DB2 Universal Database for OS/390

Data Sharing: Planning and Administration

Version 6

SC26-9007-01

Note!

Before using this information and the product it supports, be sure to read the general information under Appendix B, “Notices” on page 265.

Second Edition, Softcopy Only (April 2000)

This edition applies to Version 6 of DB2 Universal Database Server for OS/390, 5645-DB2, 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.

This softcopy version is based on the printed edition of the book and includes the changes indicated in the printed version by vertical bars. Additional changes made to this softcopy version of the manual since the hardcopy manual was published are indicated by the hash (#) symbol in the left-hand margin. Editorial changes that have no technical significance are not noted.

Contents

Chapter 1. Introduction to this book and the DB2® for OS/390® library

This chapter contains specific information about this book, a general overview of the library for the licensed program DB2 for OS/390, a summary of changes to DB2, and a summary of changes to this book.

Who should read this book

This book 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 book assume the user is already familiar with administering DB2 without data sharing. See *DB2 Administration Guide* for any concepts not explained in this book.

How this book is organized

This book contains several chapters on planning for and using DB2 data sharing:

> “Chapter 2. Introduction to DB2 data sharing” on page 11 gives an overview of the advantages of data sharing, how it works, and the requirements for hardware and software.
>
> “Chapter 3. Planning for DB2 data sharing” on page 33 describes hardware and software configurations, availability planning, and a recommended naming convention.
>
> “Chapter 4. Installing and enabling DB2 data sharing” on page 75 includes recommendations for subsystem parameters, procedures for creating a data sharing group, whether you are installing or migrating. Details on each step in the procedures are in Section 2 of *DB2 Installation Guide*. This chapter also includes information about disabling data sharing, falling back, or removing a member from the data sharing group.
>
> “Chapter 5. Communicating with a data sharing group” on page 113 describes how you can set up your data sharing group in a distributed network.
>
> “Chapter 6. Operating with data sharing” on page 133 describes how you can control DB2 with commands, monitor activity in the group, and back up and recover data.
>
> “Chapter 7. Performance monitoring and tuning” on page 181 describes how you can reduce locking contention and how to tune group buffer pools to improve the performance of applications that run in the data sharing group. It also includes the majority of information about Sysplex query parallelism.

This book also contains an appendix with reference information about naming.

How to use this book

This book is the main source of information about DB2 data sharing. It can be used to educate yourself about data sharing and to do many of the tasks associated with data sharing.

However, there are many tasks associated with data sharing, especially those of setting up the hardware and software environment for the parallel Sysplex®, that require the use of other product libraries, such as MVS.

For installing DB2, use this book with *DB2 Installation Guide* to do initial planning and to develop your installation strategy. Detailed installation procedures are in *DB2 Installation Guide*. Exceptions and deviations from those procedures are noted in this book.

# Product terminology and citations

In this book, DB2 Universal Database Server for OS/390 is referred to as "DB2 for OS/390." In cases where the context makes the meaning clear, DB2 for OS/390 is referred to as "DB2." When this book refers to other books in this library, a short title is used. (For example, "See *DB2 SQL Reference*" is a citation to *IBM DATABASE 2 Universal Database Server for OS/390 SQL Reference*.)

References in this book to "DB2 UDB" relate to the DB2 Universal Database™ product that is available on the AIX®, OS/2®, and Windows NT™ operating systems. When this book refers to books about the DB2 UDB product, the citation includes the complete title and order number.

The following terms are used as indicated:

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

C and C language Represent the C programming language.

CICS® Represents CICS/ESA® and CICS Transaction Server for OS/390 Release 1.

IMS™ Represents IMS/ESA®.

MVS Represents the MVS element of OS/390.

How to use the DB2 library

Titles of books in the library begin with DB2 Universal Database for OS/390 Version 6. 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. For a complete list of books in the library, and the sections in each book, see the bibliography at the back of this book.

Throughout the library, the DB2 for OS/390 licensed program and a particular DB2 for MVS/ESA subsystem are each referred to as “DB2.” In each case, the context makes the meaning clear.

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 associated with DB2 are grouped into the following major categories (but supplemental information relating to all of the below tasks for new releases of DB2 can be found in *DB2 Release Guide*):

Installation: If you are involved with DB2 only to install the system, *DB2 Installation Guide* might be all you need.

If you will be using data sharing then you also need this book for data sharing installation considerations.

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 Query Management Facility (QMF™) or some other program, and the library for that program might provide all the instruction or reference material they need. For a list of the titles in the QMF library, see the bibliography at the end of this book.

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 *DB2 Application Programming and SQL Guide*, *DB2 SQL Reference*, and *DB2 ODBC Guide and Reference* just as end users do.

Application programmers also need instructions on many other topics:

- How to transfer data between DB2 and a host program—written in COBOL, C, or FORTRAN, for example
- How to prepare to compile a program that embeds SQL statements
- How to process data from two systems simultaneously, say DB2 and IMS or DB2 and CICS®
- How to write distributed applications across platforms
- How to write applications that use DB2 ODBC to access DB2 servers
- How to write applications that use Open Database Connectivity (ODBC) to access DB2 servers
- How to write applications in the Java™ programming language to access DB2 servers

The material needed for writing a host program containing SQL is in *DB2 Application Programming and SQL Guide* and in *DB2 Application Programming Guide and Reference for Java™*. The material needed for writing applications that use DB2 ODBC or ODBC to access DB2 servers is in *DB2 ODBC Guide and Reference*. For handling errors, see *DB2 Messages and Codes* .

Information about writing applications across platforms can be found in *Distributed Relational Database Architecture™: Application Programming Guide*.

System and Database Administration: *Administration* covers almost everything else. *DB2 Administration Guide* divides those tasks among the following sections:

- Section 2 (Volume 1) of *DB2 Administration Guide* discusses the decisions that must be made when designing a database and tells how to bring the design into being by creating DB2 objects, loading data, and adjusting to changes.
- Section 3 (Volume 1) of *DB2 Administration Guide* 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.
- Section 4 (Volume 1) of *DB2 Administration Guide* 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.
- Section 5 (Volume 2) of *DB2 Administration Guide* 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.

In addition, the appendixes in *DB2 Administration Guide* contain valuable information on DB2 sample tables, National Language Support (NLS), writing exit routines, interpreting DB2 trace output, and character conversion for distributed data.

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, then 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, then you need this book.

Additional information about system and database administration can be found in *DB2 Messages and Codes*, which lists messages and codes issued by DB2, with explanations and suggested responses.

Diagnosis: Diagnosticians detect and describe errors in the DB2 program. They might also recommend or apply a remedy. The documentation for this task is in *DB2 Diagnosis Guide and Reference* and *DB2 Messages and Codes*.

How to obtain DB2 information

DB2 on the Web

Stay current with the latest information about DB2. View the DB2 home page on the World Wide Web. News items keep you informed about the latest enhancements to the product. Product announcements, press releases, fact sheets, and technical articles help you plan your database management strategy.

You can view and search DB2 publications on the Web, or you can download and print many of the most current DB2 books. Follow links to other Web sites with more information about DB2 family and OS/390 solutions. Access DB2 on the Web at the following address:

http://www.ibm.com/software/db2os390

DB2 publications

The DB2 publications for DB2 Universal Database Server for OS/390 are available in both hardcopy and softcopy format.

BookManager® format

Using online books on CD-ROM, you can read, search across books, print portions of the text, and make notes in these BookManager books. With the appropriate BookManager READ product or IBM Library Readers, you can view these books in the OS/390, VM, OS/2, DOS, AIX, and Windows™ environments. You can also view many of the DB2 BookManager books on the Web.

PDF format

Many of the DB2 books are available in Portable Document Format (PDF) for viewing or printing from CD-ROM or the Web. Download the PDF books to your intranet for distribution throughout your enterprise.

CD-ROMs

Books for Version 6 of DB2 Universal Database Server for OS/390 are available on CD-ROMs:

- *DB2 UDB for OS/390 Version 6 Licensed Online Book*, LK3T-3519, containing *DB2 UDB for OS/390 Version 6 Diagnosis Guide and Reference* in BookManager format, for ordering with the product.
- *DB2 UDB Server for OS/390 Version 6 Online and PDF Library*, SK3T-3518, a collection of books for the DB2 server in BookManager and PDF formats.

Periodically, the books will be refreshed on subsequent editions of these CD-ROMs.

The books for Version 6 of DB2 UDB Server for OS/390 are also available on the following collection kits that contain online books for many IBM products:

- *Online Library Omnibus Edition OS/390 Collection*, SK2T-6700, in English
- *IBM Online Library MVS Collection Kit*, SK88-8002, in Japanese, for viewing on DOS and Windows operating systems.

DB2 education

IBM Education and Training offers a wide variety of classroom courses to help you quickly and efficiently gain DB2 expertise. Classes are scheduled in cities all over the world. You can find class information, by country, at the IBM Learning Services Web site:

http://www.ibm.com/services/learning/

For more information, including the current local schedule, please contact your IBM representative.

Classes can also be taught at your location, at a time that suits your needs. Courses can even be customized to meet your exact requirements. The *All-in-One*

Education and Training Catalog describes the DB2 curriculum in the United States. You can inquire about or enroll in these courses by calling 1-800-IBM-TEACH (1-800-426-8322).

How to order the DB2 library

You can order DB2 publications and CD-ROMs through your IBM representative or the IBM branch office serving your locality. If you are located 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 1-800-565-1234.

To order additional copies of licensed publications, specify the SOFTWARE option. To order additional publications or CD-ROMs, specify the PUBLICATIONS and SLSS option. Be prepared to give your customer number, the product number, and the feature code(s) or order numbers you want.

Summary of changes to DB2 UDB for OS/390 Version 6

DB2 UDB for OS/390 Version 6 delivers an enhanced relational database server solution for OS/390. This release focuses on greater capacity, performance improvements for utilities and queries, easier database management, more powerful network computing, and DB2 family compatibility with rich new object-oriented capability, triggers, and more built-in functions.

Capacity improvements

16-terabyte tables provide a significant increase to table capacity for partitioned and LOB table spaces and indexes, and for nonpartitioning indexes.

Buffer pools in data spaces provide virtual storage constraint relief for the ssnmDBM1 address space, and data spaces increase the maximum amount of virtual buffer pool space allowed.

Performance and availability

Improved partition rebalancing lets you redistribute partitioned data with minimal impact to data availability. One REORG of a range of partitions both reorganizes and rebalances the partitions.

You can **change checkpoint frequency dynamically** using the new SET LOG command and initiate checkpoints any time while your subsystem remains available.

Utilities that are faster, more parallel, easier to use:

- **Faster backup and recovery** enables COPY and RECOVER to process a list of objects in parallel, and recover indexes and table spaces at the same time from image copies and the log.
- **Parallel index build** reduces the elapsed time of LOAD and REORG jobs of table spaces, or partitions of table spaces, that have more than one index; the elapsed time of REBUILD INDEX jobs is also reduced.
- Tests show **decreased elapsed and processor time for online REORG**.

- **Inline statistics** embeds statistics collection into utility jobs, making table spaces available sooner.
- You can **determine when to run REORG** by specifying threshold limits for relevant statistics from the DB2 catalog.

Query performance enhancements include:

- **Query parallelism extensions** for complex queries, such as outer joins and queries that use nonpartitioned tables
- **Improved workload balancing in a Parallel Sysplex**® that reduces elapsed time for a single query that is split across active DB2 members
- **Improved data transfer** that lets you request multiple DRDA query blocks when performing high-volume operations
- The ability to use an **index to access predicates with noncorrelated IN subqueries**
- **Faster query processing** of queries that include join operations

More performance and availability enhancements include:

- **Faster restart and recovery** with the ability to postpone backout work during restart, and a faster log apply process
- **Increased flexibility with 8-KB and 16-KB page sizes** for balancing different workload requirements more efficiently, and for controlling traffic to the coupling facility for some workloads
- **Direct-row access** using the new ROWID data type to re-access a row directly without using the index or scanning the table
- **Ability to retain prior access path** when you rebind a statement. You almost always get the same or a better access path. For the exceptional cases, Version 6 of DB2 for OS/390 lets you retain the access path from a prior BIND by using rows in an Explain table as input to optimization.
- An **increased log output buffer size** (from 1000 4-KB to 100000 4-KB buffers) that improves log read and write performance

Data sharing enhancements

More caching options use the coupling facility to improve performance in a data sharing environment for some applications by writing changed pages directly to DASD.

Control of space map copy maintenance with a new option avoids tracking of page changes, thereby optimizing performance of data sharing applications.

User productivity

Predictive governing capabilities enhance the resource limit facility to help evaluate resource consumption for queries that run against large volumes of data.

Statement cost estimation of processing resource that is needed for an SQL statement helps you to determine error and warning thresholds for governing, and to decide which statements need tuning.

A **default buffer pool** for user data and indexes isolates user data from the DB2 catalog and directory, and separating user data from system data helps you make better tuning decisions.

More information available for monitoring DB2 includes data set I/O activity in traces, both for batch reporting and online monitors.

Better integration of DB2 and Workload Manager delay reporting enables DB2 to notify Workload Manager about the current state of a work request.

More tables are allowed in SQL statements SELECT, UPDATE, INSERT, and DELETE, and in views. DB2 increases the limit from 15 to 225 tables. The number of tables and views in a subselect is not changed.

Improved DB2 UDB family compatibility includes SQL extensions, such as:

- A VALUES clause of INSERT that supports any expression
- A new VALUES INTO statement

Easier recovery management lets you achieve a single point of recovery and recover data at a remote site more easily.

Enhanced database commands extend support for pattern-matching characters (*) and let you filter display output.

You can easily **process dynamic SQL in batch mode** with the new object form of DSNTEP2 shipped with DB2 for OS/390.

Network computing

SQLJ, the newest Java implementation for the OS/390 environment, supports SQL embedded in the Java programming language. With SQLJ, your Java programs benefit from the superior performance, manageability, and authorization available to static SQL, and they are easy to write.

DRDA® support for three-part names offers more functionality to applications using three-part names for remote access and improves the performance of client/server applications.

Stored procedure enhancements include the ability to create and modify stored procedure definitions, make nested calls for stored procedures and user-defined functions, and imbed CALL statements in application programs or dynamically invoke CALL statements from IBM's ODBC and CLI drivers.

DB2 ODBC extensions include new and modified APIs and new data types to support the object-relational extensions.

ODBC access to DB2 for OS/390 catalog data improves the performance of your ODBC catalog queries by redirecting them to shadow copies of DB2 catalog tables.

Better performance for ODBC applications reduces the number of network messages that are exchanged when an application executes dynamic SQL.

Improvements for dynamically prepared SQL statements include a new special register that you use to implicitly qualify names of distinct types, user-defined functions, and stored procedures.

DDF connection pooling uses a new type of inactive thread that improves performance for large volumes of inbound DDF connections.

Object-relational extensions and active data

The object extensions of DB2 offer the benefits of object-oriented technology while increasing the strength of your relational database with an enriched set of data types and functions. Complementing these extensions is a powerful mechanism, triggers, that brings application logic into the database that governs the following new structures:

- **Large objects (LOBs)** are well suited to represent large, complex structures in DB2 tables. Now you can make effective use of multimedia by storing objects such as complex documents, videos, images, and voice. Some key elements of LOB support include:
  - LOB data types for storing byte strings up to 2 GB in size
  - LOB locators for easily manipulating LOB values in manageable pieces
  - Auxiliary tables (that reside in LOB table spaces) for storing LOB values
- **Distinct types** (which are sometimes called user-defined data types), like built-in data types, describe the data that is stored in columns of tables where the instances (or objects) of these data types are stored. They ensure that only those functions and operators that are explicitly defined on a distinct type can be applied to its instances.
- **User-defined functions**, like built-in functions or operators, support manipulation of distinct type instances (and built-in data types) in SQL queries.
- **New and extended built-in functions** improve the power of the SQL language with about 100 new built-in functions, extensions to existing functions, and sample user-defined functions.

