which the concurrent copy process is finished with the initialization of the target objects that are being copied. The target objects are available for update.

**logical page list (LPL)**

A list of pages that are in error and that cannot be referenced by applications until the pages are recovered. The page is in *logical error* because the actual media (coupling facility or disk) might not contain any errors. Usually a connection to the media has been lost.

**logical partition**

A set of key or RID pairs in a nonpartitioning index that are associated with a particular partition.

**logical recovery pending (LRECP)**

The state in which the data and the index keys that reference the data are inconsistent.

**logical unit (LU)**

An access point through which an application program accesses the SNA network in order to communicate with another application program. See also LU name.

**logical unit of work**

The processing that a program performs between synchronization points.

**logical unit of work identifier (LUWID)**

A name that uniquely identifies a thread within a network. This name consists of a fully-qualified LU network name, an LUW instance number, and an LUW sequence number.

**log initialization**

The first phase of restart processing during which DB2 attempts to locate the current end of the log.

**log record header (LRH)**

A prefix, in every log record, that contains control information.

**log record sequence number (LRSN)**

An identifier for a log record that is associated with a data sharing member. DB2 uses the LRSN for recovery in the data sharing environment.

**log truncation**

A process by which an explicit starting RBA is established. This RBA is the point at which the next byte of log data is to be written.

**LPL** See logical page list.

**LRECP**

See logical recovery pending.

**LRH** See log record header.

**LRSN** See log record sequence number.

**LU** See logical unit.

**LU name**

Logical unit name, which is the name by which VTAM refers to a node in a network.

**LUW** See logical unit of work.

**LUWID**

See logical unit of work identifier.

**mapping table**

A table that the REORG utility uses to map the associations of the RIDs of data records in the original copy and in the shadow copy. This table is created by the user.

**mass delete**

The deletion of all rows of a table.

**materialize**

- The process of putting rows from a view or nested table expression into a work file for additional processing by a query.
- The placement of a LOB value into contiguous storage. Because LOB values can be very large, DB2 avoids materializing LOB data until doing so becomes absolutely necessary.

**materialized query table**

A table that is used to contain information that is derived and can be summarized from one or more source tables. Contrast with base table.

**MB** Megabyte (1 048 576 bytes).

**MBCS** See multibyte character set.

**member name**

The z/OS XCF identifier for a particular DB2 subsystem in a data sharing group.

**menu** A displayed list of available functions for selection by the operator. A menu is sometimes called a *menu panel*.

**metalanguage**

A language that is used to create other specialized languages.

**migration**

The process of converting a subsystem with a previous release of DB2 to an updated or current release. In this process, you can acquire the functions of the updated or current release without losing the data that you created on the previous release.

**mixed data string**

A character string that can contain both single-byte and double-byte characters.

**mode name**

A VTAM name for the collection of physical and logical characteristics and attributes of a session.

**modify locks**

An L-lock or P-lock with a MODIFY attribute. A list of these active locks is kept at all times in the coupling facility lock structure. If the requesting DB2 subsystem fails, that DB2 subsystem's modify locks are converted to retained locks.

**multibyte character set (MBCS)**

A character set that represents single characters with more than a single byte. UTF-8 is an example of an MBCS. Characters in UTF-8 can range from 1 to 4 bytes in DB2. Contrast with single-byte character set and double-byte character set. See also Unicode.

**multidimensional analysis**

The process of assessing and evaluating an enterprise on more than one level.

**Multiple Virtual Storage (MVS)**

An element of the z/OS operating system. This element is also called the Base Control Program (BCP).

**multisite update**

Distributed relational database processing in which data is updated in more than one location within a single unit of work.

**multithreading**

Multiple TCBS that are executing one copy of DB2 ODBC code concurrently (sharing a processor) or in parallel (on separate central processors).

**MVS** See Multiple Virtual Storage.

**native SQL procedure**

An SQL procedure that is processed by converting the procedural statements to a

native representation that is stored in the database directory, as is done with other SQL statements. When a native SQL procedure is called, the native representation is loaded from the directory, and DB2 executes the procedure. Contrast with external procedure and external SQL procedure.

**nested table expression**

A fullselect in a FROM clause (surrounded by parentheses).

**network identifier (NID)**

The network ID that is assigned by IMS or CICS, or if the connection type is RRSAF, the RRS unit of recovery ID (URID).

**new-function mode (NFM)**

The normal mode of operation that exists after successful completion of a version-to-version migration. At this stage, all new functions of the new version are available for use. A DB2 data sharing group cannot coexist with members that are still at the prior version level, and fallback to a prior version is not supported. Contrast with conversion mode, conversion mode\*, enabling-new-function mode, and enabling-new-function mode\*.

**NFM** See new-function mode.

**NID** See network identifier.

**node ID index**

See XML node ID index.

**nondeterministic function**

A user-defined function whose result is not solely dependent on the values of the input arguments. That is, successive invocations with the same argument values can produce a different answer. This type of function is sometimes called a *variant* function. Contrast with deterministic function (sometimes called a *not-variant function*).

**nonleaf page**

A page that contains keys and page numbers of other pages in the index (either leaf or nonleaf pages). Nonleaf pages never point to actual data. Contrast with leaf page.

**nonpartitioned index**

An index that is not physically

partitioned. Both partitioning indexes and secondary indexes can be nonpartitioned.

**nonpartitioned secondary index (NPSI)**

An index on a partitioned table space that is not the partitioning index and is not partitioned. Contrast with data-partitioned secondary index.

**nonpartitioning index**

See secondary index.

**nonscrollable cursor**

A cursor that can be moved only in a forward direction. Nonscrollable cursors are sometimes called forward-only cursors or serial cursors.

**normalization**

A key step in the task of building a logical relational database design. Normalization helps you avoid redundancies and inconsistencies in your data. An entity is normalized if it meets a set of constraints for a particular normal form (first normal form, second normal form, and so on). Contrast with denormalization.

**not-variant function**

See deterministic function.

**NPSI** See nonpartitioned secondary index.

**NUL** The null character ('\0'), which is represented by the value X'00'. In C, this character denotes the end of a string.

**null** A special value that indicates the absence of information.

**null terminator**

In C, the value that indicates the end of a string. For EBCDIC, ASCII, and Unicode UTF-8 strings, the null terminator is a single-byte value (X'00'). For Unicode UTF-16 or UCS-2 (wide) strings, the null terminator is a double-byte value (X'0000').

**ODBC**

See Open Database Connectivity.

**ODBC driver**

A dynamically-linked library (DLL) that implements ODBC function calls and interacts with a data source.

**OLAP** See online analytical processing.

**online analytical processing (OLAP)**

The process of collecting data from one or

many sources; transforming and analyzing the consolidated data quickly and interactively; and examining the results across different dimensions of the data by looking for patterns, trends, and exceptions within complex relationships of that data.

**Open Database Connectivity (ODBC)**

A Microsoft database application programming interface (API) for C that allows access to database management systems by using callable SQL. ODBC does not require the use of an SQL preprocessor. In addition, ODBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time. This means that applications no longer need to be directly linked to the modules of all the database management systems that are supported.

**ordinary identifier**

An uppercase letter followed by zero or more characters, each of which is an uppercase letter, a digit, or the underscore character. An ordinary identifier must not be a reserved word.

**ordinary token**

A numeric constant, an ordinary identifier, a host identifier, or a keyword.

**originating task**

In a parallel group, the primary agent that receives data from other execution units (referred to as *parallel tasks*) that are executing portions of the query in parallel.

**outer join**

The result of a join operation that includes the matched rows of both tables that are being joined and preserves some or all of the unmatched rows of the tables that are being joined. See also join, equijoin, full outer join, inner join, left outer join, and right outer join.

**overloaded function**

A function name for which multiple function instances exist.

**package**

An object containing a set of SQL statements that have been statically bound and that is available for

processing. A package is sometimes also called an *application package*.

**package list**

An ordered list of package names that may be used to extend an application plan.

**package name**

The name of an object that is used for an application package or an SQL procedure package. An application package is a bound version of a database request module (DBRM) that is created by a BIND PACKAGE or REBIND PACKAGE command. An SQL procedural language package is created by a CREATE or ALTER PROCEDURE statement for a native SQL procedure. The name of a package consists of a location name, a collection ID, a package ID, and a version ID.

**page**

A unit of storage within a table space (4 KB, 8 KB, 16 KB, or 32 KB) or index space (4 KB, 8 KB, 16 KB, or 32 KB). In a table space, a page contains one or more rows of a table. In a LOB or XML table space, a LOB or XML value can span more than one page, but no more than one LOB or XML value is stored on a page.

**page set**

Another way to refer to a table space or index space. Each page set consists of a collection of VSAM data sets.

**page set recovery pending (PSRCP)**

A restrictive state of an index space. In this case, the entire page set must be recovered. Recovery of a logical part is prohibited.

**panel**

A predefined display image that defines the locations and characteristics of display fields on a display surface (for example, a *menu panel*).

**parallel complex**

A cluster of machines that work together to handle multiple transactions and applications.

**parallel group**

A set of consecutive operations that execute in parallel and that have the same number of parallel tasks.

**parallel I/O processing**

A form of I/O processing in which DB2

initiates multiple concurrent requests for a single user query and performs I/O processing concurrently (in *parallel*) on multiple data partitions.

**parallelism assistant**

In Sysplex query parallelism, a DB2 subsystem that helps to process parts of a parallel query that originates on another DB2 subsystem in the data sharing group.

**parallelism coordinator**

In Sysplex query parallelism, the DB2 subsystem from which the parallel query originates.

**Parallel Sysplex**

A set of z/OS systems that communicate and cooperate with each other through certain multisystem hardware components and software services to process customer workloads.

**parallel task**

The execution unit that is dynamically created to process a query in parallel. A parallel task is implemented by a z/OS service request block.

**parameter marker**

A question mark (?) that appears in a statement string of a dynamic SQL statement. The question mark can appear where a variable could appear if the statement string were a static SQL statement.

**parameter-name**

An SQL identifier that designates a parameter in a routine that is written by a user. Parameter names are required for SQL procedures and SQL functions, and they are used in the body of the routine to refer to the values of the parameters. Parameter names are optional for external routines.

**parent key**

A primary key or unique key in the parent table of a referential constraint. The values of a parent key determine the valid values of the foreign key in the referential constraint.

**parent lock**

For explicit hierarchical locking, a lock that is held on a resource that might have child locks that are lower in the hierarchy.

A parent lock is usually the table space lock or the partition intent lock. See also child lock.

**parent row**

A row whose primary key value is the foreign key value of a dependent row.

**parent table**

A table whose primary key is referenced by the foreign key of a dependent table.

**parent table space**

A table space that contains a parent table. A table space containing a dependent of that table is a dependent table space.

**participant**

An entity other than the commit coordinator that takes part in the commit process. The term participant is synonymous with agent in SNA.

**partition**

A portion of a page set. Each partition corresponds to a single, independently extendable data set. The maximum size of a partition depends on the number of partitions in the partitioned page set. All partitions of a given page set have the same maximum size.

**partition-by-growth table space**

A table space whose size can grow to accommodate data growth. DB2 for z/OS manages partition-by-growth table spaces by automatically adding new data sets when the database needs more space to satisfy an insert operation. Contrast with range-partitioned table space. See also universal table space.

**partitioned data set (PDS)**

A data set in disk storage that is divided into partitions, which are called members. Each partition can contain a program, part of a program, or data. A program library is an example of a partitioned data set.

**partitioned index**

An index that is physically partitioned. Both partitioning indexes and secondary indexes can be partitioned.

**partitioned page set**

A partitioned table space or an index space. Header pages, space map pages,

data pages, and index pages reference data only within the scope of the partition.

**partitioned table space**

A table space that is based on a single table and that is subdivided into partitions, each of which can be processed independently by utilities. Contrast with segmented table space and universal table space.

**partitioning index**

An index in which the leftmost columns are the partitioning columns of the table. The index can be partitioned or nonpartitioned.

**partner logical unit**

An access point in the SNA network that is connected to the local DB2 subsystem by way of a VTAM conversation.

**path** See SQL path.

**PDS** See partitioned data set.

**period** An interval of time that is defined by two datetime columns in a temporal table. A period contains a begin column and an end column. See also begin column and end column.

**physical consistency**

The state of a page that is not in a partially changed state.

**physical lock (P-lock)**

A type of lock that DB2 acquires to provide consistency of data that is cached in different DB2 subsystems. Physical locks are used only in data sharing environments. Contrast with logical lock (L-lock).

**physically complete**

The state in which the concurrent copy process is completed and the output data set has been created.

**piece** A data set of a nonpartitioned page set.

**plan** See application plan.

**plan allocation**

The process of allocating DB2 resources to a plan in preparation for execution.

**plan member**

The bound copy of a DBRM that is identified in the member clause.

**plan name**

The name of an application plan.

**P-lock** See physical lock.

**point of consistency**

A time when all recoverable data that an application accesses is consistent with other data. The term point of consistency is synonymous with sync point or commit point.

**policy** See CFRM policy.

**postponed abort UR**

A unit of recovery that was in-flight or in-abort, was interrupted by system failure or cancellation, and did not complete backout during restart.

**precision**

In SQL, the total number of digits in a decimal number (called the *size* in the C language). In the C language, the number of digits to the right of the decimal point (called the *scale* in SQL). The DB2 information uses the SQL terms.

**precompilation**

A processing of application programs containing SQL statements that takes place before compilation. SQL statements are replaced with statements that are recognized by the host language compiler. Output from this precompilation includes source code that can be submitted to the compiler and the database request module (DBRM) that is input to the bind process.

**predicate**

An element of a search condition that expresses or implies a comparison operation.

**prefix** A code at the beginning of a message or record.

**preformat**

The process of preparing a VSAM linear data set for DB2 use, by writing specific data patterns.

**prepare**

The first phase of a two-phase commit process in which all participants are requested to prepare for commit.

**prepared SQL statement**

A named object that is the executable

form of an SQL statement that has been processed by the PREPARE statement.

**primary authorization ID**

The authorization ID that is used to identify the application process to DB2.

**primary group buffer pool**

For a duplexed group buffer pool, the structure that is used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The z/OS equivalent is *old* structure. Compare with secondary group buffer pool.

**primary index**

An index that enforces the uniqueness of a primary key.

**primary key**

In a relational database, a unique, nonnull key that is part of the definition of a table. A table cannot be defined as a parent unless it has a unique key or primary key.

**principal**

An entity that can communicate securely with another entity. In Kerberos, principals are represented as entries in the Kerberos registry database and include users, servers, computers, and others.

**principal name**

The name by which a principal is known to the DCE security services.

**privilege**

The capability of performing a specific function, sometimes on a specific object. See also explicit privilege.

**privilege set**

- For the installation SYSADM ID, the set of all possible privileges.
- For any other authorization ID, including the PUBLIC authorization ID, the set of all privileges that are recorded for that ID in the DB2 catalog.

**process**

In DB2, the unit to which DB2 allocates resources and locks. Sometimes called an application process, a process involves the execution of one or more programs. The execution of an SQL statement is always associated with some process. The means of initiating and terminating a process are dependent on the environment.

**program**

A single, compilable collection of executable statements in a programming language.

**program temporary fix (PTF)**

A solution or bypass of a problem that is diagnosed as a result of a defect in a current unaltered release of a licensed program. An authorized program analysis report (APAR) fix is corrective service for an existing problem. A PTF is preventive service for problems that might be encountered by other users of the product. A PTF is *temporary*, because a permanent fix is usually not incorporated into the product until its next release.

**protected conversation**

A VTAM conversation that supports two-phase commit flows.

**PSRCP**

See page set recovery pending.

**PTF** See program temporary fix.

**QSAM**

See queued sequential access method.

**query** A component of certain SQL statements that specifies a result table.

**query block**

The part of a query that is represented by one of the FROM clauses. Each FROM clause can have multiple query blocks, depending on DB2 processing of the query.

**query CP parallelism**

Parallel execution of a single query, which is accomplished by using multiple tasks. See also Sysplex query parallelism.

**query I/O parallelism**

Parallel access of data, which is accomplished by triggering multiple I/O requests within a single query.

**queued sequential access method (QSAM)**

An extended version of the basic sequential access method (BSAM). When this method is used, a queue of data blocks is formed. Input data blocks await processing, and output data blocks await transfer to auxiliary storage or to an output device.

**quiesce point**

A point at which data is consistent as a result of running the DB2 QUIESCE utility.

**RACF** Resource Access Control Facility. A component of the z/OS Security Server.

**range-partitioned table space**

A type of universal table space that is based on partitioning ranges and that contains a single table. Contrast with partition-by-growth table space. See also universal table space.

**RBA** See relative byte address.

**RCT** See resource control table.

**RDO** See resource definition online.

**read stability (RS)**

An isolation level that is similar to repeatable read but does not completely isolate an application process from all other concurrently executing application processes. See also cursor stabilityrepeatable read, and uncommitted read.

**rebind**

The creation of a new application plan for an application program that has been bound previously. If, for example, you have added an index for a table that your application accesses, you must rebind the application in order to take advantage of that index.

**rebuild**

The process of reallocating a coupling facility structure. For the shared communications area (SCA) and lock structure, the structure is repopulated; for the group buffer pool, changed pages are usually cast out to disk, and the new structure is populated only with changed pages that were not successfully cast out.

**record** The storage representation of a row or other data.

**record identifier (RID)**

A unique identifier that DB2 uses to identify a row of data in a table. Compare with row identifier.

**record identifier (RID) pool**

An area of main storage that is used for sorting record identifiers during list-prefetch processing.

**record length**

The sum of the length of all the columns in a table, which is the length of the data as it is physically stored in the database. Records can be fixed length or varying length, depending on how the columns are defined. If all columns are fixed-length columns, the record is a fixed-length record. If one or more columns are varying-length columns, the record is a varying-length record.

**Recoverable Resource Manager Services attachment facility (RRSAF)**

A DB2 subcomponent that uses Resource Recovery Services to coordinate resource commitment between DB2 and all other resource managers that also use RRS in a z/OS system.

**recovery**

The process of rebuilding databases after a system failure.

**recovery log**

A collection of records that describes the events that occur during DB2 execution and indicates their sequence. The recorded information is used for recovery in the event of a failure during DB2 execution.

**recovery manager**

A subcomponent that supplies coordination services that control the interaction of DB2 resource managers during commit, abort, checkpoint, and restart processes. The recovery manager also supports the recovery mechanisms of other subsystems (for example, IMS) by acting as a participant in the other subsystem's process for protecting data that has reached a point of consistency.

A coordinator or a participant (or both), in the execution of a two-phase commit, that can access a recovery log that maintains the state of the logical unit of work and names the immediate upstream coordinator and downstream participants.

**recovery pending (RECP)**

A condition that prevents SQL access to a table space that needs to be recovered.

**recovery token**

An identifier for an element that is used in recovery (for example, NID or URID).

**RECP** See recovery pending.

**redo** A state of a unit of recovery that indicates that changes are to be reapplied to the disk media to ensure data integrity.

**reentrant code**

Executable code that can reside in storage as one shared copy for all threads. Reentrant code is not self-modifying and provides separate storage areas for each thread. See also threadsafe.

**referential constraint**

The requirement that nonnull values of a designated foreign key are valid only if they equal values of the primary key of a designated table.

**referential cycle**

A set of referential constraints such that each base table in the set is a descendent of itself. The tables that are involved in a referential cycle are ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table.

**referential integrity**

The state of a database in which all values of all foreign keys are valid. Maintaining referential integrity requires the enforcement of referential constraints on all operations that change the data in a table on which the referential constraints are defined.

**referential structure**

A set of tables and relationships that includes at least one table and, for every table in the set, all the relationships in which that table participates and all the tables to which it is related.

**refresh age**

The time duration between the current time and the time during which a materialized query table was last refreshed.

**registry**

See registry database.

**registry database**

A database of security information about principals, groups, organizations, accounts, and security policies.

**relational database**

A database that can be perceived as a set of tables and manipulated in accordance with the relational model of data.

**relational database management system (RDBMS)**

A collection of hardware and software that organizes and provides access to a relational database.