Triggers automatically execute a set of SQL statements whenever a specified event occurs. These statements validate and edit database changes, read and modify the database, and invoke functions that perform operations inside and outside the database.

You can use the **DB2 Extenders** feature of DB2 for OS/390 to store and manipulate image, audio, video, and text objects. The extenders automatically capture and maintain object information and provide a rich body of APIs.

More function

Some function and capability is available to both Version 6 and Version 5 users. Learn how to obtain these functions now, prior to migrating to Version 6, by visiting the following Web site:

http://www.software.ibm.com/data/db2/os390/v5apar.html

Features of DB2 for OS/390

DB2 for OS/390 Version 6 offers a number of tools, which are optional features of the server, that are shipped to you automatically when you order DB2 Universal Database for OS/390:

- DB2 Management Tools Package, which includes the following elements:

  - DB2 UDB Control Center
  - DB2 Stored Procedures Builder
  - DB2 Installer
  - DB2 Visual Explain
  - DB2 Estimator

- Net.Data® for OS/390

You can install and use these features in a “Try and Buy” program for up to 90 days without paying license charges:

- Query Management Facility
- DB2 DataPropagator™
- DB2 Performance Monitor
- DB2 Buffer Pool Tool
- DB2 Administration Tool

Migration considerations

Migration to Version 6 eliminates all type 1 indexes, shared read-only data, data set passwords, use of host variables without the colon, and RECOVER INDEX usage. You can migrate to Version 6 only from a Version 5 subsystem.

Summary of Changes to this Book

“Chapter 3. Planning for DB2 data sharing” on page 33 contains information about duplexing group buffer pools. Information about shared read-only data has been removed because that function is not supported in DB2 Version 6.

“Chapter 6. Operating with data sharing” on page 133 contains recovery scenarios for duplexed group buffer pools.

“Chapter 7. Performance monitoring and tuning” on page 181 contains the latest recommendations for performance.

Chapter 2. Introduction to DB2 data sharing

The data sharing function of the licensed program DB2 for OS/390 enables applications that run on more than one DB2 subsystem to read from and write to the same set of data concurrently.

DB2 subsystems that share data must belong to a DB2 *data sharing group*, which runs on a *Parallel Sysplex*. A data sharing group is a collection of one or more DB2 subsystems that access shared DB2 data. A Parallel Sysplex is a collection of MVS systems that communicate and cooperate with each other.

Each DB2 subsystem that belongs to a particular data sharing group is a *member* of that group. All members of a data sharing group use the same shared DB2 catalog and directory. Currently, the maximum number of members in a data sharing group is 32.

Some capabilities described in this book can be used whether or not you share data. We use the term *data sharing environment* to mean 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.

This chapter describes the following topics:

- “Advantages of DB2 data sharing”
- “How data sharing works” on page 19
- “Using DB2 data sharing” on page 25
- “Software and hardware requirements” on page 30

Advantages of DB2 data sharing

DB2 data sharing improves price for performance, improves the availability of DB2, extends the processing capacity of your system, and provides more flexible ways to configure your environment. There is no need to change SQL in your applications to use data sharing, although some tuning might be needed for optimal performance.

Improves price for performance

DB2 data sharing gives you a database solution that is powerful enough to handle complex business requirements, but which can be run on one or more smaller, less expensive System/390® (S/390®) microprocessors.

Improves availability

More DB2 users demand access to DB2 data every hour of the day, every day of the year. DB2 data sharing helps you meet this service objective by improving availability during both planned and unplanned outages.

As Figure 1 illustrates, if one subsystem comes down, users can access their DB2 data from another subsystem. Transaction managers are informed that DB2 is down and can switch new user work to another DB2 subsystem in the group.

For unplanned outages, restart and recovery can be automated by using MVS's automatic restart manager.

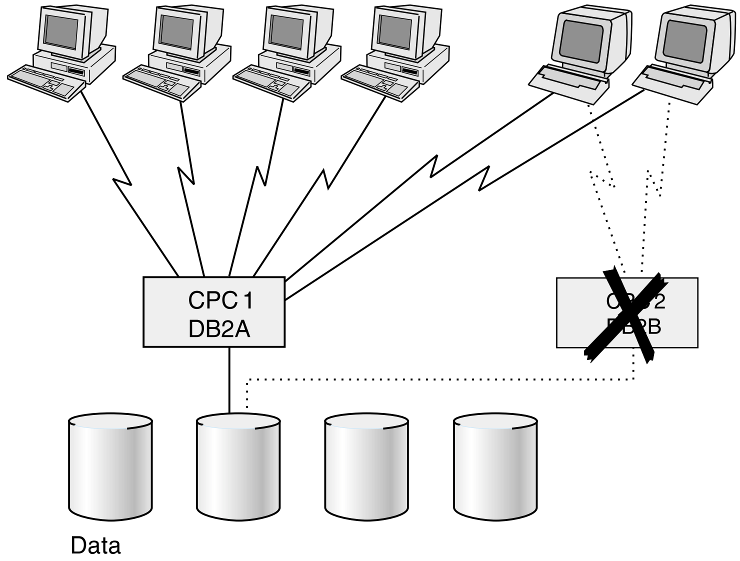


Figure 1. Data sharing improves availability during outages. If a DB2 or the entire central processor complex (CPC) comes down, work can be routed to another system.

DB2's increased availability has some performance cost, but the overhead for interprocessor communication and caching changed data is minimized by DB2's efficient locking and caching mechanisms, and its use of the coupling facility hardware. High availability for the DB2 structures in the coupling facility is provided by using automatic structure rebuild and duplexing of the structures used for caching data.

Expands capacity

As you move more data processing onto DB2, your processing needs can exceed the capacity of a single system. Data sharing relieves that constraint. This section describes the following topics:

- “Before data sharing”
- “After data sharing” on page 13
- “More capacity to process complex queries” on page 14

Before data sharing

Before DB2 data sharing, you had these options for relief:

- Move some applications, and copy the data onto another DB2.

  This approach requires that you maintain separate copies of the data. There is no communication among DB2s and no shared DB2 catalog and directory.

- Install another DB2 and rewrite applications to access the original data as distributed data.

This approach might relieve the workload on the original DB2, but it requires changes to your applications and has performance overhead of its own. Nevertheless, for DB2s that are separated by great distance or if DB2 needs to share data with a system outside the data sharing group, the distributed data facility is still your only option.

- Install a larger processor and move data and applications to that machine.

  This option can be expensive. Not only is it costly, but it means your system must come down while you move to the new machine.

After data sharing

With DB2 data sharing, you get the following benefits:

Incremental growth: The Parallel Sysplex can grow incrementally. You can add a new DB2 onto another central processor complex and access the same data through the new DB2. You no longer have to manage copies or distribute data. All DB2s in the data sharing group have concurrent read-write access, and all DB2s use a single DB2 catalog and directory.

Workload balancing: DB2 data sharing provides flexibility for growth and workload balancing. Unlike the partitioned data approach to parallelism (sometimes called the *shared-nothing* architecture), in which there is a one-to-one relationship between a particular database management system (DBMS) and a segment of data, data in a DB2 data sharing environment does not have to be redistributed when a new subsystem is added or if the workload becomes unbalanced. The new DB2 member has the same direct access to the data as all other existing members of the data sharing group.

Capacity when you need it: Consider using a data sharing configuration to handle your peak loads. You can start members to handle peak loads (such as end-of-quarter processing), and then stop them when the peak passes.

Higher transaction rates

Data sharing gives you opportunities to put more work through the system. As Figure 2 on page 14 illustrates, you can run the same application on more than one DB2 subsystem to achieve transaction rates that are higher than possible on a single subsystem.

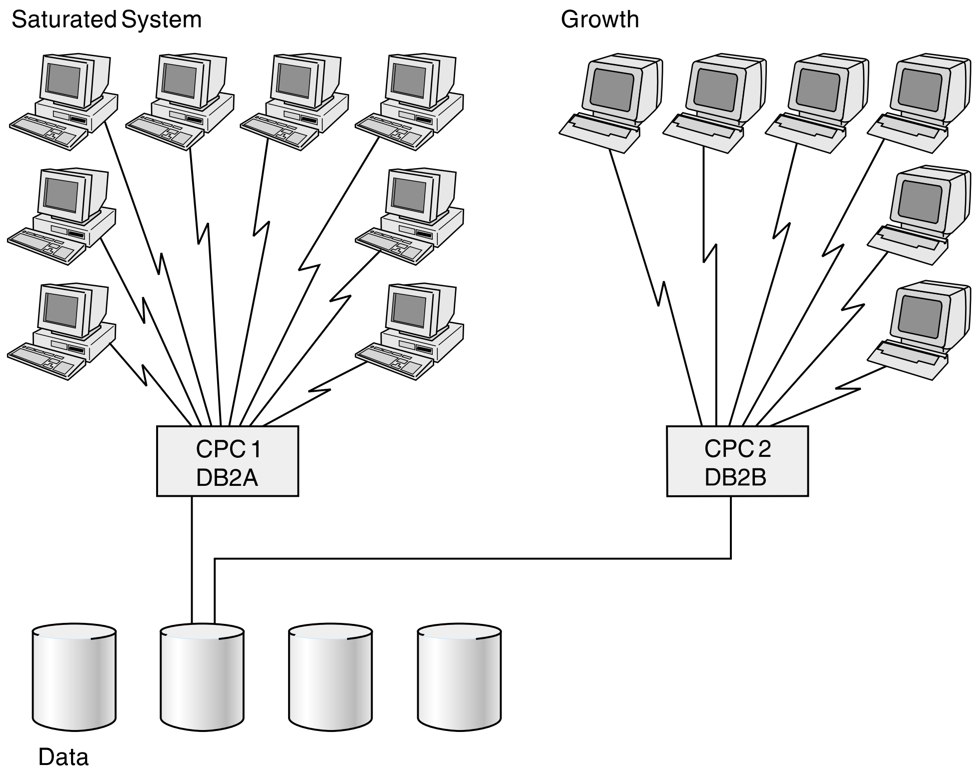


Figure 2. Data Sharing Enables Growth. You can move some of your existing DB2 workload onto another central processor complex (CPC).

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 those of you who do 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 those queries even as those queries become increasingly complex and run on larger and larger sets of data.

Figure 3 on page 15 shows that all members of a data sharing group can participate in processing a single query.

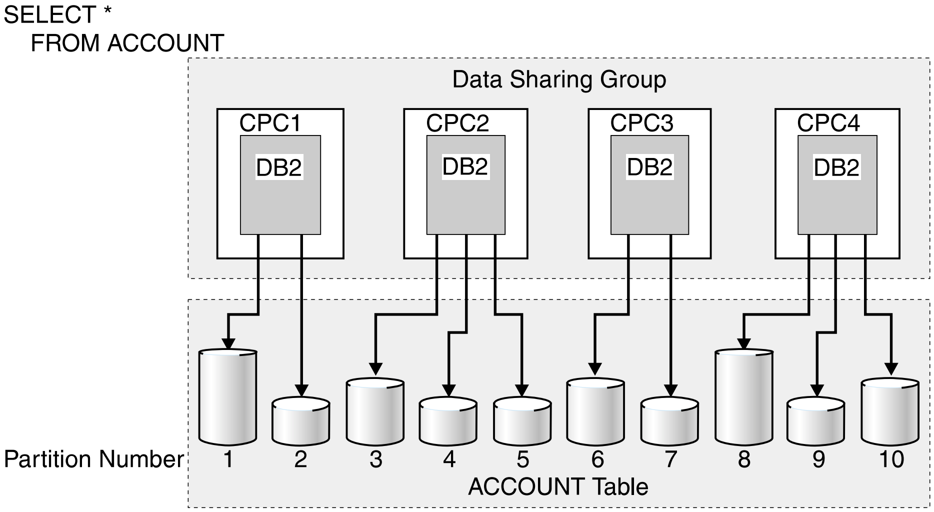


Figure 3. Query processed in parallel by members of a data sharing group. Different DB2 members process different partitions of the data.

This is a simplification of the concept—several DB2s can access the same physical partition. To take full advantage of parallelism, use partitioned table spaces.

Allows for flexible configurations

DB2 data sharing lets you configure your system environment much more flexibly.

As shown in Figure 4 on page 16, it is possible to have more than one DB2 data sharing group on the same MVS Sysplex. You might, for example, want one group for testing and another for production data. There is also a single, non-data-sharing DB2 in this example.

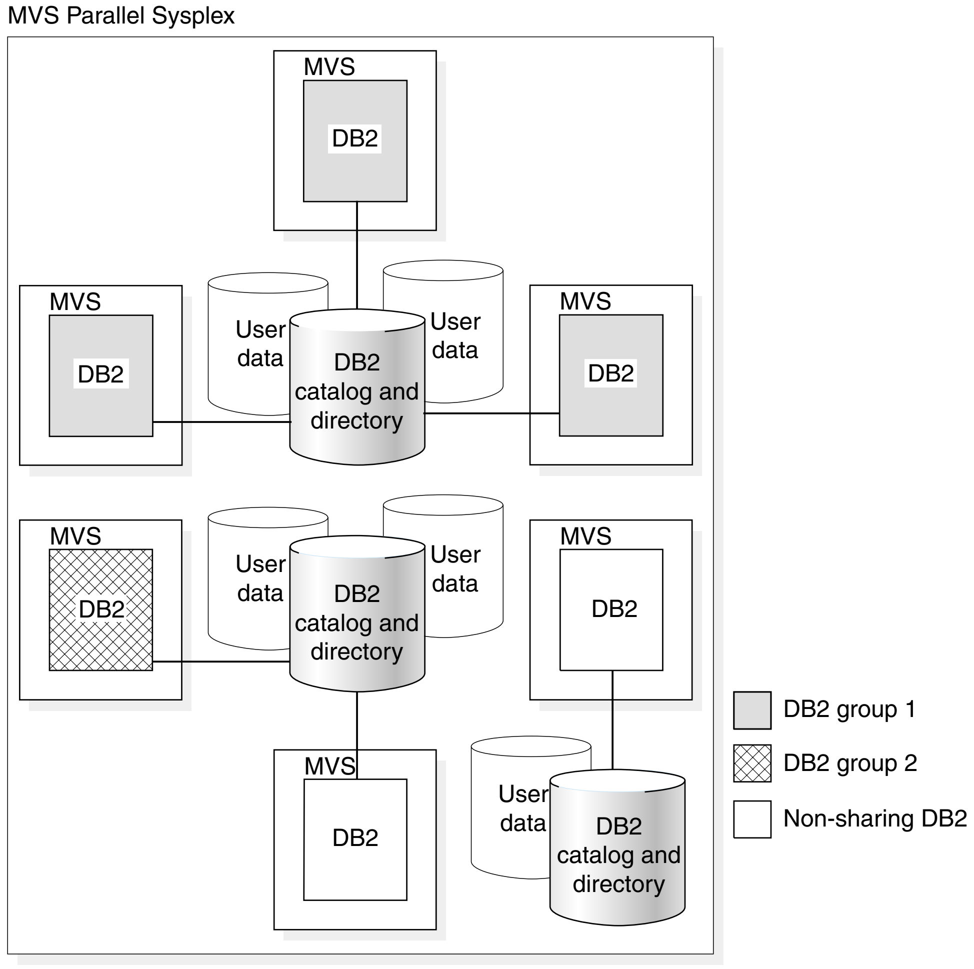


Figure 4. A possible configuration of DB2 data sharing groups. Although this example shows one DB2 per MVS, it is possible to have more.

Flexible operational systems

Figure 5 on page 17 shows how, with data sharing, you can have query user groups and online transaction user groups on separate MVS 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.

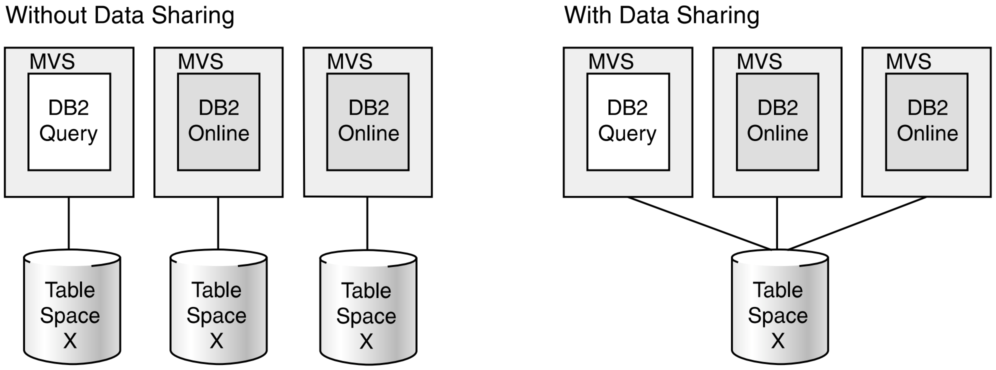


Figure 5. Flexible configurations with DB2 data sharing. Data sharing lets each set of users access the same data, which means that you no longer need to manage copies.

Flexible decision support systems

Figure 6 on page 18 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. 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 combines your operational and decision support systems into a single data sharing group and has these advantages:

- You can occasionally join decision support data and operational data with 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.

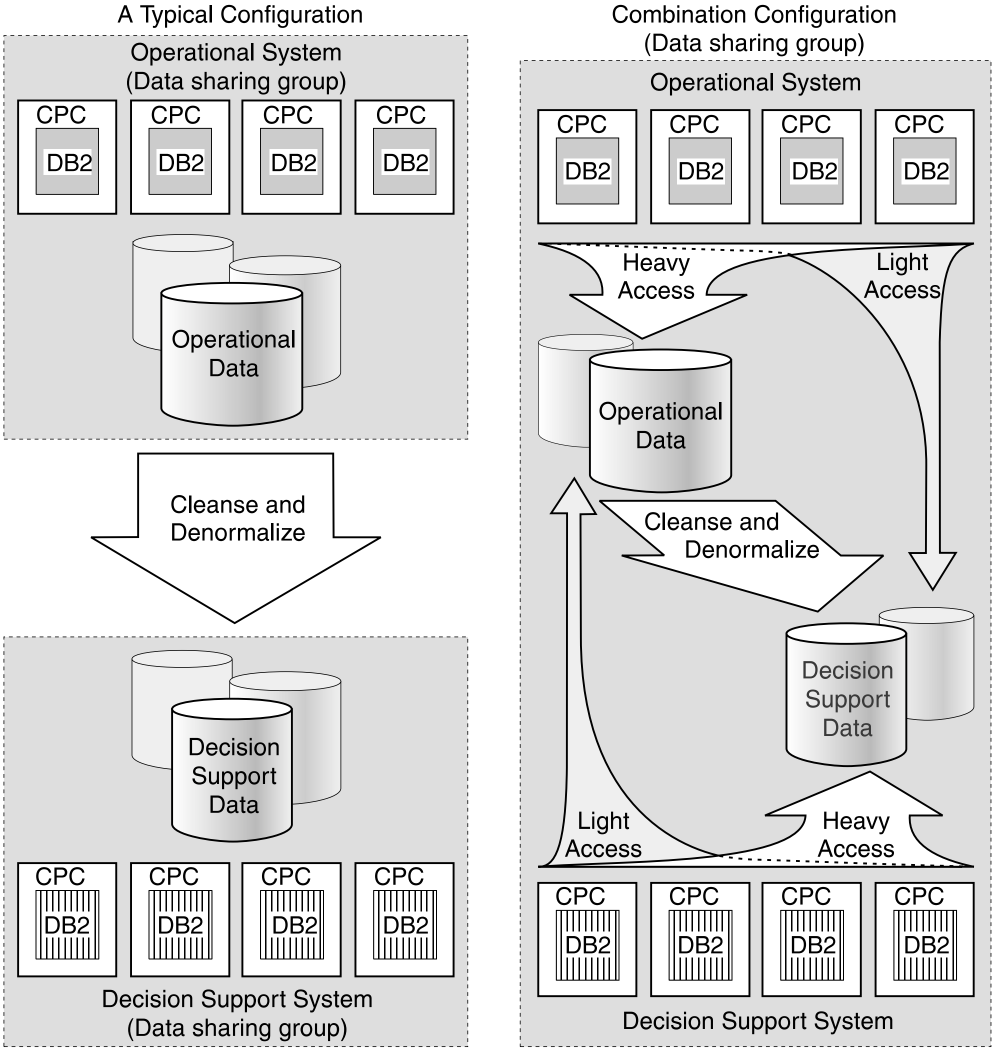


Figure 6. Flexible configurations for decision support. DB2 data sharing lets you configure your systems in the way that works best with your environment.

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

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

Application interface is unchanged

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

It is possible to bind a package or plan on one DB2 subsystem and run that package or plan on any other DB2 subsystem in a data sharing group.

How data sharing works

This section provides background information about how shared data is updated and how DB2 protects the consistency of that data. It also introduces operational and database design considerations for the shared environment.

You define a data sharing group and its members by using the installation and migration process. See “Creating the DB2 data sharing group” on page 82 for an overview of this process.

How DB2 protects data consistency

Applications can access data from any DB2 in the data sharing group. Many subsystems can potentially read and write the same data. DB2 uses special data sharing locking and caching mechanisms to ensure data consistency.

When multiple members of a data sharing group have opened the same table space, index space, or partition, and at least one of them has opened it for writing, then the data is said to be of inter-DB2 read/write interest to the members. (Sometimes we shorten this to *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. For more information about these policies, see *OS/390 MVS Setting Up a Sysplex*.

As shown in Figure 7 on page 20, a mapping exists between a group buffer pool and the buffer pools of the group members. For example, each DB2 has a buffer pool named BP0. For data sharing, you must define a group buffer pool (GBP0) in the coupling facility that maps to 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.

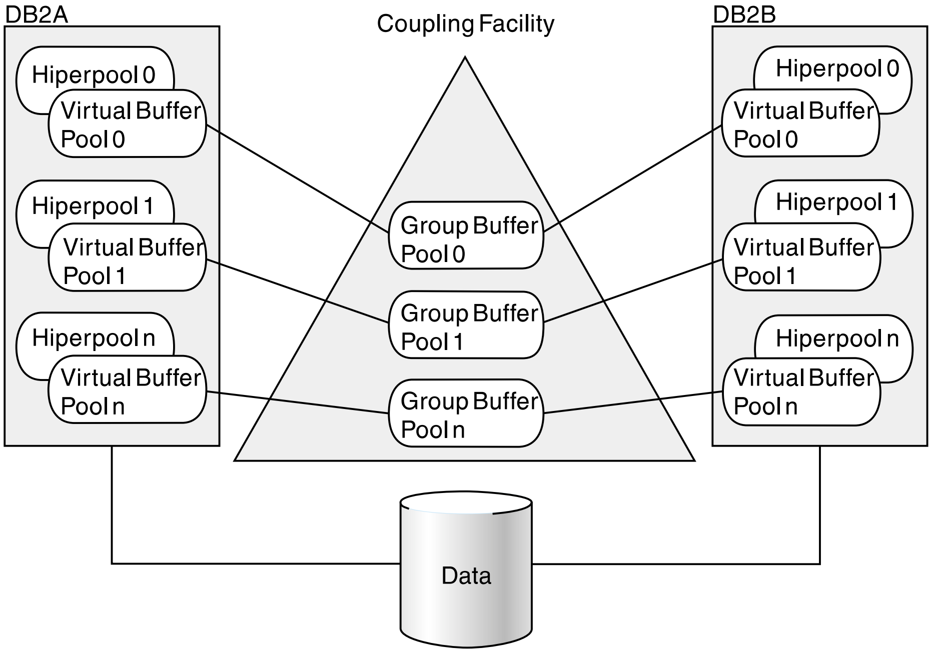


Figure 7. Relationship of virtual buffer pools and hiperpools to group buffer pools. There is one group buffer pool for all buffer pools of the same name.

When a particular page of data is changed, 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 DB2, that DB2 looks for the data in the group buffer pool.

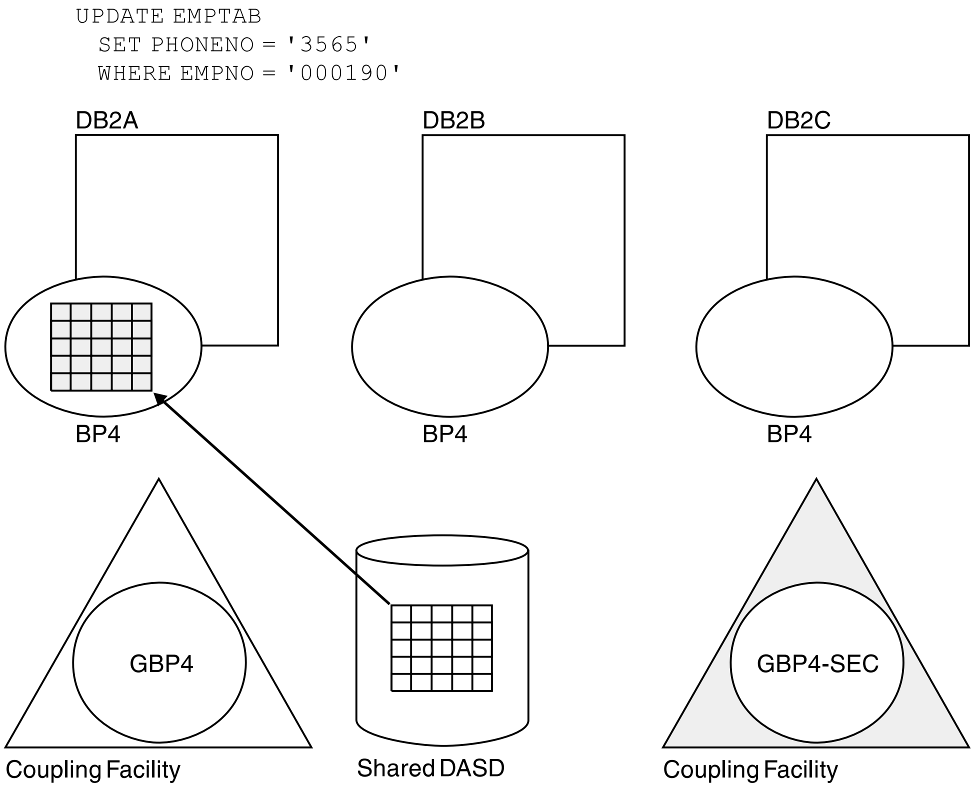
Performance options to fit your application's needs: Although the default is to cache just updated data, you also have options of caching all or none of your data. There is even an option especially for large object (LOB) data.

How an update happens

Let's follow a page of data as it goes through the update process. The most recent version of the data page is shaded in the illustrations. This scenario also assumes that the group buffer pool is used for caching just changed data and that it is *duplexed*, for high availability.

Suppose, as shown in Figure 8 on page 21, that an application issues an UPDATE statement from DB2A and that the data does not reside in the member's buffer pool, or in the group buffer pool. In this instance, DB2A must retrieve the data from DASD and get the appropriate locks to prevent another DB2 from updating the same record at the same time.

Because no other DB2 subsystem shares the table at this time, DB2 does not use data sharing processing for DB2A's update.



```
UPDATE EMPTAB
  SET PHONENO = '3565'
  WHERE EMPNO = '000190'
```

Figure 8. Data is read from DASD and updated by an application running on DB2A

Next, suppose another application, running on DB2B, needs to update that same data page (see Figure 9 on page 22). DB2 knows that inter-DB2 interest exists, so DB2A writes the changed data page to the primary group buffer pool. The write to the backup group buffer pool, the *secondary* group buffer pool, is overlapped with the write to the primary group buffer pool. DB2B then retrieves the data page from the primary group buffer pool.

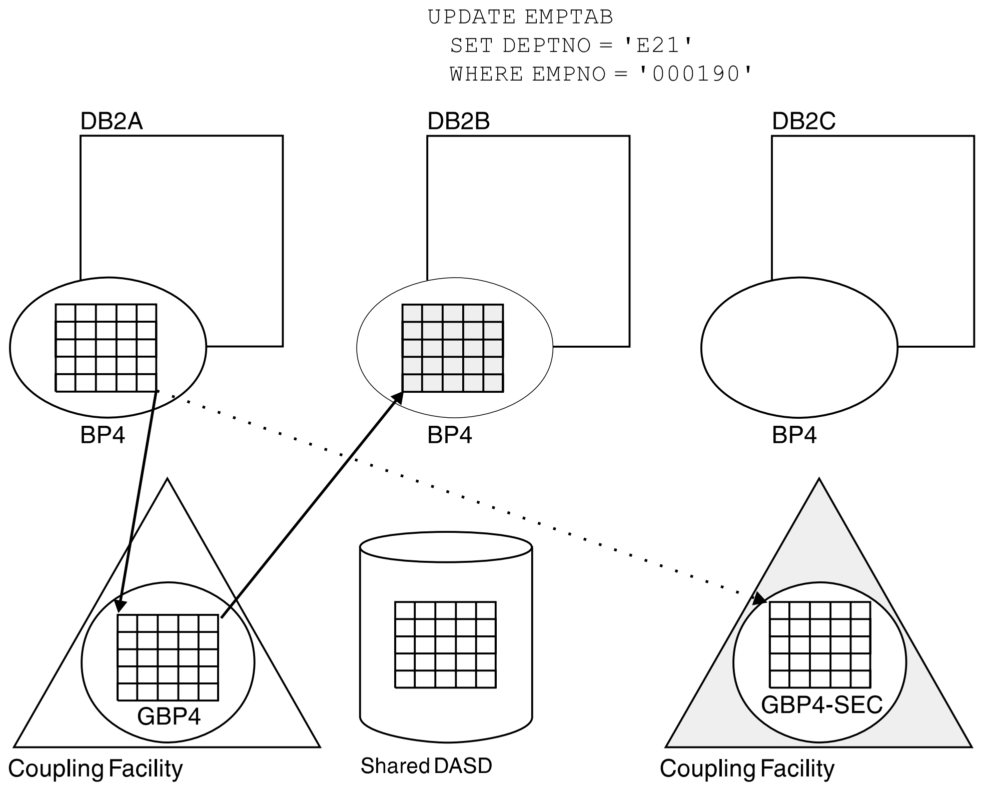


Figure 9. DB2B updates the same data page. When DB2B references the page, it gets the most current version of the data from the primary group buffer pool.

After DB2B updates the data, it moves a copy of the data page into the group buffer pool (both primary and secondary), and the data page is invalidated in DB2A's buffer pool (see Figure 10). Cross-invalidation occurs from the primary group buffer pool.