**relational schema**

See SQL schema.

**relationship**

A defined connection between the rows of a table or the rows of two tables. A relationship is the internal representation of a referential constraint.

**relative byte address (RBA)**

The offset of a data record or control interval from the beginning of the storage space that is allocated to the data set or file to which it belongs.

**remigration**

The process of returning to a current release of DB2 following a fallback to a previous release. This procedure constitutes another migration process.

**remote**

Any object that is maintained by a remote DB2 subsystem (that is, by a DB2 subsystem other than the local one). A *remote view*, for example, is a view that is maintained by a remote DB2 subsystem. Contrast with local.

**remote subsystem**

Any relational DBMS, except the local subsystem, with which the user or application can communicate. The subsystem need not be remote in any physical sense, and might even operate on the same processor under the same z/OS system.

**reoptimization**

The DB2 process of reconsidering the access path of an SQL statement at run time; during reoptimization, DB2 uses the values of host variables, parameter markers, or special registers.

**reordered row format**

A row format that facilitates improved performance in retrieval of rows that have varying-length columns. DB2 rearranges the column order, as defined in the CREATE TABLE statement, so that the fixed-length columns are stored at the beginning of the row and the

varying-length columns are stored at the end of the row. Contrast with basic row format.

**REORG pending (REORP)**

A condition that restricts SQL access and most utility access to an object that must be reorganized.

**REORP**

See REORG pending.

**repeatable read (RR)**

The isolation level that provides maximum protection from other executing application programs. When an application program executes with repeatable read protection, rows that the program references cannot be changed by other programs until the program reaches a commit point. See also cursor stability, read stability, and uncommitted read.

**repeating group**

A situation in which an entity includes multiple attributes that are inherently the same. The presence of a repeating group violates the requirement of first normal form. In an entity that satisfies the requirement of first normal form, each attribute is independent and unique in its meaning and its name. See also normalization.

**replay detection mechanism**

A method that allows a principal to detect whether a request is a valid request from a source that can be trusted or whether an untrustworthy entity has captured information from a previous exchange and is replaying the information exchange to gain access to the principal.

**request commit**

The vote that is submitted to the prepare phase if the participant has modified data and is prepared to commit or roll back.

**requester**

The source of a request to access data at a remote server. In the DB2 environment, the requester function is provided by the distributed data facility.

**resource**

The object of a lock or claim, which could be a table space, an index space, a data partition, an index partition, or a logical partition.

**resource allocation**

The part of plan allocation that deals specifically with the database resources.

**resource control table**

A construct of previous versions of the CICS attachment facility that defines authorization and access attributes for transactions or transaction groups. Beginning in CICS Transaction Server Version 1.3, resources are defined by using resource definition online instead of the resource control table. See also resource definition online.

**resource definition online (RDO)**

The recommended method of defining resources to CICS by creating resource definitions interactively, or by using a utility, and then storing them in the CICS definition data set. In earlier releases of CICS, resources were defined by using the resource control table (RCT), which is no longer supported.

**resource limit facility (RLF)**

A portion of DB2 code that prevents dynamic manipulative SQL statements from exceeding specified time limits. The resource limit facility is sometimes called the governor.

**resource limit specification table (RLST)**

A site-defined table that specifies the limits to be enforced by the resource limit facility.

**resource manager**

- A function that is responsible for managing a particular resource and that guarantees the consistency of all updates made to recoverable resources within a logical unit of work. The resource that is being managed can be physical (for example, disk or main storage) or logical (for example, a particular type of system service).
- A participant, in the execution of a two-phase commit, that has recoverable resources that could have been modified. The resource manager has access to a recovery log so that it can commit or roll back the effects of the logical unit of work to the recoverable resources.

**restart pending (RESTP)**

A restrictive state of a page set or

partition that indicates that restart (backout) work needs to be performed on the object.

**RESTP**

See restart pending.

**result set**

The set of rows that a stored procedure returns to a client application.

**result set locator**

A 4-byte value that DB2 uses to uniquely identify a query result set that a stored procedure returns.

**result table**

The set of rows that are specified by a SELECT statement.

**retained lock**

A MODIFY lock that a DB2 subsystem was holding at the time of a subsystem failure. The lock is retained in the coupling facility lock structure across a DB2 for z/OS failure.

**RID** See record identifier.

**RID pool**

See record identifier pool.

**right outer join**

The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of the second join operand. See also join, equijoin, full outer join, inner join, left outer join, and outer join.

**RLF** See resource limit facility.

**RLST** See resource limit specification table.

**role** A database entity that groups together one or more privileges and that can be assigned to a primary authorization ID or to PUBLIC. The role is available only in a trusted context.

**rollback**

The process of restoring data that was changed by SQL statements to the state at its last commit point. All locks are freed. Contrast with commit.

**root page**

The index page that is at the highest level (or the beginning point) in an index.

**routine**

A database object that encapsulates

procedural logic and SQL statements, is stored on the database server, and can be invoked from an SQL statement or by using the CALL statement. The main classes of routines are procedures and functions.

**row** The horizontal component of a table. A row consists of a sequence of values, one for each column of the table.

**row identifier (ROWID)**

A value that uniquely identifies a row. This value is stored with the row and never changes.

**row lock**

A lock on a single row of data.

**row-positioned fetch orientation**

The specification of the desired placement of the cursor as part of a FETCH statement, with respect to a single row (for example, NEXT, LAST, or ABSOLUTE *n*). Contrast with rowset-positioned fetch orientation.

**rowset**

A set of rows for which a cursor position is established.

**rowset cursor**

A cursor that is defined so that one or more rows can be returned as a rowset for a single FETCH statement, and the cursor is positioned on the set of rows that is fetched.

**rowset-positioned fetch orientation**

The specification of the desired placement of the cursor as part of a FETCH statement, with respect to a rowset (for example, NEXT ROWSET, LAST ROWSET, or ROWSET STARTING AT ABSOLUTE *n*). Contrast with row-positioned fetch orientation.

**row trigger**

A trigger that is defined with the trigger granularity FOR EACH ROW.

**RRSAF**

See Recoverable Resource Manager Services attachment facility.

**RS**

See read stability.

**savepoint**

A named entity that represents the state of data and schemas at a particular point in time within a unit of work.

**SBCS** See single-byte character set.

**SCA** See shared communications area.

**scalar function**

An SQL operation that produces a single value from another value and is expressed as a function name, followed by a list of arguments that are enclosed in parentheses.

**scale**

In SQL, the number of digits to the right of the decimal point (called the precision in the C language). The DB2 information uses the SQL definition.

**schema**

The organization or structure of a database.

A collection of, and a way of qualifying, database objects such as tables, views, routines, indexes or triggers that define a database. A database schema provides a logical classification of database objects.

**scrollability**

The ability to use a cursor to fetch in either a forward or backward direction. The FETCH statement supports multiple fetch orientations to indicate the new position of the cursor. See also fetch orientation.

**scrollable cursor**

A cursor that can be moved in both a forward and a backward direction.

**search condition**

A criterion for selecting rows from a table. A search condition consists of one or more predicates.

**secondary authorization ID**

An authorization ID that has been associated with a primary authorization ID by an authorization exit routine.

**secondary group buffer pool**

For a duplexed group buffer pool, the structure that is used to back up changed pages that are written to the primary group buffer pool. No page registration or cross-invalidation occurs using the secondary group buffer pool. The z/OS equivalent is *new* structure.

**secondary index**

A nonpartitioning index that is useful for enforcing a uniqueness constraint, for clustering data, or for providing access

paths to data for queries. A secondary index can be partitioned or nonpartitioned. See also data-partitioned secondary index (DPSI) and nonpartitioned secondary index (NPSI).

#### **section**

The segment of a plan or package that contains the executable structures for a single SQL statement. For most SQL statements, one section in the plan exists for each SQL statement in the source program. However, for cursor-related statements, the DECLARE, OPEN, FETCH, and CLOSE statements reference the same section because they each refer to the SELECT statement that is named in the DECLARE CURSOR statement. SQL statements such as COMMIT, ROLLBACK, and some SET statements do not use a section.

#### **security label**

A classification of users' access to objects or data rows in a multilevel security environment."

#### **segment**

A group of pages that holds rows of a single table. See also segmented table space.

#### **segmented table space**

A table space that is divided into equal-sized groups of pages called segments. Segments are assigned to tables so that rows of different tables are never stored in the same segment. Contrast with partitioned table space and universal table space.

#### **self-referencing constraint**

A referential constraint that defines a relationship in which a table is a dependent of itself.

#### **self-referencing table**

A table with a self-referencing constraint.

#### **sensitive cursor**

A cursor that is sensitive to changes that are made to the database after the result table has been materialized.

#### **sequence**

A user-defined object that generates a sequence of numeric values according to user specifications.

#### **sequential data set**

A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 database utilities require sequential data sets.

#### **sequential prefetch**

A mechanism that triggers consecutive asynchronous I/O operations. Pages are fetched before they are required, and several pages are read with a single I/O operation.

#### **serialized profile**

A Java object that contains SQL statements and descriptions of host variables. The SQLJ translator produces a serialized profile for each connection context.

#### **server**

The target of a request from a remote requester. In the DB2 environment, the server function is provided by the distributed data facility, which is used to access DB2 data from remote applications.

#### **service class**

An eight-character identifier that is used by the z/OS Workload Manager to associate user performance goals with a particular DDF thread or stored procedure. A service class is also used to classify work on parallelism assistants.

#### **service request block**

A unit of work that is scheduled to execute.

#### **session**

A link between two nodes in a VTAM network.

#### **session protocols**

The available set of SNA communication requests and responses.

#### **set operator**

The SQL operators UNION, EXCEPT, and INTERSECT corresponding to the relational operators union, difference, and intersection. A set operator derives a result table by combining two other result tables.

#### **shared communications area (SCA)**

A coupling facility list structure that a DB2 data sharing group uses for inter-DB2 communication.

**share lock**

A lock that prevents concurrently executing application processes from changing data, but not from reading data. Contrast with exclusive lock.

**shift-in character**

A special control character (X'0F') that is used in EBCDIC systems to denote that the subsequent bytes represent SBCS characters. See also shift-out character.

**shift-out character**

A special control character (X'0E') that is used in EBCDIC systems to denote that the subsequent bytes, up to the next shift-in control character, represent DBCS characters. See also shift-in character.

**sign-on**

A request that is made on behalf of an individual CICS or IMS application process by an attachment facility to enable DB2 to verify that it is authorized to use DB2 resources.

**simple page set**

A nonpartitioned page set. A simple page set initially consists of a single data set (page set piece). If and when that data set is extended to 2 GB, another data set is created, and so on, up to a total of 32 data sets. DB2 considers the data sets to be a single contiguous linear address space containing a maximum of 64 GB. Data is stored in the next available location within this address space without regard to any partitioning scheme.

**simple table space**

A table space that is neither partitioned nor segmented. Creation of simple table spaces is not supported in DB2 Version 9.1 for z/OS. Contrast with partitioned table space, segmented table space, and universal table space.

**single-byte character set (SBCS)**

A set of characters in which each character is represented by a single byte. Contrast with double-byte character set or multibyte character set.

**single-precision floating point number**

A 32-bit approximate representation of a real number.

**SMP/E**

See System Modification Program/Extended.

**SNA** See Systems Network Architecture.

**SNA network**

The part of a network that conforms to the formats and protocols of Systems Network Architecture (SNA).

**socket** A callable TCP/IP programming interface that TCP/IP network applications use to communicate with remote TCP/IP partners.

**sourced function**

A function that is implemented by another built-in or user-defined function that is already known to the database manager. This function can be a scalar function or an aggregate function; it returns a single value from a set of values (for example, MAX or AVG). Contrast with built-in function, external function, and SQL function.

**source program**

A set of host language statements and SQL statements that is processed by an SQL precompiler.

**source table**

A table that can be a base table, a view, a table expression, or a user-defined table function.

**source type**

An existing type that DB2 uses to represent a distinct type.

**space** A sequence of one or more blank characters.

**special register**

A storage area that DB2 defines for an application process to use for storing information that can be referenced in SQL statements. Examples of special registers are SESSION\_USER and CURRENT DATE.

**specific function name**

A particular user-defined function that is known to the database manager by its specific name. Many specific user-defined functions can have the same function name. When a user-defined function is defined to the database, every function is assigned a specific name that is unique within its schema. Either the user can provide this name, or a default name is used.

**SPUFI** See SQL Processor Using File Input.

**SQL** See Structured Query Language.

**SQL authorization ID (SQL ID)**

The authorization ID that is used for checking dynamic SQL statements in some situations.

**SQLCA**

See SQL communication area.

**SQL communication area (SQLCA)**

A structure that is used to provide an application program with information about the execution of its SQL statements.

**SQL connection**

An association between an application process and a local or remote application server or database server.

**SQLDA**

See SQL descriptor area.

**SQL descriptor area (SQLDA)**

A structure that describes input variables, output variables, or the columns of a result table.

**SQL escape character**

The symbol that is used to enclose an SQL delimited identifier. This symbol is the double quotation mark ("). See also escape character.

**SQL function**

A user-defined function in which the CREATE FUNCTION statement contains the source code. The source code is a single SQL expression that evaluates to a single value. The SQL user-defined function can return the result of an expression. See also built-in function, external function, and sourced function.

**SQL ID**

See SQL authorization ID.

**SQLJ** Structured Query Language (SQL) that is embedded in the Java programming language.

**SQL path**

An ordered list of schema names that are used in the resolution of unqualified references to user-defined functions, distinct types, and stored procedures. In dynamic SQL, the SQL path is found in the CURRENT PATH special register. In static SQL, it is defined in the PATH bind option.

**SQL PL**

See SQL procedural language.

**SQL procedural language (SQL PL)**

A language extension of SQL that consists of statements and language elements that can be used to implement procedural logic in SQL statements. SQL PL provides statements for declaring variables and condition handlers, assigning values to variables, and for implementing procedural logic. See also inline SQL PL.

**SQL procedure**

A user-written program that can be invoked with the SQL CALL statement. An SQL procedure is written in the SQL procedural language. Two types of SQL procedures are supported: external SQL procedures and native SQL procedures. See also external procedure and native SQL procedure.

**SQL processing conversation**

Any conversation that requires access of DB2 data, either through an application or by dynamic query requests.

**SQL Processor Using File Input (SPUFI)**

A facility of the TSO attachment subcomponent that enables the DB2I user to execute SQL statements without embedding them in an application program.

**SQL return code**

Either SQLCODE or SQLSTATE.

**SQL routine**

A user-defined function or stored procedure that is based on code that is written in SQL.

**SQL schema**

A collection of database objects such as tables, views, indexes, functions, distinct types, schemas, or triggers that defines a database. An SQL schema provides a logical classification of database objects.

**SQL statement coprocessor**

An alternative to the DB2 precompiler that lets the user process SQL statements at compile time. The user invokes an SQL statement coprocessor by specifying a compiler option.

**SQL string delimiter**

A symbol that is used to enclose an SQL string constant. The SQL string delimiter

is the apostrophe ('), except in COBOL applications, where the user assigns the symbol, which is either an apostrophe or a double quotation mark (").

**SRB** See service request block.

**stand-alone**

An attribute of a program that means that it is capable of executing separately from DB2, without using DB2 services.

**star join**

A method of joining a dimension column of a fact table to the key column of the corresponding dimension table. See also join, dimension, and star schema.

**star schema**

The combination of a fact table (which contains most of the data) and a number of dimension tables. See also star join, dimension, and dimension table.

**statement handle**

In DB2 ODBC, the data object that contains information about an SQL statement that is managed by DB2 ODBC. This includes information such as dynamic arguments, bindings for dynamic arguments and columns, cursor information, result values, and status information. Each statement handle is associated with the connection handle.

**statement string**

For a dynamic SQL statement, the character string form of the statement.

**statement trigger**

A trigger that is defined with the trigger granularity FOR EACH STATEMENT.

**static cursor**

A named control structure that does not change the size of the result table or the order of its rows after an application opens the cursor. Contrast with dynamic cursor.

**static SQL**

SQL statements, embedded within a program, that are prepared during the program preparation process (before the program is executed). After being prepared, the SQL statement does not change (although values of variables that are specified by the statement might change).

**storage group**

A set of storage objects on which DB2 for z/OS data can be stored. A storage object can have an SMS data class, a management class, a storage class, and a list of volume serial numbers.

**stored procedure**

A user-written application program that can be invoked through the use of the SQL CALL statement. Stored procedures are sometimes called procedures.

**string** See binary string, character string, or graphic string.

**strong typing**

A process that guarantees that only user-defined functions and operations that are defined on a distinct type can be applied to that type. For example, you cannot directly compare two currency types, such as Canadian dollars and U.S. dollars. But you can provide a user-defined function to convert one currency to the other and then do the comparison.

**structure**

- A name that refers collectively to different types of DB2 objects, such as tables, databases, views, indexes, and table spaces.
- A construct that uses z/OS to map and manage storage on a coupling facility. See also cache structure, list structure, or lock structure.

**Structured Query Language (SQL)**

A standardized language for defining and manipulating data in a relational database.

**structure owner**

In relation to group buffer pools, the DB2 member that is responsible for the following activities:

- Coordinating rebuild, checkpoint, and damage assessment processing
- Monitoring the group buffer pool threshold and notifying castout owners when the threshold has been reached

**subcomponent**

A group of closely related DB2 modules that work together to provide a general function.

**subject table**

The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

**subquery**

A SELECT statement within the WHERE or HAVING clause of another SQL statement; a nested SQL statement.

**subselect**

That form of a query that includes only a SELECT clause, FROM clause, and optionally a WHERE clause, GROUP BY clause, HAVING clause, ORDER BY clause, or FETCH FIRST clause.

**substitution character**

A unique character that is substituted during character conversion for any characters in the source program that do not have a match in the target coding representation.

**subsystem**

In z/OS, a service provider that performs one or many functions, but does nothing until a request is made. For example, each WebSphere MQ for z/OS queue manager or instance of a DB2 for z/OS database management system is a z/OS subsystem.

**surrogate pair**

A coded representation for a single character that consists of a sequence of two 16-bit code units, in which the first value of the pair is a high-surrogate code unit in the range U+D800 through U+DBFF, and the second value is a low-surrogate code unit in the range U+DC00 through U+DFFF. Surrogate pairs provide an extension mechanism for encoding 917 476 characters without requiring the use of 32-bit characters.

**SVC dump**

A dump that is issued when a z/OS or a DB2 functional recovery routine detects an error.

**sync point**

See commit point.

**syncpoint tree**

The tree of recovery managers and resource managers that are involved in a logical unit of work, starting with the recovery manager, that make the final commit decision.

**synonym**

In SQL, an alternative name for a table or view. Synonyms can be used to refer only to objects at the subsystem in which the synonym is defined. A synonym cannot be qualified and can therefore not be used by other users. Contrast with alias.

**Sysplex**

See Parallel Sysplex.

**Sysplex query parallelism**

Parallel execution of a single query that is accomplished by using multiple tasks on more than one DB2 subsystem. See also query CP parallelism.

**system administrator**

The person at a computer installation who designs, controls, and manages the use of the computer system.

**system agent**

A work request that DB2 creates such as prefetch processing, deferred writes, and service tasks. See also allied agent.

**system authorization ID**

The primary DB2 authorization ID that is used to establish a trusted connection. A system authorization ID is derived from the system user ID that is provided by an external entity, such as a middleware server.

**system conversation**

The conversation that two DB2 subsystems must establish to process system messages before any distributed processing can begin.

**system-defined routine**

In DB2 10 for z/OS and later, an object (function or procedure) for which system DBADM and SQLADM authorities have implicit execute privilege on the routine and any packages executed within the routine.

**System Modification Program/Extended (SMP/E)**

A z/OS tool for making software changes in programming systems (such as DB2) and for controlling those changes.

**system period**

A pair of columns with system-maintained values that indicate the period of time when a row is valid.

| **system-period data versioning**  
| Automatic maintenance of historical data  
| by DB2 using a system period.

| **system-period temporal table**  
| A table that is defined with system-period  
| data versioning.