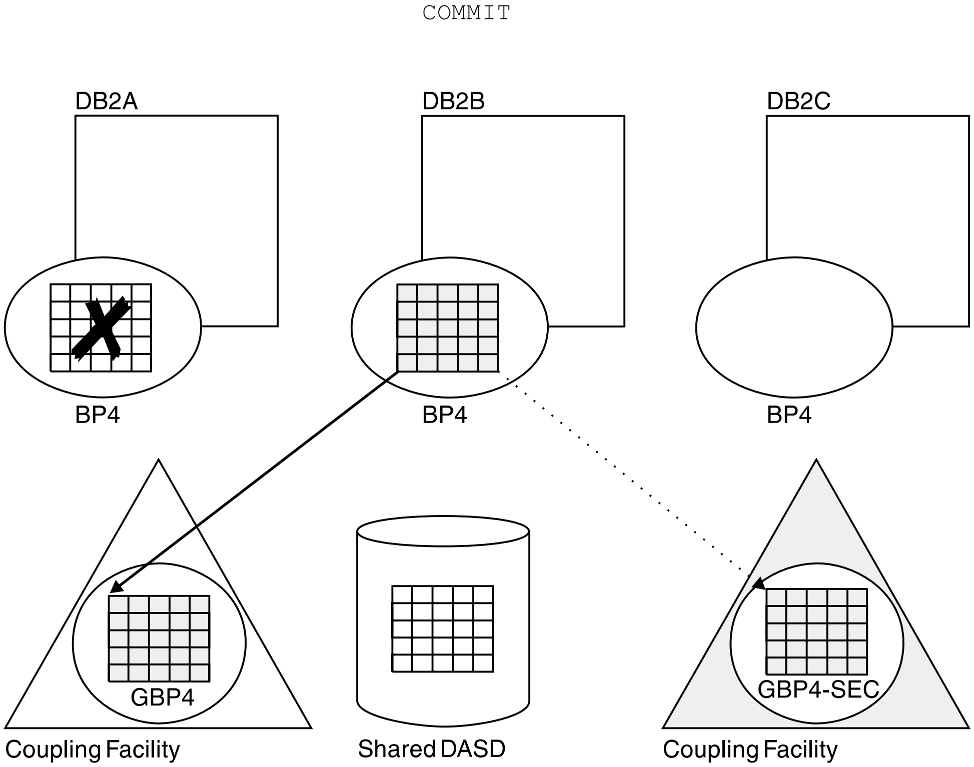
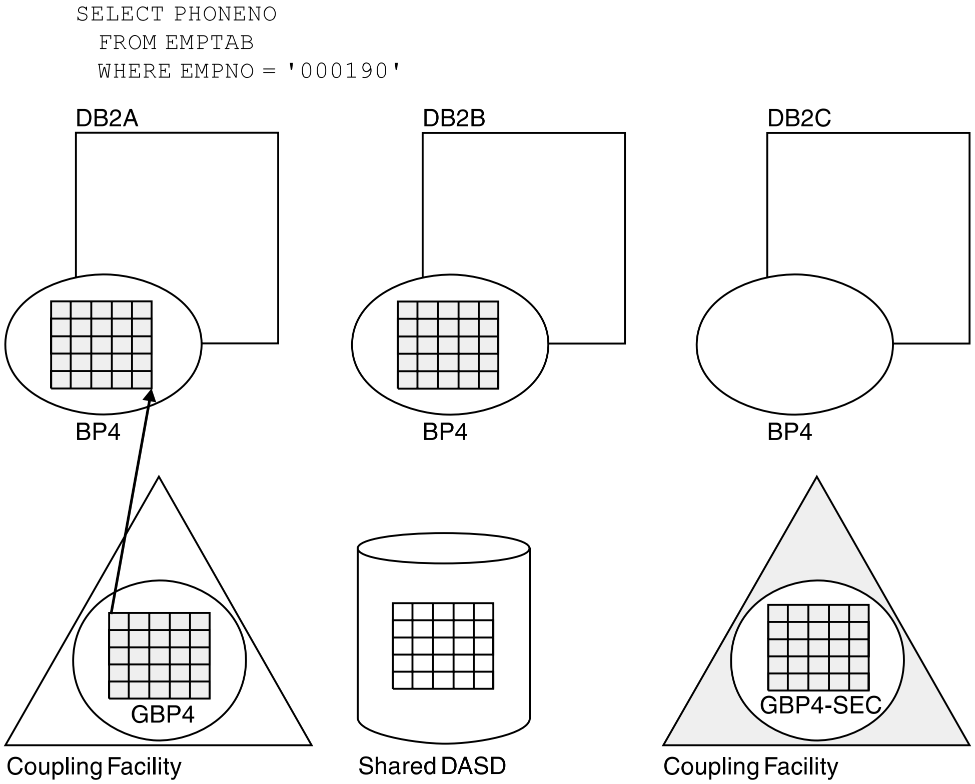


Figure 10. The updated page is written to the group buffer pool. The data page is invalidated in DB2A's buffer pool.

Now, as shown in Figure 11, when DB2A needs to read the data, the data page in its own buffer pool is invalid. Therefore, it reads the latest copy from the primary group buffer pool.

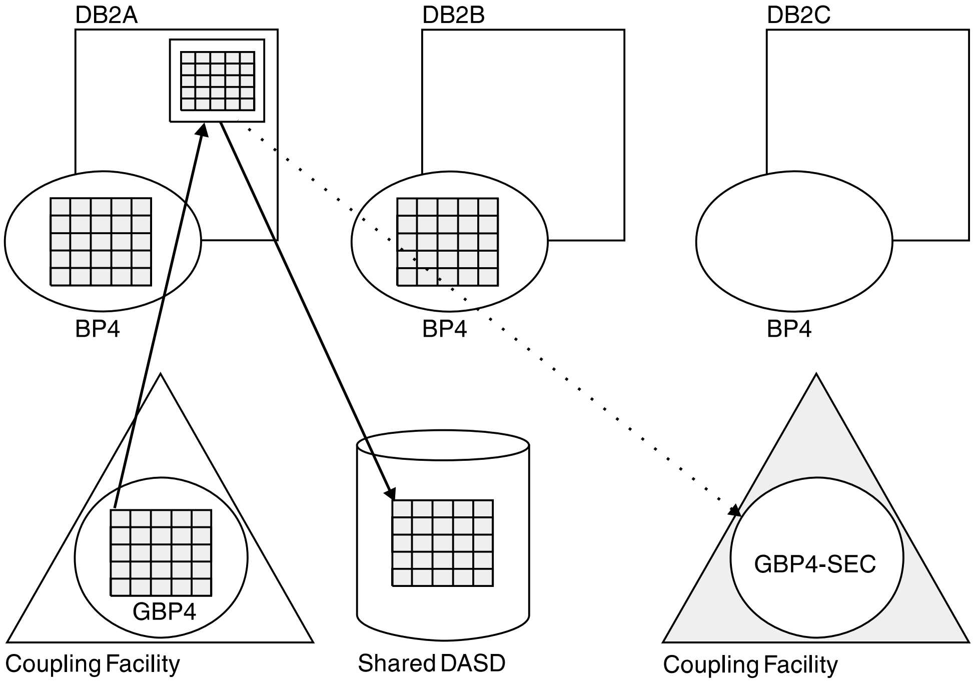


```
SELECT PHONENO
  FROM EMPTAB
  WHERE EMPNO = '000190'
```

Figure 11. Data is being read from the group buffer pool by DB2A

How DB2 writes changed data to DASD

Periodically, DB2 must write changed pages from the group buffer pool to DASD. This process is called *castout*. Suppose that DB2A is responsible for casting out the changed data. That data must first pass through DB2A's address space because there is no direct connection from a coupling facility to DASD (Figure 12). This data passes through a private buffer, not DB2's virtual buffer pools.



| *Figure 12. Writing Data to DASD. There is no direct connection from the coupling facility to DASD. The data must pass through DB2A's address space before being written to DASD.*

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

Using DB2 data sharing

Because data sharing does not change the application interface, there are no new tasks for application programmers and end users. However, system programmers, operators, and database administrators must perform additional tasks in a data sharing environment. Those tasks are briefly described in this section.

Enabling data sharing

You must plan a naming convention before enabling data sharing on the first DB2 in the group (the *originating* member). Because many names in the Sysplex and in the 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 unique names must also be given for every group resource (such as a name for the group). See “Planning a naming convention” on page 38 for more information about naming.

The originating member's DB2 catalog is the catalog for the entire group. Additional members of the group use the originating member's DB2 catalog. Add those as new member installations.

If you have data from existing DB2s to move into the group, you must merge their catalog definitions into the catalog used by the data sharing group and ensure that the data can be physically reached by each member of the data sharing group. DB2 does not provide a way to merge DB2 catalogs automatically.

Using the data sharing group for network computing

Applications can communicate with a data sharing group by using either systems network architecture (SNA) or, if you are running on OS/390 Release 3 or subsequent releases, Transmission Control Protocol/Internet Protocol (TCP/IP) network protocols. Applications attach to a data sharing group by using a single location name, and a single-system image is provided to all requesting applications. For more information about setting up a data sharing group for communicating in a network, see “Chapter 5. Communicating with a data sharing group” on page 113.

Administering a database

Because the DB2 catalog is shared, data definition, authorization, and control is the same as for non–data–sharing. Be sure every object has a unique name, considering that existing data might be merged into this group. Be sure that the data resides on shared DASD.

In this section, we briefly describe some database administrative tasks and their special considerations for data sharing.

Planning for performance

When you create objects, some options affect data sharing performance.

The GBPCACHE option: Use the GBPCACHE option when you create or alter table spaces or indexes to define what data, if any, should be cached in the group buffer pool. The options are NONE, SYSTEM, CHANGED, or ALL, with the default being CHANGED. See “Tuning group buffer pools” on page 220 for more information about choosing these options.

The TRACKMOD option: TRACKMOD NO can benefit applications when there are frequent updates from multiple members. This option can degrade incremental image copy performance; if you never use incremental copies, or if you use DFSMS™ concurrent copies and LOGONLY recovery, then choosing TRACKMOD NO can help transaction performance. See “Reducing space map page contention” on page 229 for more information about these options.

The MEMBER CLUSTER option: MEMBER CLUSTER can benefit applications when there are many inserts to the same table from multiple members. See “Reducing space map page contention” on page 229 for more information.

Planning for exit routines

If you use exit routines, such as a field or validation procedure, or the access control authorization routine, you must ensure that all members of the group are using the same routines. We recommend that all exit routines be placed in a program library shared by all members of the group.

Authorizing users

Use the same authorization mechanisms that are in place for non–data– sharing DB2s to control access to shared DB2 data and to the member DB2 subsystem. Because all DB2s in the group access the same catalog, any authorization ID has the same granted privileges and authorities in every member of the group.

As suggested for non–data–sharing DB2s, use a security system outside of DB2 (such as RACF® or its equivalent) to control which user IDs can access a particular DB2 subsystem. RACF does not recognize an entire 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 the data sharing group uses the same names for the connection and sign-on exit routines. We recommend that these routines be shared by all members of the group to avoid authorization anomalies such as primary authorization IDs that are treated differently by different members of the group or are associated with different sets of secondary IDs by different members.

Loading and reorganizing data

You can load or reorganize data from any DB2 in the data sharing group. For more information about LOAD and REORG, see Section 2 of *DB2 Utility Guide and Reference* .

Operating a data sharing group

This section describes some of the operational considerations for data sharing:

Entering commands

Sysplex technology lets you manage the DB2 data sharing group from consoles that are attached to a single MVS system or from separate systems. Figure 13 on page 28 shows how commands are passed from a single MVS system.

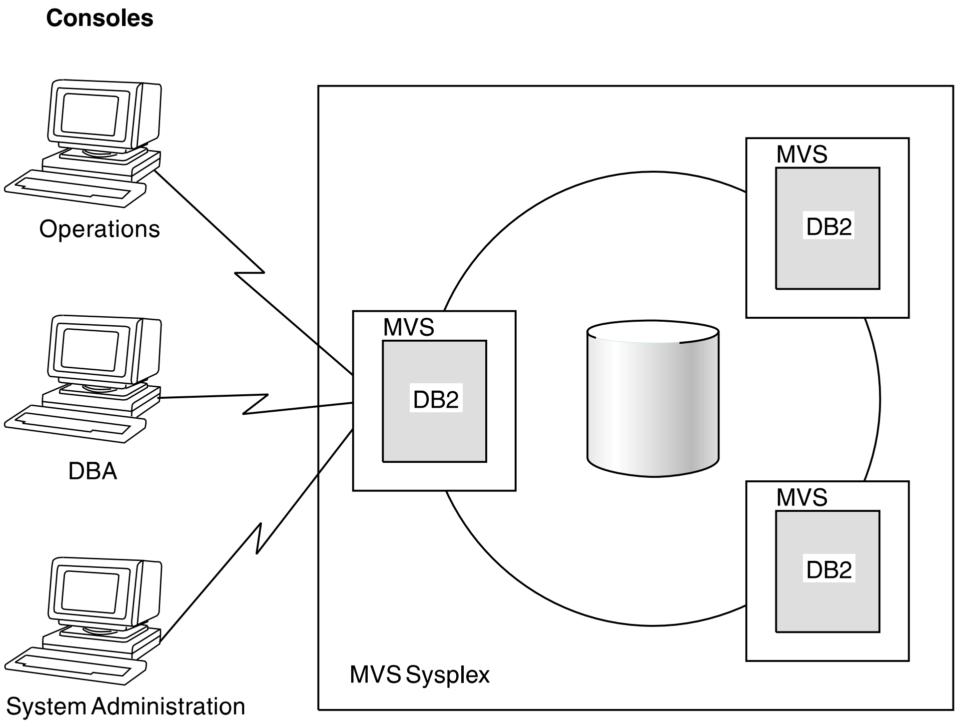


Figure 13. Issuing Commands

Using commands: Some commands manage group resources while others manage resources on a member-wide basis. See *DB2 Command Reference* for more information about specific commands.

Recovering data

DB2 recovers data from information contained in the logs and bootstrap data sets (BSDSs) of each member. However, because updates to any particular table can be logged on many different DB2 subsystems, DB2 coordinates recovery by using the *shared communications area* (SCA) in a coupling facility. The SCA contains the member names, BSDS data set names, and status information about objects and members in the group. It is also used to coordinate startup.

The RECOVER utility: You can run the RECOVER utility from any DB2 subsystem in the group. The process for data recovery is basically the same for DB2 data sharing as it is for non–data–sharing DB2s. However, updates to a single table space can come from many different DB2 subsystems. Therefore, to recover that object DB2 must merge log records from those DB2s, using a *log record sequence number* (LRSN). The LRSN is a value derived from the store clock timestamp and synchronized across the members by the Sysplex Timer.

| See “How recovery works in a data sharing group” on page 149 for more
| information about recovery.

| ***Coupling facility availability and recovery:*** In addition to data objects, coupling
| facilities contain vital resources needed for data sharing. We recommend that you
| have more than one coupling facility, to allow for group buffer pool duplexing, and
| for automatic recovery in the event a coupling facility fails. See “Coupling facility
| availability” on page 47 for more detailed suggestions.

Stopping and starting DB2

You can stop and start individual members in a data sharing group while the other members continue to run. The startup process for each DB2 member is similar to that of non–data–sharing DB2s.

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 this 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 start all members of the group. If a particular DB2 is not started, then one of the started DB2s performs group restart on behalf of that stopped DB2, but if that stopped DB2 is holding locks on resources, you must restart that DB2 to remove those locks.

Maintaining a data sharing group

To apply maintenance, most changes can be made on one DB2 at a time, as shown in Table 1. If you must take DB2, IRLM, or MVS offline for the change to take place, and the outage is unacceptable to users, you can move those users onto another DB2. Some sites have found it useful to define an extra member that they brin 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 group.

Table 1. Planned maintenance changes

Type of Change	Action Required
Early code	Bring down one MVS at a time and re-IPL.
DB2 code	Bring down and restart each DB2 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 DB2's update process. Stop and restart the DB2 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 use CASTOUT(NO) when you shut down multiple members of a data sharing group and you need to maintain consistent data on DASD. For example, you shut down all members to get a consistent copy of the databases on DASD that you can copy and 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 DASD copy might not have all the most recent data.

Hint: Consider specifying 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, specify NO for field DISCONNECT IRLM? on installation panel DSNTIPJ. There are operational issues to be considered; see Section 2 of *DB2 Installation Guide* for more information about the ramifications of choosing this option.

Applying maintenance to IRLM: Each DB2 has its own IRLM. The set of IRLMs have their own data sharing group. Similar to DB2, you have to stop and restart each IRLM in the group to roll the change throughout the group.

IRLM has a concept of 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 group. The group function level is the function level at which all IRLMs in the group must run, even if individual members are capable of running at a later function level.

It is a good idea to keep all members of the 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 MVS. See “Determining the function level of the IRLM group” on page 97 for more information.

Software and hardware requirements

This section describes at a high level the software and hardware that are required to run data sharing. For more detailed information about setting up the Parallel Sysplex, see *System/390 MVS Sysplex Hardware and Software Migration*.

Software

For the capability of dynamically changing structure sizes, as described in “Changing structure sizes” on page 63, structures must be allocated in a coupling facility at CFLEVEL=1 or higher.