**Systems Network Architecture (SNA)**  
The description of the logical structure,  
formats, protocols, and operational  
sequences for transmitting information  
through and controlling the configuration  
and operation of networks.

**table** A named data object consisting of a  
specific number of columns and some  
number of unordered rows. See also base  
table or temporary table. Contrast with  
auxiliary table, clone table, materialized  
query table, result table, and transition  
table.

**table-controlled partitioning**  
A type of partitioning in which partition  
boundaries for a partitioned table are  
controlled by values that are defined in  
the CREATE TABLE statement.

**table function**  
A function that receives a set of  
arguments and returns a table to the SQL  
statement that references the function. A  
table function can be referenced only in  
the FROM clause of a subselect.

**table locator**  
A mechanism that allows access to trigger  
tables in SQL or from within user-defined  
functions. A table locator is a fullword  
integer value that represents a transition  
table.

**table space**  
A page set that is used to store the  
records in one or more tables. See also  
partitioned table space, segmented table  
space, and universal table space.

**table space set**  
A set of table spaces and partitions that  
should be recovered together for one of  
the following reasons:

- Each of them contains a table that is a  
parent or descendent of a table in one  
of the others.
- The set contains a base table and  
associated auxiliary tables.

A table space set can contain both types  
of relationships.

**task control block (TCB)**  
A z/OS control block that is used to  
communicate information about tasks  
within an address space that is connected  
to a subsystem. See also address space  
connection.

**TB** Terabyte. A value of 1 099 511 627 776  
bytes.

**TCB** See task control block.

**TCP/IP**  
A network communication protocol that  
computer systems use to exchange  
information across telecommunication  
links.

**TCP/IP port**  
A 2-byte value that identifies an end user  
or a TCP/IP network application within a  
TCP/IP host.

**template**  
A DB2 utilities output data set descriptor  
that is used for dynamic allocation. A  
template is defined by the TEMPLATE  
utility control statement.

| **temporal table**  
| A table which records the period of time  
| when a row is valid. See also  
| system-period temporal table,  
| application-period temporal table, and  
| bitemporal table.

**temporary table**  
A table that holds temporary data.  
Temporary tables are useful for holding  
or sorting intermediate results from  
queries that contain a large number of  
rows. The two types of temporary table,  
which are created by different SQL  
statements, are the created temporary  
table and the declared temporary table.  
Contrast with result table. See also created  
temporary table and declared temporary  
table.

| **textual XML format**  
| A representation of XML data that uses  
| character values, an approach that allows  
| for direct reading by people.

**thread** See DB2 thread.

**threadsafe**  
A characteristic of code that allows

multithreading both by providing private storage areas for each thread, and by properly serializing shared (global) storage areas.

**three-part name**

The full name of a table, view, or alias. It consists of a location name, a schema name, and an object name, separated by a period.

**time** A three-part value that designates a time of day in hours, minutes, and seconds.

**timeout**

Abnormal termination of either the DB2 subsystem or of an application because of the unavailability of resources. Installation specifications are set to determine both the amount of time DB2 is to wait for IRLM services after starting, and the amount of time IRLM is to wait if a resource that an application requests is unavailable. If either of these time specifications is exceeded, a timeout is declared.

**Time-Sharing Option (TSO)**

An option in z/OS that provides interactive time sharing from remote terminals.

**timestamp**

A seven-part value that consists of a date and time. The timestamp is expressed in years, months, days, hours, minutes, seconds, and microseconds.

**timestamp with time zone**

A two-part value that consists of a timestamp and time zone. The timestamp with time zone is expressed in years, months, days, hours, minutes, seconds, microseconds, time zone hours, and time zone minutes.

**trace** A DB2 facility that provides the ability to monitor and collect DB2 monitoring, auditing, performance, accounting, statistics, and serviceability (global) data.

**transaction**

An atomic series of SQL statements that make up a logical unit of work. All of the data modifications made during a transaction are either committed together as a unit or rolled back as a unit.

**transaction lock**

A lock that is used to control concurrent execution of SQL statements.

**transaction program name**

In SNA LU 6.2 conversations, the name of the program at the remote logical unit that is to be the other half of the conversation.

**transition table**

A temporary table that contains all the affected rows of the subject table in their state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the table of changed rows in the old state or the new state. Contrast with auxiliary table, base table, clone table, and materialized query table.

**transition variable**

A variable that contains a column value of the affected row of the subject table in its state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the set of old values or the set of new values.

**tree structure**

A data structure that represents entities in nodes, with a most one parent node for each node, and with only one root node.

**trigger**

A database object that is associated with a single base table or view and that defines a rule. The rule consists of a set of SQL statements that run when an insert, update, or delete database operation occurs on the associated base table or view.

**trigger activation**

The process that occurs when the trigger event that is defined in a trigger definition is executed. Trigger activation consists of the evaluation of the triggered action condition and conditional execution of the triggered SQL statements.

**trigger activation time**

An indication in the trigger definition of whether the trigger should be activated before or after the triggered event.

**trigger body**

The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to

true. A trigger body is also called triggered SQL statements.

**trigger cascading**

The process that occurs when the triggered action of a trigger causes the activation of another trigger.

**triggered action**

The SQL logic that is performed when a trigger is activated. The triggered action consists of an optional triggered action condition and a set of triggered SQL statements that are executed only if the condition evaluates to true.

**triggered action condition**

An optional part of the triggered action. This Boolean condition appears as a WHEN clause and specifies a condition that DB2 evaluates to determine if the triggered SQL statements should be executed.

**triggered SQL statements**

The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. Triggered SQL statements are also called the trigger body.

**trigger granularity**

In SQL, a characteristic of a trigger, which determines whether the trigger is activated:

- Only once for the triggering SQL statement
- Once for each row that the SQL statement modifies

**triggering event**

The specified operation in a trigger definition that causes the activation of that trigger. The triggering event is comprised of a triggering operation (insert, update, or delete) and a subject table or view on which the operation is performed.

**triggering SQL operation**

The SQL operation that causes a trigger to be activated when performed on the subject table or view.

**trigger package**

A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

**trust attribute**

An attribute on which to establish trust. A trusted relationship is established based on one or more trust attributes.

**trusted connection**

A database connection whose attributes match the attributes of a unique trusted context defined at the DB2 database server.

**trusted connection reuse**

The ability to switch the current user ID on a trusted connection to a different user ID.

**trusted context**

A database security object that enables the establishment of a trusted relationship between a DB2 database management system and an external entity.

**trusted context default role**

A role associated with a trusted context. The privileges granted to the trusted context default role can be acquired only when a trusted connection based on the trusted context is established or reused.

**trusted context user**

A user ID to which switching the current user ID on a trusted connection is permitted.

**trusted context user-specific role**

A role that is associated with a specific trusted context user. It overrides the trusted context default role if the current user ID on the trusted connection matches the ID of the specific trusted context user.

**trusted relationship**

A privileged relationship between two entities such as a middleware server and a database server. This relationship allows for a unique set of interactions between the two entities that would be impossible otherwise.

**TSO** See Time-Sharing Option.

**TSO attachment facility**

A DB2 facility consisting of the DSN command processor and DB2I. Applications that are not written for the CICS or IMS environments can run under the TSO attachment facility.

**typed parameter marker**

A parameter marker that is specified along with its target data type. It has the general form:

CAST(? AS data-type)

**type 2 indexes**

Indexes that are created on a release of DB2 after Version 7 or that are specified as type 2 indexes in Version 4 or later.

**UCS-2** Universal Character Set, coded in 2 octets, which means that characters are represented in 16-bits per character.

**UDF** See user-defined function.

**UDT** User-defined data type. In DB2 for z/OS, the term distinct type is used instead of user-defined data type. See distinct type.

**uncommitted read (UR)**

The isolation level that allows an application to read uncommitted data. See also cursor stability, read stability, and repeatable read.

**underlying view**

The view on which another view is directly or indirectly defined.

**undo** A state of a unit of recovery that indicates that the changes that the unit of recovery made to recoverable DB2 resources must be backed out.

**Unicode**

A standard that parallels the ISO-10646 standard. Several implementations of the Unicode standard exist, all of which have the ability to represent a large percentage of the characters that are contained in the many scripts that are used throughout the world.

| **union** An SQL operation that involves the  
| UNION set operator, which combines the  
| results of two SELECT statements. Unions  
| are often used to merge lists of values  
| that are obtained from two tables.

**unique constraint**

An SQL rule that no two values in a primary key, or in the key of a unique index, can be the same.

**unique index**

An index that ensures that no identical key values are stored in a column or a set of columns in a table.

**unit of recovery (UOR)**

A recoverable sequence of operations within a single resource manager, such as an instance of DB2. Contrast with unit of work.

**unit of work (UOW)**

A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations. In a multisite update operation, a single unit of work can include several *units of recovery*. Contrast with unit of recovery.

| **universal table space**

| A table space that is both segmented and  
| partitioned. Contrast with partitioned  
| table space, segmented table space,  
| partition-by-growth table space, and  
| range-partitioned table space.

**unlock**

The act of releasing an object or system resource that was previously locked and returning it to general availability within DB2.

**untyped parameter marker**

A parameter marker that is specified without its target data type. It has the form of a single question mark (?).

**updatability**

The ability of a cursor to perform positioned updates and deletes. The updatability of a cursor can be influenced by the SELECT statement and the cursor sensitivity option that is specified on the DECLARE CURSOR statement.

**update hole**

The location on which a cursor is positioned when a row in a result table is fetched again and the new values no longer satisfy the search condition. See also delete hole.

**update trigger**

A trigger that is defined with the triggering SQL operation update.

**UR** See uncommitted read.

**user-defined data type (UDT)**

See distinct type.

**user-defined function (UDF)**

A function that is defined to DB2 by using the CREATE FUNCTION statement and that can be referenced thereafter in SQL statements. A user-defined function can be an external function, a sourced function, or an SQL function. Contrast with built-in function.

**user view**

In logical data modeling, a model or representation of critical information that the business requires.

**UTF-16**

Unicode Transformation Format, 16-bit encoding form, which is designed to provide code values for over a million characters and a superset of UCS-2. The CCSID value for data in UTF-16 format is 1200. DB2 for z/OS supports UTF-16 in graphic data fields.