Coupling facility performance enhancements as described in “Prefetch processing” on page 232 and “Locking optimizations” on page 203 require CFLEVEL=2 or higher.

Group buffer pool duplexing requires OS/390 Release 3 or subsequent releases and APAR OW28460. The APAR is not needed in Release 6 and subsequent releases. Group buffer pool duplexing also requires CFLEVEL=5 or higher. See “Duplexing group buffer pools” on page 51 for more information about group buffer pool duplexing.

Group buffer pool checkpoint performs better when a group buffer pool is allocated in a CFLEVEL=5 coupling facility and when MVS APAR OW28460 is applied to all members of the group. Group buffer pool checkpointing is described in “Group buffer pool checkpoint” on page 238.

Hardware

DB2 data sharing requires a S/390 Parallel Sysplex:

- Central processor complexes (CPCs) that can attach to the coupling facility
- At least one coupling facility and the appropriate channels and channel cards
- At least one Sysplex Timer®
- Connection to shared DASD

If you archive the DB2 log to tape, you might need a number of tape units greater than or equal to the number of DB2s in the group. These tape units must be accessible and sharable by the DB2 running a RECOVER utility.

Storage estimates

Installers must estimate the sizes of the various structures in the coupling facility. See “General information about coupling facility storage” on page 54 for more information.

Chapter 3. Planning for DB2 data sharing

To plan for the data sharing function of the licensed program DB2 for OS/390, you must coordinate your efforts with system hardware and software groups. You must complete these tasks before creating a DB2 data sharing group:

- “Planning for DB2 in the Parallel Sysplex”
- “Planning a naming convention” on page 38
- “Planning for availability” on page 46
- “Estimating storage” on page 54
- “Planning to enable data sharing” on page 66

The process of enabling data sharing is described in “Chapter 4. Installing and enabling DB2 data sharing” on page 75.

If you already have a Version 5 data sharing group, read this chapter for new information, and see “Migrating an existing data sharing group to the new release” on page 96.

Planning for DB2 in the Parallel Sysplex

This section has information about planning for DB2 in the MVS Parallel Sysplex and about the special connectivity needs for DB2 data sharing. For more information about specific hardware and software requirements for the Parallel Sysplex, see *System/390 MVS Sysplex Hardware and Software Migration*.

Considerations for the Parallel Sysplex

This section has information about the Parallel Sysplex and its relationship to DB2 data sharing. DB2 data sharing is dependent on the hardware and software components in the Parallel Sysplex.

Cross-system coupling facility (XCF) component of MVS

The cross-system coupling facility (XCF) component of MVS must be set up to enable the Sysplex. During startup, DB2 members join one XCF group, and the associated internal resource lock managers (IRLMs) join another XCF group. The MVS cross-system extended services will also join an XCF group implicitly on behalf of the IRLM connection to the lock structure. To join a particular group, the IRLMs and DB2s use the names you specify during DB2 installation.

DB2 uses the XCF for certain intersystem communications. It is recommended that you use both the coupling facility and channel-to-channel connections for XCF signalling. See *OS/390 MVS Setting Up a Sysplex* for more information about configuring the XCF.

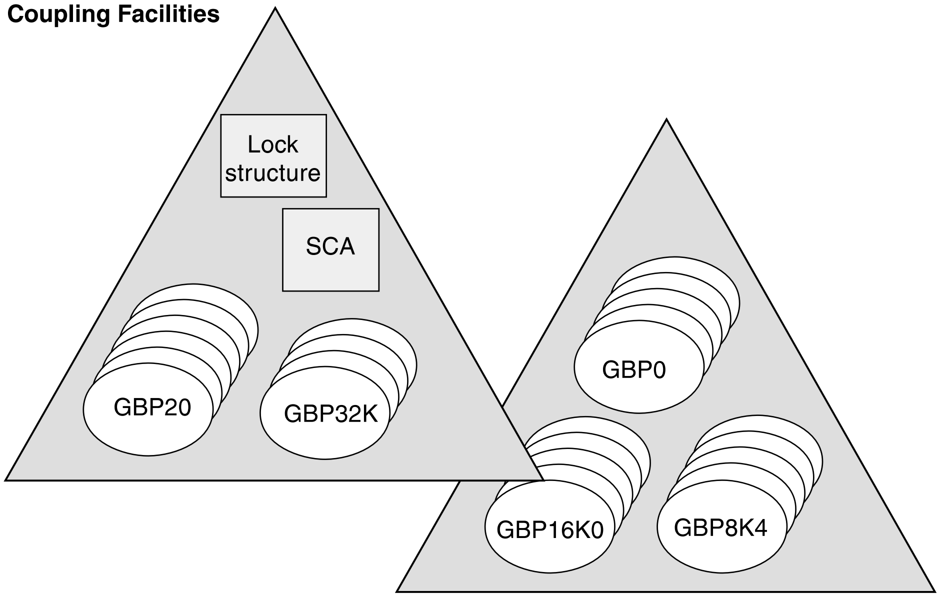
Sysplex timer

At least one Sysplex Timer must be installed in the Sysplex. The Sysplex Timer keeps the timestamps of the S/390 processors synchronized for all DB2s in the data sharing group. DB2 data sharing uses a value derived from the timestamp to recover data using the log.

Coupling facility

At least one coupling facility must be installed and defined to MVS before you can run DB2 with data sharing capability.

DB2 relies on areas of storage in the coupling facility called *structures*. There are three types of structure: lock, list, and cache. Each type of structure has a unique function that DB2 uses. Figure 14 shows the coupling facility structures used by DB2. Before starting DB2 for data sharing, you must have defined one lock structure, one list structure, and at least one cache structure (group buffer pool 0).



| *Figure 14. Coupling Facility Structures Used by DB2. This is a sample configuration. The lock structure and SCA do not have to be in the same coupling facility. Individual structures cannot span coupling facilities.*

- Lock structure

  There is one lock structure per data sharing group used by IRLM to control locking.

- List structure (DB2's SCA)

  There is one list structure per data sharing group used as the shared communications area (SCA) for the members of the group. The SCA contains information about databases in an exception condition and other information.

- Cache structures (DB2 group buffer pools)

  Cache structures are used as group buffer pools for the DB2 data sharing group. Group buffer pools are also used to maintain the consistency of data across the buffer pools of members of the group by using a cross-invalidating mechanism. Cross-invalidation is used when a particular member's buffer pool does not contain the latest version of the data.

  | The group buffer pools are also used to cache data of interest to more than | one DB2 in the data sharing group. The options for caching data are:

  | – Cache data that has been updated

  | – Cache all data (read-only and updated)

  | – Cache system control pages only (used only for LOB table spaces)

  - Cache no data, and just use the group buffer pool for cross-invalidation

One group buffer pool is used for all buffer pools of the same name in the DB2 group. For example, a buffer pool 0 (BP0) must exist on each member, to contain the catalog and directory table spaces. Thus, there must be a group buffer pool 0 (GBP0) in a coupling facility.

Similarly, if any member creates table space X and associates it with buffer pool 1, then X is associated with BP1 on every member, because there is only one definition of X in the catalog, which is used by the entire group. If the data in X is to be shared, you must define a group buffer pool 1 (GBP1) cache structure. If you don't define the group buffer pool, a single DB2 can update X, or more than one DB2 can read X. But there can be no inter-DB2 R/W activity for X.

Recommendation: For data that is private to each member, such as work files or user data that only one member reads, choose a buffer pool for those non-shared page sets. Assume you choose BP6. If you wanted just DB2A to have access to non-shared table space Y, define Y (and any indexes) to buffer pool 6. Every other member must define its virtual buffer pool 6 with a size of 0 and then there is no need to define the coupling facility structure for group buffer pool 6.

By moving this data to buffer pools that are separate from those used by shared data, it is easier to monitor the performance and provide more predictable performance for that private data.

Defining coupling facility structures: Before you start DB2 data sharing, you must define coupling facility structures. Use MVS's coupling facility resource management (CFRM) policies to define these structures to the MVS Sysplex. A CFRM policy determines how and where the structure resources are allocated.

We strongly recommend that you plan for and define the availability characteristics of the coupling facility structures for lost connectivity failures, which also includes total failure of the coupling facility. This requires a Sysplex failure management (SFM) policy, as described in “Rebuilding structures when connectivity is lost” on page 49.

See *OS/390 MVS Setting Up a Sysplex* for information about how to create CFRM and SFM policies.

A sample CFRM policy is shown in Figure 15 on page 36.

```
//POLICYX  JOB MSGCLASS=Z,REGION=2000K,CLASS=A,
//            MSGLEVEL=(1,1)
//STEP1  EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
 DATA TYPE(CFRM) REPORT(YES)
 DEFINE POLICY NAME(POLICYX) REPLACE(YES)
   STRUCTURE NAME(DSNDB0G_LOCK1)
             INITSIZE(32000)
             SIZE(64000)
             REBUILDPERCENT(5)
             PREFLIST(CF01,CF02)
   STRUCTURE NAME(DSNDB0G_GBP0)
             INITSIZE(50000)
             SIZE(100000)
             REBUILDPERCENT(5)
             DUPLEX(ALLOWED)
             PREFLIST(CF02,CF01)
   STRUCTURE NAME(DSNDB0G_SCA)
             INITSIZE(10000)
             SIZE(20000)
             REBUILDPERCENT(5)
             PREFLIST(CF01,CF02)

   CF NAME(CF01) TYPE(009674)
                 MFG(IBM)
                 PLANT(00)
                 SEQUENCE(000000040016)
                 PARTITION(1)
                 CPCID(00)
                 DUMPSPACE(1200)

   CF NAME(CF02) TYPE(009674)
                 MFG(IBM)
                 PLANT(00)
                 SEQUENCE(000000040029)
                 PARTITION(1)
                 CPCID(00)
                 DUMPSPACE(1200)
//
```

Figure 15. Sample CFRM policy

What you need to know: For DB2, know the following before creating the policy definitions:

- Initial size and maximum size (see “General information about coupling facility storage” on page 54)

  The structures can be dynamically resized from INITSIZE up to the value in SIZE. See “Don't overestimate the SIZE parameter” on page 55 and“Changing structure sizes” on page 63 for more information.

- Structure names (see “Coupling facility structure names” on page 42)
- Availability characteristics

You need to know the preference list (PREFLIST) for rebuilding or reallocating a structure should the coupling facility fail. See “Coupling facility availability” on page 47 for more information.

See “Rebuilding structures when connectivity is lost” on page 49 for information about specifying the value for REBUILDPERCENT.

See “Duplexing group buffer pools” on page 51 for information about specifying the value for DUPLEX.

Authorize DB2 to access the structures: Optionally, you can set up a facility class profile to limit access to the structures in the coupling facility. If you do this, ensure that DB2 does have access by ensuring that the IDs associated with the DB2 address spaces have update access authority to the coupling facility structures through RESOURCE(IXLSTR.*structure_name*) in SAF class CLASS(FACILITY).

If you do not create a facility class profile, the default allows any authorized user or program (supervisor state and program key mask allowing key 0-7) to issue coupling facility macros for the structure.

Common MVS libraries

As you will read in “Naming recommendations” on page 43, DB2 supports a configuration with a SYS1.PARMLIB and SYS1.PROCLIB shared by all MVS systems in the Parallel Sysplex. This can make adding and modifying systems much easier.

One detail to remember, especially if you intend to have many DB2 subsystems in the Sysplex, is that each DB2 and each IRLM you define to MVS in the IEFSSN*xx* parmlib member requires an MVS system linkage index (LX). The default number of these indexes that MVS reserves is 55. If you place all your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 55 LXs, or risk having subsystems not able to start.

If you need more than 55 LXs, use the NSYSLX option on the MVS IEASYS*xx* parmlib member to increase this number. See *OS/390 MVS Initialization and Tuning Guide* for more information.

Connectivity considerations

DB2 data sharing requires that all DB2-related resources reside on shared DASD. The DB2 catalog and directory, as well as any user data that is to be shared, must be on shared DASD. The integrated catalog for DB2 data sets must also be on shared DASD.

Also, all the members' logs and bootstrap data sets (BSDSs) must be on shared DASD for recovery purposes. A DB2 performing recovery must have access to the logs of other members in the group.

We strongly recommend that you place work files on shared DASD for the following reasons:

- It is required for processing many types of queries when those queries use Sysplex query parallelism. Each assisting DB2 writes to its own work file, and the coordinator can read the results from the assistants' work files.
- It lets you create or drop a work file table space from any other member in the group.

- It keeps DB2 connected to its work file even if you have to restart the DB2 on another processor.

Make sure you have physical connectivity by checking the following connections:

- Verify that there is one user integrated catalog facility for cataloging DB2 system data sets and that this catalog is accessible from each MVS in the Sysplex.
- Verify connectivity from each system on which a DB2 resides to:
  - A set of DB2 target libraries
  - A single DB2 catalog
  - A single DB2 directory
  - All databases that are to be shared
  - All log data sets
  - All BSDS data sets
  - User integrated catalog facility catalogs for shared databases
  - All coupling facilities used by the DB2 data sharing group

Planning a naming convention

Carefully consider the naming convention you will use to name the various parts of the data sharing system. Names must be assigned for both IRLM and DB2 groups, and for members within a group. One recommendation is to make names and prefixes unique within the MVS Sysplex. Although this uniqueness is not required for all names, it can help you avoid problems identifying entities and moving them among MVS systems in the Sysplex.

In this section, we describe the names for which you must choose values. Other names are generated during DB2 installation. A complete list of names is in Appendix A, “DB2 and IRLM names” on page 261. “Naming recommendations” on page 43 contains a suggested naming convention, which we use when we describe the names in this section. If you want to change the name of an existing DB2 subsystem to conform to your naming convention, see “Renaming the DB2 subsystem” on page 85.

Choosing names used in the DB2 group

Certain names are shared by all members of the data sharing group.

DB2 group name

The name that encompasses the entire group. The coupling facility structure names are based on the group name, as described in “Coupling facility” on page 34. The group name must be unique within the Sysplex. If you use this name as a basis for the location name, this name must be unique within the network.

The group name can be up to 8 characters long and can consist of the characters A-Z, 0-9, $, #, and @. The group name must begin with an alphabetic character.

To avoid names that IBM uses for its XCF groups, do not begin with the letters A-I unless the first three characters are DSN. Do not use the string SYS as the first three characters, and do not use the string UNDESIG as your group name.

catalog alias

This name can be up to 8 characters long. This is the name of the MVS catalog alias, which you must place in the MVS master catalog. We recommend that this name be the same as the group name. An example of a DB2 catalog alias name is DSNDB0G.

group attachment name

This name can be up to 4 characters long. This name is used by the TSO/batch attachment, the call attachment facility (CAF), utilities, and the Recoverable Resource Manager Services attachment facility (RRSAF) as a "generic" attachment name. For more information about using the group attachment name, see "Using the group attachment name" on page 136.

An example of a group attachment name is DB0G.

DRDA port

If the data sharing group is servicing DRDA requests that use TCP/IP, specify the same DRDA port number for the group. The recommended port number is 446. See "DRDA access through TCP/IP" on page 117 for more information about using TCP/IP.

location name

This name is used if the group is going to be processing distributed requests. The group is treated as a single location by remote requesters. This name can be up to 16 characters long.

generic LU name

This name allows remote requesters to configure their systems to treat the data sharing group as a single LU. This name can be up to 8 characters long. See "Chapter 5. Communicating with a data sharing group" on page 113 for more information about the generic LU name.

Sysplex domain name

This name lets you take advantage of workload balancing for TCP/IP connections. The name must be registered in the domain name server. The Sysplex domain name is of the format location.sysplexname. If using OS/390 Release 4 with APAR PQ06340, or with a subsequent release of OS/390, workload manager registers this name for you. See "Registering names in the domain name server" on page 129 for more information about the Sysplex domain name.

DRDA PORT

This name specifies the DRDA port number for the entire data sharing group. See "Chapter 5. Communicating with a data sharing group" on page 113 for more information about the generic LU name.