**UTF-8**

Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems. The CCSID value for data in UTF-8 format is 1208. DB2 for z/OS supports UTF-8 in mixed data fields.

**value** The smallest unit of data that is manipulated in SQL.

**variable**

A data element that specifies a value that can be changed. A COBOL elementary data item is an example of a host variable. Contrast with constant.

**variant function**

See nondeterministic function.

**varying-length string**

A character, graphic, or binary string whose length varies within set limits. Contrast with fixed-length string.

**version**

A member of a set of similar programs, DBRMs, packages, or LOBs.

- **A version of a program** is the source code that is produced by precompiling the program. The program version is identified by the program name and a timestamp (consistency token).
- **A version of an SQL procedural language routine** is produced by

issuing the CREATE or ALTER PROCEDURE statement for a native SQL procedure.

- **A version of a DBRM** is the DBRM that is produced by precompiling a program. The DBRM version is identified by the same program name and timestamp as a corresponding program version.
- **A version of an application package** is the result of binding a DBRM within a particular database system. The application package version is identified by the same program name and consistency token as the DBRM.
- **A version of a LOB** is a copy of a LOB value at a point in time. The version number for a LOB is stored in the auxiliary index entry for the LOB.
- **A version of a record** is a copy of the record at a point in time.

**view** A logical table that consists of data that is generated by a query. A view can be based on one or more underlying base tables or views, and the data in a view is determined by a SELECT statement that is run on the underlying base tables or views.

**Virtual Storage Access Method (VSAM)**

An access method for direct or sequential processing of fixed- and varying-length records on disk devices.

**Virtual Telecommunications Access Method (VTAM)**

An IBM licensed program that controls communication and the flow of data in an SNA network (in z/OS).

**volatile table**

A table for which SQL operations choose index access whenever possible.

**VSAM**

See Virtual Storage Access Method.

**VTAM**

See Virtual Telecommunications Access Method.

**warm start**

The normal DB2 restart process, which involves reading and processing log records so that data that is under the control of DB2 is consistent. Contrast with cold start.

**WLM application environment**

A z/OS Workload Manager attribute that is associated with one or more stored procedures. The WLM application environment determines the address space in which a given DB2 stored procedure runs.

**WLM enclave**

A construct that can span multiple dispatchable units (service request blocks and tasks) in multiple address spaces, allowing them to be reported on and managed by WLM as part of a single work request.

**write to operator (WTO)**

An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that might need to be corrected (in z/OS).

**WTO** See write to operator.

**WTOR**

Write to operator (WTO) with reply.

**XCF** See cross-system coupling facility.

**XES** See cross-system extended services.

**XML** See Extensible Markup Language.

**XML attribute**

A name-value pair within a tagged XML element that modifies certain features of the element.

**XML column**

A column of a table that stores XML values and is defined using the data type XML. The XML values that are stored in XML columns are internal representations of well-formed XML documents.

**XML data type**

A data type for XML values.

**XML element**

A logical structure in an XML document that is delimited by a start and an end tag. Anything between the start tag and the end tag is the content of the element.

**XML index**

An index on an XML column that provides efficient access to nodes within an XML document by providing index keys that are based on XML patterns.

**XML lock**

A column-level lock for XML data. The operation of XML locks is similar to the operation of LOB locks.

**XML node**

The smallest unit of valid, complete structure in a document. For example, a node can represent an element, an attribute, or a text string.

**XML node ID index**

An implicitly created index, on an XML table that provides efficient access to XML documents and navigation among multiple XML data rows in the same document.

**XML pattern**

A slash-separated list of element names, an optional attribute name (at the end), or kind tests, that describe a path within an XML document in an XML column. The pattern is a restrictive form of path expressions, and it selects nodes that match the specifications. XML patterns are specified to create indexes on XML columns in a database.

**XML publishing function**

A function that returns an XML value from SQL values. An XML publishing function is also known as an XML constructor.

**XML schema**

In XML, a mechanism for describing and constraining the content of XML files by indicating which elements are allowed and in which combinations. XML schemas are an alternative to document type definitions (DTDs) and can be used to extend functionality in the areas of data typing, inheritance, and presentation.

**XML schema repository (XSR)**

A repository that allows the DB2 database system to store XML schemas. When registered with the XSR, these objects have a unique identifier and can be used to validate XML instance documents.

**XML serialization function**

A function that returns a serialized XML string from an XML value.

**XML table**

An auxiliary table that is implicitly created when an XML column is added to

| a base table. This table stores the XML  
| data, and the column in the base table  
| points to it.

| **XML table space**

A table space that is implicitly created when an XML column is added to a base table. The table space stores the XML table. If the base table is partitioned, one partitioned table space exists for each XML column of data.

**X/Open**

An independent, worldwide open systems organization that is supported by most of the world's largest information systems suppliers, user organizations, and software companies. X/Open's goal is to increase the portability of applications by combining existing and emerging standards.

**XRF** See Extended Recovery Facility.

| **XSR** See XML schema repository.

| **zIIP** See IBM System z9 Integrated Processor.

**z/OS** An operating system for the System z product line that supports 64-bit real and virtual storage.

**z/OS Distributed Computing Environment (z/OS DCE)** A set of technologies that are provided by the Open Software Foundation to implement distributed computing.



---

# Index

## A

ACCESS DATABASE MODE  
  NGBPDEP 141  
  OPEN 141  
access methods  
  SNA  
    descriptions 59  
  TCP/IP  
    descriptions 34  
access methods, SNA  
  overview 59  
access methods, TCP/IP  
  overview 34  
access path, effects 224  
accessibility  
  keyboard viii  
  shortcut keys viii  
accounting trace  
  global lock contention 175  
advisory restart-pending 131  
affinity  
  for online utility jobs 82  
affinity, breaking between systems 72  
ALIAS option  
  of DSNJU003 utility 51  
ALTER GROUPBUFFERPOOL command  
  change group buffer pools 220, 221  
  GBPCACHE option 223  
application design recommendations 138  
AREST status 131  
authorization  
  commands 78  
  controlling access in a group 14  
automatic  
  rebuild of coupling facility structures  
    description 102  
availability  
  provided by data sharing 2

## B

BACKOUT DURATION option of panel DSNTIPL 130  
backout processing, postponing 130  
backup database, recommendations 96  
BACKUP SYSTEM utility 98  
batch processing, designing 140  
buffer pool  
  allocating for parallel processing 155  
  relationship to group buffer pool 8  
  setting thresholds for Sysplex query parallelism 146  
  tuning for Sysplex query parallelism 152

## C

castout 13  
  causes 196  
  class queue 201  
  description 196  
  displaying castout owner 197  
CD-ROM, books on 233

central processor complex 8  
change log inventory (DSNJU003) utility 49  
change log inventory utility  
  specifying resynchronization port number 50  
checkpoint, group buffer pool 198  
cold start 133  
commands 16  
  authorizing 78  
  MODIFY DDF 45, 51  
  prefix  
    displaying 83  
  scope 77  
communications database (CDB)  
  identifying remote data sharing groups, SNA 62  
  identifying remote data sharing groups, TCP/IP 47  
  mapping DB2 location names 33  
  updating 49, 64  
conditional restart 132  
configuration  
  possibilities with data sharing 4  
connection request  
  DB2 location name 33  
connections  
  displaying, group buffer pool 206  
  exit routine, considerations 14  
  failed-persistent 79  
  maximum number of distributed 33  
  monitoring connections to remote systems 91  
contention, false  
  avoiding 162  
  detecting 163  
  reducing 164  
coupling facility  
  DiscFailConn event 128  
  duplexed 109, 112  
  group buffer pool 108, 109  
  lock structure 110  
  non-duplexed 108, 111  
  planning for shutdown 135  
  recovery scenarios 108  
  reducing page size overhead 180  
  SCA structure 111  
  structures  
    connection disposition 79  
    dealing with hung 128  
    displaying size and usage 83  
    monitoring 137  
    structure disposition 79  
    subset of members 111, 112  
Coupling Facility Activity Report of RMF 171, 205  
CREATE TABLESPACE statement  
  MEMBER CLUSTER option 189  
  TRACKMOD option 190  
cross-invalidation 10  
cross-system extended services (XES) 128  
CURRENT MEMBER, special register 90