Choosing names used by members

These are names that must be unique within the data sharing group or, in certain cases, the MVS Sysplex.

member name

This name can be up to 8 characters long and can consist of the characters A-Z, 0-9, $, #, and @. This is the name of an individual member of a particular DB2 group. DB2 uses this name to form its MVS cross-system coupling facility (XCF) member name. An example of a member name is DB1G.

member subsystem name

This name can be up to 4 characters long and is the name used by all the attachment interfaces. It must be unique within the MVS Sysplex. We recommend that the member name and member subsystem name be the same. An example of a member subsystem name is DB1G.

LU name

This name must be unique within the data sharing group and the network. See Section 3 of *DB2 Installation Guide* for more information about choosing LU names.

member domain name

This name lets DB2 handle indoubt thread resolution for TCP/IP connections. The name must be registered in the domain name server and is of the format luname.location.sysplexname.domainname. If using OS/390 Release 4 with APAR PQ06340, or with subsequent releases of OS/390, workload manager registers this name for you. See “Registering names in the domain name server” on page 129 for more information about the member domain name.

resynchronization port

This port is used for TCP/IP connections. The resynchronization (resync) port allows each member to handle outstanding resync requests, even if the member is started on another CPC. See “DRDA access through TCP/IP” on page 117 for more information.

command prefix

This prefix can be up to 8 characters long and is used to direct commands entered from an MVS console to a particular DB2 subsystem. The default is the concatenation of the hyphen character (-) with the subsystem name. See Section 2 of *DB2 Installation Guide* for more information about valid characters for a command prefix.

As described in “Registering command prefixes and group attachment name” on page 68, this string is specified as a parameter on the IEFSSN*xx* subsystem definition. An example of a command prefix is -DB1G. You can have blanks between the command prefix and the command.

Do not assign a command prefix that is used by another subsystem or that can be interpreted as belonging to more than one subsystem or MVS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix starting from the first character. For example, it is invalid to assign a hyphen (-) to one subsystem and `'-DB2A'` to another. Similarly, it is also invalid to assign `'?DB2'` to one subsystem and `'?DB2A'` to another. It is valid, for example, to assign `'-DB2A'` and `'-DB2B'` to different DB2 subsystems.

work file database

This name can be up to 8 characters long. Each DB2 member has its own work file database (known as DSNDB07 in a non-data-sharing environment). One member of the data sharing group can have the name DSNDB07, but you might want to create a work file database with a more meaningful name, such as WRKDB1G, for member DB1G.

You cannot specify a name that begins with DSNDB unless the name is DSNDB07.

load module for subsystem parameters
This name can be up to 8 characters long. Each member has its own subsystem parameters. An example name is DSNZP01G, a naming convention which you can use to associate member DB1G with DSNZP01G.

Choosing names for data sets: When choosing names for member data sets, remember that data set names beginning with *membname* must have a master catalog alias to point to the catalog where the data sets are cataloged. The DB2 installation process does not create this catalog alias. One way to handle this is to begin member data set names with *catalias* and a member-related qualifier. For example, member data set names could have a form *catalias.membname.xxxxx*. This format eliminates the need to have a master catalog alias for *membname*.

member BSDS names These names can be up to 33 characters long. Sample BSDS names are DSNDB0G.DB1G.BSDS01 and DSNDB0G.DB1G.BSDS02.

active log data set prefixes This prefix can be up to 30 characters long. Sample active log data prefixes are DSNDB0G.DB1G.LOGCOPY1 and DSNDB0G.DB1G.LOGCOPY2.

archive log data set prefixes These prefixes can be up to 35 characters long unless you want the data sets time stamped. If they are time stamped with a 2-digit year, only 19 characters can be used. If they are time stamped with a 4-digit year, only 17 characters can be used. Use the TIMESTAMP field of installation panel DSNTIPH to determine which timestamp format you use.

Sample archive log data set prefixes are DSNDB0G.DB1G.ARC1 and DSNDB0G.DB1G.ARC2.

Choosing IRLM names

Each DB2 in the data sharing group must have its own IRLM. The IRLM group name, subsystem name, and member ID are parameters on the IRLM startup procedure. This means that there must be a separate IRLM procedure for every IRLM in the group. Figure 16 shows the relationship between DB2 and IRLM groups and names.

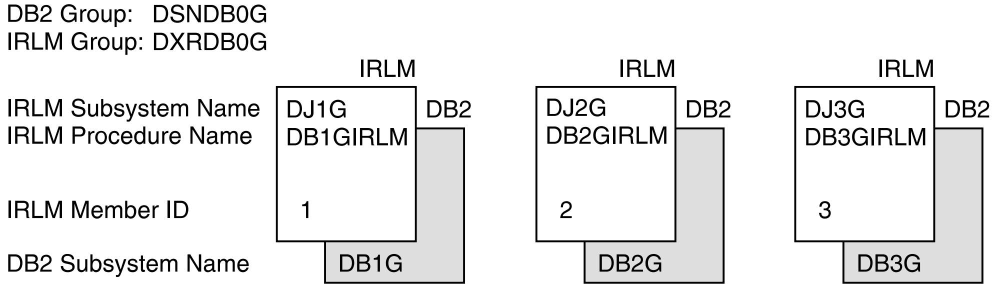


Figure 16. Relationship between DB2 and IRLM Group Names

You must choose the following IRLM names before installing DB2:

group name The name that encompasses the entire IRLM group. This name can be up to 8 characters long and can consist of the characters A-Z, 0-9, $, #, and @. The group name must begin with an alphabetic character. To avoid names that IBM uses for its XCF groups, do not begin with the letters A-I unless the first three characters are DXR. Do not use the string SYS as the first three characters, and do not use the string UNDESIG as your group name.

| The IRLM group name is a parameter (IRLMGRP=) on each DB2 member's IRLM procedure. A sample IRLM group name is DXRDB0G, which is a convention that makes it easy for you to tell which IRLM group is paired with which DB2 group.

| **subsystem name** Each IRLM must have a subsystem name. This name can be up to 4 characters long and is a parameter on the IRLM procedure.

A sample subsystem name is DJ1G. The “1G” characters indicate that this IRLM is paired with the DB2 subsystem DB1G. See “Suggested convention” on page 44 for more information about how this identifier is used.

IRLM procedure name This name can be up to 8 characters long. We recommend using the DB2 member subsystem name followed by “IRLM.”

Each DB2 member knows its IRLM by the procedure and subsystem name saved in that member's installation parameter load module.

A sample IRLM procedure name is DB1GIRLM.

IRLM member ID This ID is a number between 1 and 255 (inclusive). The ID uniquely names an IRLM within a group. See the description of START *irlmproc* in Chapter 2 of *DB2 Command Reference* for more information about this value.

Coupling facility structure names

The names for coupling facility structures must conform to a strict naming convention based on the DB2 group name. Sample names are shown here:

Table 2. Sample Coupling Facility Structure Names

Structure Type	Example Name
Cache structure (group buffer pools)	DSNDB0G_GBP0
Lock structure	DSNDB0G_LOCK1
List structure (shared communications area)	DSNDB0G_SCA

Lock structure name

The name you must use on the CFRM policy to define the lock structure to the coupling facility is:

*groupname*_LOCK1

Shared communications area

The name you must use on the CFRM policy to define the SCA to the coupling facility is:

*groupname*_SCA

Group buffer pool names

The name you must use on the CFRM policy to define the group buffer pool to the coupling facility is:

```
groupname_GBPxxxx
```

Where GBP*xxxx* is the name of the group buffer pool:

- 4KB group buffer pools are named GBP0, GBP1, ..., GBP49
- 8KB group buffer pools are named GBP8K0, GBP8K1, ..., GBP8K9
- 16KB group buffer pools are named GBP16K0, GBP16K1, ..., GBP16K9
- 32K group buffer pools are named GBP32K, GBP32K1, ... , GBP32K9

When DB2 duplexes a group buffer pool structure, the same structure name is used for both the primary and secondary structures. A duplexed structure requires a single CFRM policy structure definition, with one structure name.

Naming recommendations

You have control over the names you assign to entities during the DB2 installation process. **After installation, some names, such as the group name and member names, cannot be changed.** Because you must name a large number of items, and you might have to add new members or move existing members to different systems in the future, managing all these items is easier if you choose and maintain a consistent naming convention.

If you are enabling the originating member of the group from an existing DB2, it is possible to build a naming scheme around the existing names used in your DB2 subsystem to reduce the disruption for existing applications. However, you might want to change some names now, *before* enabling data sharing, to lay the foundation for a solid naming convention.

Another name to choose carefully is the catalog alias for the group. It is very difficult to change that name. The procedure to do this for a single system is documented in Section 2 (Volume 1) of *DB2 Administration Guide*. To change the catalog alias for the group, you have to bring the entire group down and do the procedure for every member of the group.

Assumptions

The DB2 data sharing naming recommendations and installation process support assume the following MVS system Sysplex configuration:

- One SYS1.PARMLIB shared by all MVS systems
- One SYS1.PROCLIB shared by all MVS systems
- One integrated catalog facility master catalog shared by all MVS systems

DB2 data sharing does not require that the MVS Sysplex be configured in this manner. However, the following DB2 naming recommendations support such a configuration, and the DB2 installation process assumes such a configuration.

If the MVS Sysplex is configured differently, you must customize the install process. For example, if you are using different SYS1.PARMLIBs, make sure that the DB2 data sets are in the APF list in each PARMLIB. If you are using different PROCLIBs, modify the JCL to point to the correct libraries during installation. And if you are using more than one integrated catalog facility master catalog, put the DB2 catalog alias in each master catalog.

Suggested convention

Following are some suggestions to consider when naming various DB2 entities:

- The names with the shortest length are the subsystem names. Subsystem names are limited to four characters. After these names are selected, they can be used as the basis for choosing the other names. 4-character names are needed for the following:

  – Group attachment name; one per group

  – DB2 member subsystem name; one for each DB2 member

  – IRLM member subsystem name; one for each DB2 member

  One possible naming convention is the one suggested in *OS/390 Parallel Sysplex Application Migration*, which takes a Sysplex-wide approach to naming. That publication suggests assigning subsystem names of the form `'`*ctmg*`'` where:

  c Denotes a particular collection of logically related applications or subsystems.

  t Denotes a type of resource, such as “B” for DB2 or “J” for IRLM.

  m Denotes a particular member within a DB2 data sharing group or IRLM group. This identifier is also used to associate subsystems with their MVS for recovery purposes when using automatic restart.

  g Denotes a particular DB2 group.

  Our examples use G as the group identifier. Therefore, our naming scheme has a subsystem name of DB1G for the first member of the DB2 data sharing group. The second member is DB2G, and so on. Table 3 shows how the member identifier relates to a particular MVS and how the group identifier associates DB2 members across a Sysplex.

Table 3. Subsystem Names in a Sysplex. Columns are names associated with a particular MVS recovery group (MVS `'2'`, for example). Rows include DB2 and IRLM data sharing groups.

MVS1	MVS2	MVS3	MVS4
DA11 (CICS)	DA21 (CICS)	DA31 (CICS)	DA41 (CICS)
⋮	⋮	⋮	⋮
DB1G (DB2)	DB2G (DB2)	DB3G (DB2)	DB4G (DB2)
DJ1G (IRLM)	DJ2G (IRLM)	DJ3G (IRLM)	DJ4G (IRLM)
DD11 (DBCTL)	DD21 (DBCTL)	DD31 (DBCTL)	DD41 (DBCTL)
⋮	⋮	⋮	⋮

  If you are enabling an existing DB2 subsystem to take advantage of data sharing, and have existing applications using that DB2, consider keeping the existing subsystem name and using it for the group attach subsystem name. This allows existing applications to use the data sharing member without changing to use a new subsystem name.

- Make the DB2 group name, the DB2 location name, and the DB2 integrated catalog alias the same.

A data sharing group uses the same catalog alias name. This catalog alias is used as the high level qualifier for the DB2 directory (DSNDB01), catalog (DSNDB06), default database (DSNDB04), and work file database VSAM data sets.

- Consider making a member name and the member's subsystem name the same.
- For a member's command prefix, use the member's subsystem name with a hyphen (-) in front of it.

DB2 data sharing naming example

Here is an example of names chosen for a 12-member DB2 data sharing group. The example uses the subsystem naming convention suggested above. We use `'0'` (a zero) to represent the group identifier. In Table 4 the '#' character is used to denote a character from the set {1,2,3,4,5,6,7,8,9,A,B,C}. These characters are used to denote a particular member's name.

Table 4. DB2 data sharing naming example

Entity	Name
Group name	DSNDB0G
Catalog alias	DSNDB0G
Group attach name	DB0G
Member names	DB#G
Member command prefix	-DB#G
Member subsystem name	DB#G
Member BSDS	DSNDB0G.DB#G.BSDS01 DSNDB0G.DB#G.BSDS02
Member active log prefix	DSNDB0G.DB#G.LOGCOPY1 DSNDB0G.DB#G.LOGCOPY2
Member archive log prefix	DSNDB0G.DB#G.ARC1 DSNDB0G.DB#G.ARC2
Member work file database	WRKDB#G
Member procedure names	DB#GMSTR DB#GDBM1 DB#GDIST DB#GSPAS
Member subsystem parameters load module	DSNZP0#G
IRLM group name	DXRDB0G
IRLM member subsystem name	DJ#G
IRLM procedure name	DB#GIRLM
IRLM member ID	Number 1-12 corresponding to #

Distributed naming convention: The example in Table 4 does not include names for distributed processing. Those naming conventions are probably part of a much broader convention. See your network administrator for more information about choosing names for distributed access.

Planning for availability

When planning your data sharing configuration for the highest availability, a primary concern should be for the physical protection of the coupling facility and the structures within the coupling facility. The SCA and lock structure are both necessary for the group to function. If something happens to those structures, they can be dynamically rebuilt on another coupling facility. However, if the rebuild fails, all members of the group fail, and this necessitates a group restart.

Although a loss of a group buffer pool does not require a group restart, availability for users and important applications require that data in a group buffer pool be available as quickly as possible after a failure. Group buffer pools have several availability options, depending on the type of failure to occur. For the highest availability, you can duplex the group buffer pool, allowing DB2 to switch to a secondary group buffer pool if the first one fails.

If a simplexed group buffer pool structure fails, the group buffer pool can be recovered automatically from data on the DB2 logs. If members lose connectivity to the group buffer pool, the group buffer pool can be rebuilt onto another coupling facility to which the members can connect.

This section describes the following topics:

- "Using MVS's automatic restart capability"
- "Coupling facility availability" on page 47
- "Duplexing group buffer pools" on page 51
- "DB2 resource availability" on page 53

Using MVS's automatic restart capability

The purpose of using automatic restart is to reduce the time a particular system is down. When DB2 or IRLM stops abnormally, the surviving MVSs analyze the situation to determine whether MVS failed, too, and where DB2 or IRLM should be restarted. It then restarts DB2 or IRLM appropriately.

Advantage of automatic restart

By restarting DB2 and IRLM quickly, locks held by failed members are released quickly, allowing applications running on other DB2 members to access data for which the failed member is holding incompatible locks. Consider using automatic restart in conjunction with the RETAINED LOCK TIMEOUT option of installation panel DSNTIPI described in Table 13 on page 79.

You must have DB2 installed with a command prefix scope of "started" to take advantage of automatic restart. See "Registering command prefixes and group attachment name" on page 68 for more information.

Using an automatic restart policy

You control how automatic restart works by using MVS automatic restart policies. When the automatic restart function is active, the default action (for both sharing and non-sharing DB2s and IRLMs alike) is to restart the subsystems when they fail. If this default action is not what you want, then you must create a policy defining the action you want taken.

Creating the automatic restart policy

If the default action of restarting DB2 is the action you want, you do not need to create a policy. However, if you want to change the default, you need to know the automatic restart ELEMENT name. In a data sharing group, the DB2 ELEMENT is the concatenated DB2 group name and member name (such as DSNDB0GDB1G).

For IRLM, the element name is the IRLM group name, concatenated with the IRLM subsystem name and 3-character member ID (such as DXRDB0GDJ1G001).

You can also specify a wild card (such as DSNDB0G*) if you want to use a single policy statement for all members in the group.

DB2 startup can be a little faster when you have the automatic restart manager start IRLM because the activity is done in parallel. DB2 does not have to start IRLM and wait. To specify that DB2 or IRLM is not to be restarted after a failure, include RESTART_ATTEMPTS(0) in the policy for that DB2 or IRLM element. For IRLM, you can also use the following command to indicate that you want to stop IRLM and to deregister it from automatic restart manager when it comes down. Deregistering prevents IRLM from automatically restarting after you bring it down:

```
MODIFY irlmproc,ABEND,NODUMP
```

However, if your DB2 has YES for the AUTO START option of installation panel DSNTIPI, and if MVS restarts DB2 automatically, DB2 will restart IRLM, too.

Coupling facility availability

For high availability, you should have at least two coupling facilities, and one of those should be a *dedicated* coupling facility, which means that the coupling facility is not running in the same CPC as another MVS system. With two coupling facilities, you can specify that structures be allocated in the second coupling facility in case the first one is damaged.

With more than one coupling facility, you can also consider duplexing the group buffer pools. With duplexing, a secondary group buffer pool is always on standby in another coupling facility, ready to take over should the primary group buffer pool structure fail or if there is a connectivity failure. If you have three or more coupling facilities, you can even maintain duplexing while performing maintenance on one of the coupling facilities. For more information about duplexing group buffer pools, see “Duplexing group buffer pools” on page 51.

It is also recommended that at least one of the coupling facilities be non-volatile.

See *System/390 MVS Sysplex Hardware and Software Migration* for more information about configuring the coupling facility for high availability.

Recommendations for placement

We suggest that you physically separate the coupling facilities from each other and from the rest of the MVS images that are using those coupling facilities. The reason for this is that after a major outage, having the lock and SCA structures separated from the systems using them can minimize the chances of having to do a lengthy group restart.

Quick recovery of the group buffer pools uses information in the lock structure and SCA to determine which databases must be recovered. This is known as *damage assessment*. Consider putting the lock structure and SCA in a different coupling facility than important cache structures (such as group buffer pool 0). This makes it less likely that you would lose the lock structure, SCA, and the group buffer pool at the same time.

If you should lose the lock structure or SCA at the same time as one or more group buffer pools, DB2 waits until the lock structure and SCA are rebuilt before doing damage assessment.

Preparing for structure failures

For structure failures, DB2 can recover resources in the structures. For the lock and SCA, this is done by rebuilding the information in the structures from information contained in DB2's virtual storage. For group buffer pools, DB2 can automatically recover that information by reading the logs and applying the changes to the data. Or, if you are duplexing the group buffer pool, DB2 just switches over to using the secondary group buffer pool.

Lock and SCA structure failures: DB2 can rebuild the lock and SCA in the same coupling facility, or in an alternate coupling facility, assuming:

- You have specified the alternate coupling facility in the CFRM policy preference list.
- You have allocated enough storage in the alternate coupling facility to allow the rebuild of the structures there.

The information used to recover the lock structure and SCA is contained in DB2's virtual storage, not the logs. If the lock structure and SCA cannot be rebuilt, all active members in the group terminate abnormally, and a group restart is required to recover the lost information from the logs.

Group buffer pool structure failure: Group buffer pools can be recovered from the log when they fail, or you can switch to a secondary group buffer pool if the CFRM policy for the group buffer pool indicates that duplexing is allowed and the group buffer pool is currently running in duplexed mode.

Recovery from the log can occur manually, as the result of a START DATABASE command, or it can occur automatically, because the group buffer pool is defined with the AUTOREC(YES) option. In either case, to reduce the time needed for group buffer pool recovery, use the ALTER GROUPBUFFERPOOL command to make group buffer pool checkpoints more frequent. However, weigh the benefit of faster recovery with the processing resources used for the checkpointing. (You can reduce checkpoint costs considerably when OS/390 is at the appropriate level of maintenance. See “Tuning the group buffer pool checkpoint interval” on page 239 for more information.) Checkpointing is also used with duplexed group buffer pool, in case both structures fail.

Consider also using DB2's fast log apply to speed up recovery. To enable fast log apply, indicate how much *ssnm*DBM1 storage can be used for the log apply function on the LOG APPLY STORAGE field of installation panel DSNTIPL.

Be sure to specify one or more alternate coupling facilities in the CFRM preference list for the group buffer pools because a group buffer pool can be allocated in an alternative coupling facility when a new connection is made to it. See “Problem: group buffer pool structure failure (no duplexing)” on page 164 for more information about group buffer pool structure failures.

More about automatic recovery: Automatic recovery is enabled by the AUTOREC option of the ALTER GROUPBUFFERPOOL command. Automatic recovery is faster than manual recovery because DB2 can optimize internal processing of the recovery. For automatic recovery to be initiated for a group buffer pool, all of the following conditions must be true:

- AUTOREC (YES) is specified for the group buffer pool.
- There was at least one actively connected member at the time that the group buffer pool failed. This member must have successfully completed damage assessment.

DB2 never initiates automatic recovery during restart.

For a duplexed group buffer pool, DB2 can use automatic recovery if both instances of the group buffer pool are damaged. Automatic recovery is not needed for group buffer pools that are defined as GBPCACHE (NO).

Preparing for connectivity failures

To prevent having DB2 fail because of a coupling facility channel failure, consider having dual channels (sometimes called *links*) between each CPC and a coupling facility. Without dual links, a channel failure is more likely than a failure in the coupling facility. Losing connectivity to the SCA or lock structure can bring that particular DB2 member down, unless you specify an alternative coupling facility in the CFRM policy preference list.

A total failure of the coupling facility, such as a power failure to the coupling facility or some problem with coupling facility control code, is also interpreted as a connectivity failure by MVS and DB2. See "Problem: all members have lost connectivity" on page 160 for more information about recovery from lost connections.

Rebuilding structures when connectivity is lost: This section describes the recovery action when some or all DB2 members have lost their connectivity to the simplexed group buffer pool. See "What if the group buffer pool is actively duplexing?" on page 50 for information about connectivity failures for duplexed group buffer pools.

When connectivity is lost, DB2 can rebuild those structures on the alternate coupling facility that is specified in the CFRM policy. To make this happen, you **must** have an active Sysplex failure management (SFM) policy or MVS with APAR OW30814 installed. When DB2 rebuilds, it attempts to allocate storage on the alternate coupling facility. It uses the value of INITSIZE for the structure on the CFRM policy of the alternate coupling facility. If DB2 cannot allocate the storage for the SCA or lock structure, the rebuild fails. If MVS cannot allocate the storage for the group buffer pools, the changed pages are written to DASD. See "Connectivity failure for lock structure and SCA" on page 157 and "Connectivity failure for non-duplexed group buffer pools" on page 157 for more information about failure scenarios.

To control when a rebuild can occur, MVS lets you specify a *rebuild threshold*. This value is a percentage that you specify in the CFRM policy on the REBUILDPERCENT parameter. MVS uses the REBUILDPERCENT value in the CFRM policy to determine whether to initiate a structure rebuild when there is a loss of connectivity to the coupling facility that contains the structure. The percentage is based on the SFM *weights* of all the systems that have active connectors to a structure at the time. You also specify weights on the SFM policy.

MVS calculates the total system weight of (A) all systems with at least one active connection to the structure that have lost connectivity, and (B) all systems with at least one active connection to the structure. If there are multiple connections to a structure from a single system (for example, two DB2 members on the same MVS), then that system weight is counted only once.

For example, if a group buffer pool has one connection per MVS system and all systems are of equal weight 10, then in an eight-system sysplex if one system lost connectivity, then the value of A (total system weight of all systems containing an active connection that have lost connectivity) is 10 and the value of B (total system weight of all systems containing an active connection) is 80.

MVS determines whether or not to rebuild as follows:

- If (A/B) × 100 is greater than or equal to the REBUILDPERCENT value, then rebuild.
- Otherwise, don't rebuild. Instead, the affected DB2 disconnects from the group buffer pool.

In the example above, (10/80) × 100 is the value compared to the REBUILDPERCENT value. If the value of REBUILDPERCENT is 25, then a rebuild is not initiated. See “Problem: a subset of members have lost connectivity” on page 162 for more information about this situation.

REBUILDPERCENT(100), the default, means that the structure is not automatically rebuilt unless all the connected members lose connectivity to the structure. Certain types of coupling facility failure, such as a power failure, are treated by MVS as a complete loss of connectivity and are handled by DB2 as a structure failure. See “Problem: all members have lost connectivity” on page 160 for more information about this situation.

A larger rebuild percent is most appropriate for a group of many small systems. In such a group, you might prefer to have one member lose the use of one group buffer pool rather than temporarily disrupt all of the members using the structure while the structure is rebuilt. If you have high availability requirements, it might be better to always allow the automatic rebuild by specifying a small rebuild percent. For more information, see *OS/390 MVS Setting Up a Sysplex*.

What if the group buffer pool is actively duplexing?: If the group buffer pool is duplexed, the recovery action is to use the group buffer pool with good connectivity. This action is better than rebuilding because it is far less disruptive. If your CFRM policy specifies DUPLEX(ENABLED), then automatic reduplexing is attempted.

Monitoring rebuild events

You can use a performance class 20 trace (IFCIDs 0267 and 0268), or you can examine the messages returned to the console to monitor how long a rebuild of a structure takes and the reason for the rebuild (such as lost connectivity, operator command, or to establish duplexing).

Coupling facility volatility

There are times when a coupling facility can go into a *volatile* state, which means that if the power fails, data in the coupling facility at the time of failure is not saved. If the coupling facility is configured to be non-volatile (using the proper power backup), volatility is generally a transient state that might occur, for example, if you take the battery out.

DB2 issues a warning message if allocation occurs into a volatile coupling facility. A change in volatility after allocation does not have an effect on your existing structures.

Advantages of a non-volatile coupling facility: If you lose power to a coupling facility that is configured to be non-volatile, the coupling facility enters power save mode, saving the data contained in the structures. When power is returned, there is no need to do a group restart, and there is no need to recover the data from the group buffer pools. For systems requiring high availability, non-volatile coupling facilities are recommended.

For more information about coupling facility volatility options, see *System/390 9672/9674 System Overview*.

Duplexing group buffer pools

Running some or all of your group buffer pools in duplex mode is one way to achieve high availability for group buffer pools across many types of failures, including lost connections and damaged structures.

How duplexing works

With a duplexed group buffer pool, you have two allocations of the same group buffer pool that use one connection from each member. Each structure allocation must be in a different coupling facility. MVS prefers to place the structures in different coupling facilites that are failure-isolated from one another. One allocation is called the *primary structure*. DB2 uses the primary structure to keep track of page registration and cross-invalidation, and that primary structure is the one from which changed data is cast out to DASD. From an MVS perspective, duplexing is really an extended rebuild, so OS/390 documentation and commands sometimes call the primary structure the *old* structure.

The other allocation of the structure is called the *secondary structure* (referred to by MVS as the *new* structure). When changed data is written to the primary structure, it is also written to the secondary structure.

MVS commands let you stop and start duplexing and switch which structure is the primary structure.

We recommend that at least one of the group buffer pool structures be in a non-volatile coupling facility. If power is lost to both coupling facilities and both coupling facilities are volatile, then the group buffer pool must be recovered from the logs.

Characteristics of the secondary structure: These characteristics are different than those of the primary structure:

- DB2 does not read data from the secondary structure.
- DB2 does not use the secondary structure for cross-invalidation of pages.

- DB2 does not cast out data to DASD from the secondary structure.

When planning for storage, secondary and primary group buffer pools are the same size. If you are already properly configured for availability, there is usually no extra coupling facility storage required for duplexing. Instead of having unused storage to take over in case of a coupling facility failure, that storage is instead used by the secondary structure.

Coupling facility storage considerations for duplexing: Duplexing usually does not require any additional coupling facility storage beyond that which you would use for a highly available simplexed group buffer pool. For simplexed structures, you must reserve enough spare capacity in the coupling facilities to be able to absorb the structures of any failed coupling facility. For example, if you have two coupling facilities, each with 1 GB of memory for a total of 2 GB, then you need to ensure that the total size of the structures across the two coupling facilities does not exceed 1 GB (50% of the total coupling facility storage). With duplexing, you are using that previously reserved storage for the secondary structure, and you would not need to configure extra coupling facility storage for duplexing in this case. If you have three or more coupling facilities configured, then you might need additional coupling facility for duplexing.

Requirements

For duplexing to work, the following conditions must be true:

- There must be at least two coupling facilities with a CFLEVEL of 5 or higher in the CFRM policy preference list for the group buffer pool. All members of the data sharing group must have physical connectivity to both coupling facilities in which the primary and secondary structures reside.

  If you are going to do automatic reduplexing, you need three coupling facilities that are physically connected to members of the data sharing group.
- At least one DB2 member must be actively connected to the group buffer pool.
- All connected DB2s must be duplexing-capable. To be capable means that each connected DB2 member must either be at Version 6 or a subsequent release, or at Version 5 with APAR PQ17797 applied.
- All connected DB2s must be running on OS/390 Release 3 or a subsequent release with APAR OW28460 applied. (The function is included in the base for OS/390 Release 6 or subsequent releases.)
- The group buffer pool must be defined with GBPCACHE(YES), the default.

Establishing duplexing

There are three options on the CFRM policy for duplexing. DUPLEX(ENABLED) automatically starts duplexing. DUPLEX(ALLOWED) is not automatic. DUPLEX(DISABLED) disables duplexing. You must issue a command to start duplexing. For more information about starting and stopping duplexing, see “Starting and stopping duplexing for a group buffer pool” on page 178.

Performance of duplexing

The process of establishing duplexing can be somewhat disruptive because access to the group buffer pool is quiesced while the secondary structure is allocated and changed pages are copied from the primary structure to the secondary structure (or cast out to DASD). Transactions that need access to the group buffer pool during this process are suspended until the process is complete. Because of this disruption, it is best to establish duplexing at a time of low activity on the system. How long the process takes depends upon how many pages are copied to the secondary group buffer pool.

In general, it takes a bit more processor and elapsed time to do duplexed group buffer pool writes and castout processing than it does to do simplexed group buffer pool writes and castout processing. Workloads that are more update-intensive will probably experience a slight increase in host CPU usage when duplexing is activated. In most cases, the majority of the CPU increase will occur in the DB2 address space. Duplexing can cause a slight increase in the transaction elapsed time. Read performance is unaffected by duplexing.

You will also see an increase in the coupling facility CPU usage in the CF that contains the secondary structure. You can estimate about how much the CF CPU usage will increase when you establish duplexing as follows:

1. Determine the amount of CF CPU that the simplexed Primary structure consumes.
2. Divide the result in half to determine how much CF CPU that the duplexed Secondary structure will consume.

Duplexing should have little or no impact on the CF CPU usage for the CF containing the Primary structure.

The statistics and accounting trace classes contain information about group buffer pool duplexing.

Monitoring duplexing rebuilds: When a group buffer pool is duplexed, it is considered to be in an extended rebuild status called a duplexing rebuild. This activity is reported in IFCIDs 0267 and 0268 along with other reasons for rebuilding.

DB2 resource availability

DB2 availability considerations for a data sharing group are basically the same as for a single subsystem. This section describes high availability options for the catalog and directory, for data in group buffer pools, and for DB2 restart.

Catalog and directory

For a data sharing group, the catalog and directory data sets are even more important than with a single subsystem because there is a single catalog and directory for all members of the group. Consider placing the catalog and directory behind a 3990 control unit with dual write capability for hardware duplexing. Another possibility is to use the RAMAC® Array Subsystem for its improved availability benefits.

Group buffer pool data

Assign data that require high availability to group buffer pools that reside on non-volatile coupling facilities and consider using group buffer pool duplexing.