## D

D XCF,STR command of z/OS  
  output 85

- D XCF,STR command of z/OS *(continued)*
  - output showing group buffer pool information 204
- data entries for group buffer pool
  - description 209
  - symptoms of too few 215
- data sharing
  - advantages 1
  - deleting members 30
  - flexible configurations 4
  - member on which SQL statements run 90
  - quiescing members 29
  - removing members 29
    - data sets to keep 29
- data sharing group
  - access methods 33
  - accessing remote groups in SNA networks 70
  - accessing remote groups in TCP/IP networks 54
  - configuring as SNA server 65
    - for group-generic access 65
    - specifying generic LU name 65
  - configuring as TCP/IP server 49
    - designating subsets of members 51
    - specifying DRDA port number 49
    - specifying generic LU name 52
    - specifying resynchronization port number 50
  - definition 1
  - description 1
  - identifying generic LU name 68
  - maintaining 17
  - member 1
  - shutting down network connection to requester 72
  - subsetting aliases 52
- data sharing groups
  - identifying remote groups 47, 62
- data sharing members
  - deactivating 30
  - destroying 31
- database
  - backup, recommendations 96
  - recovery 94
- DB2 books online 233
- DB2 commands
  - routing 77
  - scope 77
- DB2 Connect requester
  - configuring to use group access, TCP/IP 54
  - requirement to enable Sysplex support 54
- DB2 GENERIC LUNAME parameter 63, 65, 67
- DB2 Information Center for z/OS solutions 233
- DB2 location name
  - mapped to network element 33
- DB2 location names
  - mapping to network element 33
- DB2 Performance Expert
  - accounting report 175
  - data sharing reports 138
  - statistics report 217
- DB2 release level
  - displaying 83
- DB2 requester
  - configuring to use group access, TCP/IP 55
  - configuring to use group-generic access, SNA 71
  - configuring to use member-routing access, SNA 70
  - configuring to use member-specific access, TCP/IP 57
- DDF
  - location alias
    - dynamic 45
- DDF *(continued)*
  - updates the LOCATION 74
- deactivating
  - data sharing members 30
    - restoring 32
- deadlock 167
- deferred restart 133
- deferring restart processing 133
- DELETE NAME 206
- deleting
  - data sharing members 30
- destroying
  - data sharing members 31
- DFSMSHsm (Data Facility Hierarchical Storage Manager)
  - archiving logs in data sharing 92
- directory entries of group buffer pool
  - description 209
  - symptoms of too few 213
- disability viii
- disaster recovery, in a group 99
- DiscFailConn (Disconnected/Failed Connection) event 128
- DISPLAY BUFFERPOOL command
  - LIST option 184
- DISPLAY DATABASE command 166
- DISPLAY GROUP command
  - displaying information about the group 83
- DISPLAY GROUPBUFFERPOOL command
  - group detail report example 210
  - MDETAIL option 191
  - monitor group buffer pools 206
  - summary report example 206
- DISPLAY THREAD command 149
- displaying
  - information about
    - coupling facility structures 84
    - data sharing group 83
    - LPL (logical page list) 87
    - retained locks 89
- distributed data
  - connection limit in a data sharing group 33
  - recommendations for SNA alternatives 59
- Distributed Data Facility
  - of DSNJU003 utility 74
- distributed data facility (DDF) 33, 90
- DNS network addressing 41
  - example configuration 42
- domain name server (DNS) 41
- dormant 29
- DRDA 33
- DRDA port
  - updating using DSNJU003 utility 74
- DRDA port number
  - recommendation 49
  - reserving 50
  - specifying 49
- DSN3@ATH connection exit routine 14
- DSN3@SGN sign-on exit routine 14
- DSN7501A 111
- DSN7501I 111
- DSNB228I 114
- DSNB325A 114
- DSNB331 111
- DSNB744 112
- DSNB7445 112
- DSNB745 112
- DSNDWQ04 mapping macro 188

- DSNHDECP load module
  - group attachment name 81
  - subgroup attachment name 81
- DSNJU003 (change log inventory) utility 74
  - ALIAS option 51
  - designating subsets of members 51
  - specifying DRDA port number 49
- DSNJU003 utility
  - specifying resynchronization port number 50
- DSNR020I 29
- DSNR030I 29
- DSNTIP5 installation panel
  - specifying DRDA port number 49
- DUPLEX
  - option of CFRM policy 134
- duplexing 10
- DVIPA
  - specify 36
- DVIPA network addressing 35
  - configuration requirements 40
  - example configuration 37
  - example of group access 55
  - example of member-specific access 57
  - preparing for failure recovery 41
- DXR143I 111
- dynamic location aliases
  - defining 44
  - managing 45
- dynamic method 221

## E

- exit routine, considerations 14
- EXPLAIN statement, executing 224
- explicit hierarchical locking 159

## F

- failed-persistent connection
  - description 79
  - protects retained locks 127
- failure scenario
  - connectivity failure
    - lock structure and SCA 104
  - duplexed group buffer pool 104
- false lock contention, preventing 162
- field procedure, considerations 14
- function level of IRLM 17

## G

- GBP-dependent 8
- GBPCACHE 15
- GBPCACHE ALL 193
- GBPCACHE ALL clause
  - caching pages during a read 192
  - planning consideration 192
  - when to use 192
- GBPCACHE attribute of group buffer pools
  - duplexed group buffer pools 223
  - procedure for altering 223
- GBPCACHE CHANGED 193
- GBPCACHE clause
  - effect on guidelines for group buffer pool castout thresholds 202
- GBPCACHE NONE 193

- GBPCACHE SYSTEM 193
- GBPCACHE SYSTEM clause
  - use for LOB table spaces 192
- general-use programming information, described 240
- GENERIC column
  - SYSIBM.LUNAMES table 67
- GENERIC column, SYSIBM.LUNAMES table 63
- generic LU name 59
  - and RACF PassTickets 70
  - configuring members to use 67
  - defining for data sharing group 65
  - identifying for data sharing group 68
  - needed for RACF PassTickets 52
  - removing affinity to a partner 91
  - updating using DSNJU003 utility 74
- GENERIC option
  - of DSNJU003 utility 74
- global lock contention
  - limits 163
- global transaction, locking 158
- governor (resource limit facility) 141
- GRECP (group buffer pool RECOVER-pending) status 87
- group access, TCP/IP
  - configuring DB2 Connect requester 54
  - configuring DB2 requester 55
  - description 35
  - example using dynamic VIPA network addressing 55
- group attachment name
  - displaying 83
  - submitting applications using 81
- group attachment of online utility jobs 82
- group buffer pool 141
  - assigning to a page set 180
  - castout threshold guidelines 202
  - changing size 221
  - changing using ALTER GROUPBUFFERPOOL 220, 221
  - checkpoint 198
  - connectivity failure scenarios 104
  - cross-invalidation 179
  - data entries 209
  - default castout threshold 202
  - dependency 181, 185
  - description 179
  - determining inner-system sharing 185
  - directory entries 209
  - duplexing
    - altering ratio 223
    - connections 86
    - GBPCACHE attribute 223
    - monitoring using DISPLAY GROUPBUFFERPOOL 206
    - monitoring using RMF report 205
    - monitoring using z/OS command D XCFSTR 204
    - read operations 191
    - recommended castout threshold 202
  - RECOVER-pending (GRECP) status
    - removing using START DATABASE command 108
  - relationship to virtual buffer pools 8
  - storage shortage recovery scenario 114
  - structure failure scenarios 104
  - thresholds 200
  - too few data entries 215
  - too few directory entries 213
  - too small 210
  - tuning size and ratio 209
  - write operations 193
- group detail report of DISPLAY GROUPBUFFERPOOL command 210

- group domain name 34
- group DVIPA 34
- group IP address
  - associated with data sharing group 74
- group name
  - displaying 83
- group release level, displaying 83
- group restart 16
  - description 119
- group-generic access, SNA
  - configuring data sharing groups for 65
  - configuring DB2 requester 71
  - description 61
  - example of 71
  - switching to member-specific access 72
  - two-phase commit processing 61
- group-scope in DB2 Performance Expert reports 138
- GUI symbols 241

## H

- hung coupling facility connections 128

## I

- ICF (Integrated Coupling Facility) 99
- IFCID (instrumentation facility component identifier)
  - identifiers by number
    - 0045 176
    - 0221 151
    - 0222 151
    - 0231 151
    - 0250 113
    - 0251 188
    - 0259 189
- in-memory statistics, maintaining 224
- Integrated Coupling Facility (ICF) 99
- IP address
  - accept incoming connection requests 74
- IRLM (internal resource lock manager)
  - applying maintenance 17
  - avoiding false contention 162
  - displaying subsystem name and procedure name 83
  - failed-persistent connections 127
  - function level 17
  - global transaction
    - locking 158
- ISTGENERIC coupling facility structure for VTAM 65

## L

- library 233
- light restart 122
- LIMIT BACKOUT parameter of installation panel
  - DSNTIPL1 130
- LOB table spaces
  - use GBPCACHE SYSTEM clause 192
- location alias
  - dynamic 44
  - limitation with RACF PassTickets 70
  - updating using DSNJU003 utility 74
- LOCK ENTRY SIZE parameter of IRLMPROC 164
- LOCKINFO field of DISPLAY DATABASE output, page set
  - P-locks 188
- locking
  - explicit hierarchical 159

- locking (*continued*)
  - optimization 158
  - selected partitions using LOCKPART YES 166
- locking protocol 2
  - effect on parent L-locks 164
- locks
  - decreasing lock table entry size 165
  - detecting global contention 171, 172
  - false contention 162
  - global deadlock detection 167
  - hierarchy
    - description 166
    - explicit hierarchical locking 159
  - partition locks 166
  - physical
    - description 181
    - instrumentation data 188
    - page 187
    - retained state 187
    - when obtained 181
  - retained
    - P-locks 187
    - releasing 89
  - structure
    - changing size 176
    - displaying size and usage 83
    - monitoring usage 170
    - symptoms of storage shortage 116
  - XCF message buffer size effect on resolving global contention 166
- log
  - active 92
  - archive 92
  - archiving using DFSMSHsm 92
- log record
  - applying 94
  - header 94
- log record sequence number (LRSN) 94
- logical page list (LPL) 87
  - description 87
  - failed group buffer pool write 195
  - recovering pages
    - methods 102
  - status, description 87
- LU name of requester
  - specifying 68
- LUWID option of DISPLAY THREAD command 91

## M

- maintenance, applying to a group 17
- mapping macro, DSNDWQ04 188
- MAX REMOTE ACTIVE option 73
- MAX REMOTE CONNECTED thread limit 50
- MAXDBAT parameter 73
- MAXROWS option of CREATE TABLESPACE statement 187
- MAXUSRS
  - decreasing parameter of IRLMPROC 165
  - effect on lock entry size 164
- member
  - configuring to use generic LU name 67
  - definition 1
  - preventing from processing DDF requests 73
  - subsets 51
  - unregistering from Workload Manager 73
- MEMBER CLUSTER 15

- MEMBER CLUSTER option of CREATE TABLESPACE 140, 189
- member consolidation
  - changes for 25
- member name
  - displaying 83
- member on which SQL statements run, determining 90
- member release level, displaying 83
- member-routing access, SNA
  - description 60
  - RACF PassTicket limitation 60
  - two-phase commit resynchronization 60
- member-scope, DB2 Performance Expert reports 138
- member-specific access, SNA
  - configuring DB2 requester 70
  - example of 70
  - switching from group-generic access 72
- member-specific access, TCP/IP
  - configuring DB2 requester 57
  - description 42
  - example using DVIPA network addressing 57
- members
  - consolidate 23
- message by identifier
  - DSN7501I 111
  - DSN7504I 111
  - DSN7505I 115
  - DSN7512A 115
  - DSNB228I 108
  - DSNB250E 108
  - DSNB301E 113
  - DSNB303E 108, 111
  - DSNB304I 108
  - DSNB305I 108
  - DSNB314I 108
  - DSNB319A 114
  - DSNB320I 108
  - DSNB321I 108
  - DSNB353I 108
  - DSNB354I 108
  - DSNB743 109, 112
  - DSNB744 109
  - DSNB745 109
  - DSNI006I 102
  - DSNI021I 102
  - DSNI022I 102
  - DXR136I 110
  - DXR142E 116
  - DXR143I 110
  - DXR170I 116
- MODIFY DDF command 45

## N

- naming conventions 13
- network protocol
  - choosing 64
- network protocol, choosing 33, 49
- node profile
  - requester 52

## O

- online 233
- online books 233

## P

- P-locks, page
  - reducing contention 187
  - space map contention 189
- page
  - locks, physical 187
  - validity test in data sharing 191
- page set
  - determining if group buffer pool dependent 185
  - P-lock, determining retained state 187
- page set of partition 141
- page size
  - reducing coupling facility overhead 180
- parallel queries
  - monitoring 149
  - tuning 149
- parallel sysplex 13, 16
- Parallel Sysplex
  - definition 1
- parallel tasks
  - displaying information about 149
  - using a performance monitor 149
- parallelism assistant
  - definition 142
- parallelism coordinator
  - definition 142
- parallelism, Sysplex query 142
- partitioned table space, locking 166
- pending work 131
- performance
  - designing table spaces 140
  - improving 138
  - migrating batch applications 140
  - recommendation 138
  - setting expectations 139
- performance trace
  - global lock contention 176
- phases of restart 123
- physical lock
  - description 181
  - instrumentation data 188
  - retained state 187
  - when obtained 181
- physical open 141
- PLAN\_TABLE table, GROUP\_MEMBER column 224
- PORT option
  - of DSNJU003 utility 74
- postponing backout 130
- prefetch processing
  - for GBP-dependent page sets and partitions 191
- priority
  - of parallel query work 155
- product-sensitive programming information, described 241
- programming interface information, described 240, 241
- PSPI symbols 241

## Q

- query applications, designing for data sharing 142
- QUIESCED 29
- quiescing data sharing members 29
- QXREPOP1 field of statistics and accounting traces 151
- QXREPOP2 field of statistics and accounting traces 151

## R

- RACF PassTickets
  - generic LU name 70
  - generic LU name requirement 52
  - location alias limitation 60, 70
- ratio of directory entries to data entries
  - changing 223
  - choosing a value 209
  - description 209
- reason codes
  - X'00C20204' 113
  - X'00C20205' 108
  - X'00F70609' 114, 115
- rebinding automatically
  - access path change in data sharing 224
- RECOVER TABLESPACE utility, options
  - TOCOPY 97
  - TOLOGPOINT 97
  - TORBA 97
- recovering databases 94
- recovery
  - after disaster 99
  - coupling facility 102
  - description 94
  - LPL (logical page list) 102
  - options 97
  - point-in-time 97
  - preparing for fast 96
  - system-level backup 97
- remote data sharing groups
  - identifying 47, 62
- removing
  - data sharing members 29
- reports
  - coupling facility
    - RMF reports 137
  - group detail report 210
  - member-scope reports of DB2 Performance Expert 138
  - reports
    - Coupling Facility reports of RMF 137
    - Response Time Distribution report of RMF 137
    - Shared Device report of RMF 137
    - summary report 206
    - Sysplex Summary report of RMF 137
- requester
  - definition 33
  - setting up DB2 for z/OS 47, 62
  - shutting down network connection to data sharing
    - group 72
- requester LU name
  - specifying 68
- RESET GENERICLU command 72
- resource limit facility (governor)
  - data sharing 141
  - Sysplex query parallelism 155
- resource limit specification table (RLST)
  - command scope 141
  - unique name 141
- resource measurement facility (RMF)
  - Coupling Facility Activity Report 205
- Resource Measurement Facility (RMF)
  - reports for data sharing 137
- restart
  - conditional 132
  - DB2 data sharing group 119
  - deferring processing 133
  - group 119

- restart (*continued*)
  - light 122
  - postponing backout processing 130
- RESTORE SYSTEM utility 98
- restoring
  - data sharing members 32
- resynchronization port
  - updating using DSNJU003 utility 74
- resynchronization port number
  - specifying 50
- retained utility ID lock 120
- reverting to simplex mode of structure 134
- RMF (resource measurement facility)
  - Coupling Facility Structure Activity report 171
  - monitor group buffer pools 205
  - reports for data sharing 137
- row locking, page P-lock contention 187

## S

- SCA (shared communications area)
  - displaying size and usage 83
  - increasing storage 115
- scope of commands 77
- SCOPE parameter of IRLMPROC, NODISCON option 17
- secure port
  - updating using DSNJU003 utility 74
- selective partition locking
  - description 166
  - monitoring 166
- SETXCF STOP, REBUILD command of z/OS 134
- shared data, restricting access to 180
- SHAREPORT option, TCP/IP PORT configuration profile
  - statement 50
- shortcut keys
  - keyboard viii
- sign-on exit routine, considerations 14
- simplex mode of structure, reverting 134
- single-member access, SNA
  - description 62
- single-member access, TCP/IP
  - description 46
- SNA
  - access methods 59
    - group-generic access, description 61
    - group-generic, example 71
    - member-routing, description 60
    - member-specific, example 70
    - single-member access, description 62
    - switching from group-generic to member-specific 72
  - connecting distributed partners 70
  - definition 59
- SNA server
  - configuring data sharing group as
    - for group-generic access 65
    - specifying generic LU name 65
- softcopy publications 233
- space map, reducing P-lock contention 189
- START DATABASE
  - restrictions 141
- startup, DB2 79
- static method 221
- statistics trace
  - global locking 172
- status
  - displaying member status 83
  - GRECP 87

- status (*continued*)
  - LPL 87
- STOP DATABASE
  - restrictions 141
- STOP DDF command 72
- structure
  - duplexing
    - starting 134
    - stopping 134
- subgroup attachment name
  - submitting applications using 81
- subgroup attachment of online utility jobs 82
- subsets of members 51
- subsystem
  - name
    - for data sharing group members 83
- summary report, DISPLAY GROUPBUFFERPOOL
  - command 206
- SYSIBM.IPLIST table
  - for member-routing access, TCP/IP 47
  - for single-member access, TCP/IP 47
  - for TCP/IP access 48
- SYSIBM.IPNames table
  - for group access, TCP/IP 47
  - for member-routing access, TCP/IP 47
  - for single-member access, TCP/IP 47
  - for TCP/IP access 48
- SYSIBM.LOCATIONS table
  - for group access, TCP/IP 47
  - for group-generic access, SNA 62
  - for member-routing access, TCP/IP 47
  - for member-specific access, SNA 62
  - for single-member access, SNA 62
  - for single-member access, TCP/IP 47
  - for SNA access 63
  - for TCP/IP access 48
- SYSIBM.LULIST table
  - for member-specific access, SNA 62
  - for single-member access, SNA 62
  - for SNA access 64
- SYSIBM.LUNAMES table
  - for group-generic access, SNA 62
  - for member-specific access, SNA 62
  - for single-member access, SNA 62
  - for SNA access 63
  - GENERIC column 63, 67
- SYSLGRNX table space 94
- Sysplex
  - definition 1
- Sysplex Distributor
  - role in DVIPA network addressing 35
- SYSPLEX parameter, DB2 Connect requester 54
- Sysplex query parallelism 2
  - accounting report 151
  - configuration requirements 142
  - data set placement 152
  - description 142
  - determining parallel degree 151
  - disabling 157
  - improving response time 152
  - monitoring processor use 151
  - planning for 142
  - setting limits 155
  - setting workload management goals 143
    - example 144
- system-level point-in-time recovery 98
- Systems Network Architecture (SNA) 14

## T

- table spaces
  - LOB
    - use GBPCACHE SYSTEM clause 192
- TCP/IP
  - access methods 34
    - group access, description 35
    - member-specific access, description 42
    - single-member access, description 46
  - connecting distributed partners 54
  - definition 34
  - DNS network addressing 41
    - example configuration 42
  - DVIPA network addressing 35
    - configuration requirements 40
    - example configuration 37
    - example of group access 55
    - preparing for failure recovery 41
- TCP/IP server
  - configuring data sharing group as 49
  - designating subsets of members 51
  - specifying DRDA port number 49
  - specifying generic LU name 52
  - specifying resynchronization port number 50
- threads
  - monitoring 91
- threshold, group buffer pool
  - castout 200
    - class castout default 201
  - defaults 202
  - guidelines 202
- timeout in a group 167
- trace
  - accounting 175
  - events in a group 137
  - statistics, global locking activity 172
- TRACKMOD 15
- TRACKMOD option of CREATE and ALTER
- TABLESPACE 140, 190
- transaction rates 8
- Transmission Control Protocol/Internet Protocol (TCP/IP) 14
- two-phase commit processing
  - using group-generic access, SNA 61
- two-phase commit resynchronization
  - using member-routing access, SNA 60

## U

- utilities
  - identifier 82
  - stand-alone 83
  - submitting to a group 82
  - work data sets 82
- utility ID lock, retained 120

## V

- validation routine, considerations 14
- VARY NET,INACT,ID command 72
- VPPSEQT
  - definition 155
- VPXPSEQT
  - buffer pool threshold 146
  - definition 155
- VTAM (Virtual Telecommunications Access Method)
  - breaking affinity between systems 72

VTAM (Virtual Telecommunications Access Method)  
(*continued*)  
  generic resources 61  
VTAM affinity 91

## W

workload  
  management facility of z/OS  
    period switches on parallelism assistants 144  
    setting goals for parallel queries 143  
workload balancing 2  
Workload Manager (WLM)  
  role in DNS network addressing 41  
  role in dynamic VIPA network addressing 35  
  role in member-specific access 42  
  unregistering members 73

## X

XCF (cross-system coupling facility) component of z/OS  
  message buffer size effect on resolving global  
  contention 166  
  signal contention 152  
XES (cross-system extended services) 128

## Z

z/OS commands  
  D XCF 84





Product Number: 5605-DB2  
5697-P31

Printed in USA

SC19-2973-03



Spine information:

DB2 10 for z/OS

Data Sharing: Planning and Administration