Another possible option for certain cases might be to assign the data to a GBPCACHE(NO) group buffer pool, or defining the page set as GBPCACHE NONE. The performance penalties for this option can be quite high, but there are certain types of applications that can perform better with this option. See “GBPCACHE NONE” on page 234 for more information. The coupling facility is still used for cross-invalidation, so GBPCACHE NONE page sets are affected by coupling facility failures.

Data availability at restart

To make DB2 restart faster, enable fast log apply. Enable fast log apply by specifying some amount of storage on the LOG APPLY STORAGE field of installation panel DSNTIPL.

If your installations sometimes has problems with long-running units of recovery (URs) taking a long time to back out after a failure, make plans to reroute work to other members of the group. Another solution is to postpone backout processing for those long-running URs until DB2 is up and receiving new work. See “Postponing backout processing” on page 175 for more information.

Estimating storage

This section gives you information about estimating storage for coupling facility structures and for DB2 resources. The following topics are described:

- “General information about coupling facility storage”
- “Group buffer pool sizes” on page 55
- “Lock structure size” on page 61
- “SCA size” on page 62
- “Changing structure sizes” on page 63
- “Estimating a value for the IRLM MAXCSA parameter” on page 63
- “Storage for DB2 objects” on page 66

General information about coupling facility storage

It is difficult to give precise estimates for coupling facility structure sizes used in DB2, partly because every environment is different, and partly because storage allocation is affected by the processor model and level of coupling facility control code you have. Use the information given here for your initial estimate for initial size values (INITSIZE) and, depending on how much your work load varies, give a larger value for SIZE.

You can use an MVS SETXCF START,ALTER command to alter the structure to as large as SIZE. There is no minimum limit enforced by the CFRM policy, so you can alter the size to be smaller than INITSIZE if you want. See “Changing structure sizes” on page 63 for more information.

For duplexed group buffer pools, the SIZE and INITSIZE values apply to both instances of the structure.

The information given in this section assumes that all page sets in a particular group buffer pool are defined with the same GBPCACHE attribute. You can put page sets with different GBPCACHE attributes in the same group buffer pool, but you must adjust the formulas accordingly.

Don't overestimate the SIZE parameter

Coupling facility structures contain some control structures that are static. When the structure is initially allocated, these static structures are allocated to accommodate the potential size of the structure. In other words, the size of the static structures is proportional to the maximum size (the SIZE parameter). If the SIZE is very much larger than the INITSIZE, there might be cases where a large percentage of the INITSIZE is used for these static structures, leaving you little in the way of usable storage for storing items in the structure.

In general, it is a good idea to specify a SIZE that is larger than INITSIZE but to keep the SIZE parameter to no more than 2 to 3 times the INITSIZE for the lock and SCA, and to no more than 4 times the INITSIZE for group buffer pools. For example, if INITSIZE for a group buffer pool is 100 MB, specify a SIZE of 400 MB or less.

Other sources of information

For the cache structures (group buffer pools) we give both “rule of thumb” estimates and input you can use for the storage formulas given in an appendix of *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide*. Consult this guide if you are looking for detailed information about planning for storage in the coupling facility.

When you decide what your structure sizes are, include those values in the CFRM policy definition. See *OS/390 MVS Setting Up a Sysplex* for more information about creating CFRM policies.

Group buffer pool sizes

A group buffer pool (coupling facility cache structure) consists of two parts: data pages (sometimes called *data entries*) and directory entries.

Data pages: This is the place in the group buffer pool in which the data page resides. The size of a data page is the same as the page size supported by the corresponding DB2 buffer pools (that is, 4KB, 8KB, 16KB, or 32KB).

If you are caching changed data only, you need enough space to cache changed data plus enough extra for pages that are frequently re-referenced. By having those frequently referenced pages in the group buffer pool, you can decrease the amount of time it takes for any member to refresh that page in its member buffer pool because you are avoiding the DASD I/O.

If you choose GBPCACHE NONE or SYSTEM, no user data pages are actually stored in the coupling facility. However, with GBPCACHE SYSTEM, space map pages for LOBs are cached in the coupling facility.

Directory entries: A directory entry specifies the location and status of an image of a page somewhere in the data sharing group, whether the image appears in the group buffer pool or in one of the member buffer pools. There is only one directory entry for any given page, no matter in how many places that page is cached.

The size of a directory entry is approximately 200 bytes, but it varies somewhat based on the size of the data pages and the CFLEVEL you are using. See *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide* for the exact size.

Specifying a ratio: The space allocated for a group buffer pool is divided into two parts according to the ratio of the number of directory entries to the number of data pages. When you originally define a structure in the CFRM policy for a group buffer pool, you specify its total size. For DB2, the ratio defaults to five directory entries per data page. Later, you can change the ratio with the ALTER GROUPBUFFERPOOL command. That new value takes effect when the group buffer pool is rebuilt or reallocated. See "Determining the correct size and ratio" on page 247 for information about detecting problems with the size and ratio of group buffer pools.

For group buffer pools defined with GBPCACHE(NO), ratios are ignored because no data is actually stored in the group buffer pool.

In this section: When possible, both a formula and rules-of-thumb are provided to help you estimate the initial sizes and ratios of your group buffer pools. (The exception is for GBPCACHE ALL group buffer pools, for which we provide only a rule-of-thumb.)

The formula is not too complex and is likely to be more accurate, assuming that you are fairly confident of the values for the variables in the formulas. Otherwise, use the rules of thumb and then adjust from there.

Storage estimate for group buffer pools that cache changed data

The size of a group buffer pool is related to the amount of sharing and the amount of updating. An estimate must be based on the total amount of member buffer pool storage multiplied by a percentage based on the amount of update activity. The more sharing and updating there is, the more pages must be cached in the group buffer pool and the more directory entries are needed to track inter-DB2 buffering.

Formula: The formula for estimating storage for group buffer pools that cache changed data is:

```
Data_entries = U * D * R
Data(MB)     = Data_entries * P / 1024
Dir_entries  = Data_entries + (U * (VP+HP))
Dir(MB)      = 1.1 * Dir_entries * 0.2 / 1024
GBP(MB)      = Data(MB) + Dir(MB)
RATIO        = Dir_entries / Data_entries
```

Where:

U A variable related to the estimated degree of data sharing:

- **1** A high amount of sharing with a lot of update activity
- **0.7** A moderate amount of sharing with a moderate amount of update activity
- **0.5** A low amount of sharing with a low amount of update activity

D The number of data pages written to DASD per second for all members, peak rate. Do not use the number of pages written to the group buffer pool; it must be a count of distinct pages. To determine this value, use the field

QBSTPWS from IFCID 0002 (the PAGES WRITTEN field of the buffer pool section of the DB2 PM Statistics report.)

R The average page residency time in the group buffer pool, in seconds. This value is application-dependent, but you can assume that the typical range is 30 to 180 seconds. If you have no information about residency time, use 120.

In general, make this value high enough so that when a changed page is written to the group buffer pool and invalidates local copies of the page in other DB2 members, the page remains resident in the group buffer pool long enough for other members to refresh the page from the group buffer pool if they need to re-reference their local copy of the cross-invalidated page.

P The page size (4, 8, 16, or 32).

HP The number of data pages defined for the hiperpool (the sum across all the members).

VP The number of data pages defined for the virtual pool (the sum across all the members).

0.2 The approximate size of a directory entry, in KB.

1.1 The additional storage needed for coupling facility control structures.

Example: Assume that you have a 2-member data sharing group for which you have determined the following information:

- The degree of data sharing is very high (1)
- There are 500 DASD writes per second across both members
- The page size is 4KB
- Member 1 is configured with a virtual pool of 80000 buffers and a hiperpool of 160000 buffers
- Member 2 is configured with a virtual pool of 40000 buffers and a hiperpool of 80000 buffers

The calculation is as follows:

```
Data_entries = 1 * 500 * 120 = 60000
Data(MB)     = 60000 * 4 / 1024 = 234 MB
Dir_entries  = 60000 + 1 * (240000 + 120000) = 420000
Dir(MB)      = 1.1 * 420000 * 0.2 / 1024 = 90 MB
GBP(MB)      = 234 MB + 90 MB = 324 MB
RATIO        = 420000  /60000 = 7.0
```

The above calculation indicates that the group buffer pool should be defined with an INITSIZE of 324MB. Use the ALTER GROUPBUFFERPOOL command to change RATIO to 7.

Rule-of- thumb: For installation planning purposes, the following rule-of-thumb is offered as an initial estimate for the size of a DB2 group buffer pool for table spaces, indexes, or partitions that cache only changed data (GBPCACHE CHANGED):

Sum the local buffer pool storage for this buffer pool number (both virtual and hiperpool) across all the DB2s of the group. Then, multiply this amount by one of these factors:

Factor	Condition
10%	for light sharing with a low amount of updating
20%	for medium sharing with a moderate amount of update activity
40%	for a high amount of sharing with a lot of update activity

You can run a trace for IFCID 0002 to obtain an estimate of the amount of data sharing in your system. Calculate the "degree of data sharing" by dividing QBGLGG, the number of get pages for group buffer pool-dependent objects, by QBSTGET, the number of get pages. A value that is less than 25 percent is considered to be light data sharing, a value between 25 and 75 percent is medium data sharing, and a value greater than 75 percent is high data sharing.

Bear in mind that the type of workload you run can influence the amount of storage used. For example, if you have “hot spots” in which a single page gets updated frequently rather than having updates spread throughout the table space, then you might not need as much storage for caching.

Example: The total virtual buffer pool storage for all the DB2s of the group is 400MB. A medium amount of read/write sharing is expected in the environment. The calculation is now:

```
400MB x 20% = 80MB
```

Rule-of-thumb storage estimate for caching all data

For installation planning purposes, the following rule-of-thumb provides an initial estimate of the size of a DB2 group buffer pool when the installation is caching read-only pages with changed pages (GBPCACHE ALL):

Calculate the sum of local buffer pool storage for this buffer pool number (virtual only) across all the DB2s of the group. Then, multiply this amount by one of these factors:

Factor	Condition
50%	Few table spaces, indexes, or partitions specify GBPCACHE ALL
75%	Half the table spaces, indexes, or partitions specify GBPCACHE ALL
100%	Almost all the table spaces, indexes, or partitions specify GBPCACHE ALL for a heavy sharing environment

Example: The local virtual buffer pool storage (do not count hiperpool storage) on all the DB2s of the group adds up to 200 megabytes. Half of the page sets coming into the pool are defined as GBPCACHE ALL. The calculation is now:

```
200MB x 75% = 150MB
```

Storage estimate for caching no data

The formula for estimating storage for group buffer pools that cache no data is:

```
  Dir_entries  = U * (VP+HP)
  Dir(MB)      = 1.1 * Dir_entries*0.2/1024
  GBP(MB)      = Dir(MB)
  RATIO        = n/a
```

If the group buffer pool itself is defined with GBPCACHE(NO), then the ratio is ignored.

The variables are the same as described in “Formula” on page 56. In summary, they are:

U The estimated degree of data sharing.

P The page size (4, 8, 16, or 32).

HP The number of data pages defined for the hiperpool (the sum across all the members).

VP The number of data pages defined for the virtual pool (the sum across all the members).

Example: Assume that you have a 2-member data sharing group for which you have determined the following information:

- The degree of data sharing is very high (1)
- Member 1 is configured with a virtual pool of 80000 buffers and a hiperpool of 160000 buffers
- Member 2 is configured with a virtual pool of 40000 buffers and a hiperpool of 80000 buffers

The calculation is as follows:

```
Dir_entries  = 1 * (240000 + 120000) = 360000
Dir(MB)      = 1.1 * 360000 * 0.2 / 1024 = 77 MB
GBP(MB)      = 77 MB
```

The above calculation indicates that the group buffer pool should be defined with an INITSIZE of 77MB. Use the ALTER GROUPBUFFERPOOL command to change the GBPCACHE attribute to NO. If you put GBPCACHE NONE page sets in a GBPCACHE(YES) group buffer pool, then the calculation becomes more complicated, because the RATIO is observed and you are probably going to waste a lot of space on unneeded data entries.

Storage estimate for caching LOB space maps (GBPCACHE SYSTEM)

The formula for estimating storage for group buffer pools that cache LOB space map data is as follows.

```
Data_entries = (U * D / 10) * R
Data(MB)     = Data_entries * P / 1024
Dir_entries  = Data_entries + (U * (HP + VP))
Dir(MB)      = 1.1 * Dir_entries * 0.2 / 1024
GBP(MB)      = Data(MB) + Dir(MB)
RATIO        = MIN(Dir_entries / Data_entries, 255)
```

The variables are the same as described in “Formula” on page 56. In summary, they are:

U The estimated degree of data sharing.

D The number of data pages written to DASD per second for all members, peak rate. Do not use the number of pages written to the group buffer pool; it must be a count of distinct pages. To determine this value, use the field QBSTPWS from IFCID 0002 (the PAGES WRITTEN field of the buffer pool section of the DB2 PM Statistics report.)

10 Estimate of the LOB system pages that are written for every LOB data page.

P The page size (4, 8, 16, or 32).

R The average page residency time in the group buffer pool, in seconds.

HP The number of data pages defined for the hiperpool (the sum across all the members).

VP The number of data pages defined for the virtual pool (the sum across all the members).

Example: Assume that you have a 2-member data sharing group for which you have determined the following information:

- The degree of data sharing is moderate (.7)
- There are 10 DASD writes per second for across both members, peak rate
- The space map page is resident in the group buffer pool page for 120 seconds
- The page size is 32 KB
- Member 1 is configured with a virtual pool of 20000 buffers and a hiperpool of 70000 buffers
- Member 2 is configured with a virtual pool of 10000 buffers and a hiperpool of 20000 buffers

The calculation is as follows:

```
Data_entries = ((.7 * 10)/ 10) * 120 = 84
Data(MB)     = 84 * 32 / 1024 = 2.6 MB
Dir_entries  = 84 + (.7 * (90000 + 30000)) = 84084
Dir(MB)      = 1.1 * 84084 * 0.2 / 1024 = 18.6 MB
GBP(MB)      = 2.6 MB + 18.6 MB = 21.2 MB
RATIO        = MIN (84084 / 84, 255)  = 255
```

The above calculation indicates that the group buffer pool should be defined with an INITSIZE of 21.2 MB. The ratio is greater than the maximum value, which is not unusual with GBPCACHE(SYSTEM), so use the ALTER GROUPBUFFERPOOL command to change the ratio to 255.

PR/SM™ formulas for calculating group buffer pool sizes

Another way to calculate group buffer pool sizes is to use the coupling facility allocation formulas for cache structures found in *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide* Table 5 contains information used in those formulas. The size of cache structures in DB2 can vary greatly based on the amount of data for which there is inter-DB2 R/W interest at any given time. You will most likely have to monitor the use of the group buffer pools and adjust their sizes accordingly.

Table 5 (Page 1 of 2). Information for calculating cache structure sizes

Parameter	DB2 Value	Explanation
MSC	**1**	Maximum storage class
MCC	**1024**	Maximum castout class

Table 5 (Page 2 of 2). Information for calculating cache structure sizes

Parameter	DB2 Value	Explanation
MDAS	Dependent on page size being cached: **1** for 4KB pages **2** for 8KB pages **4** for 16KB pages **8** for 32KB pages	Maximum data area size.
DAEX	**4**	Data area element characteristic.
AAI	**0**	Adjunct assignment indicator.
R_de	Set on ALTER GROUPBUFFERPOOL. Default is **5**. See Table 6 on page 61 for more information on determining this value.	The directory portion of the directory-to-data ratio.
R_data	Set on ALTER GROUPBUFFERPOOL command. See Table 6 on page 61 for more information on determining this value.	The data object portion of the target directory-to-data ratio.

Table 6. Formulas for determining R_data and R_de. N is the RATIO entered on ALTER GROUPBUFFERPOOL command.

Page Size	N has no decimal point	N has decimal point
4KB R_de	N	N×10
4KB R_data	1	10
8KB R_de	N	N×10
8KB R_data	2	20
16KB R_de	N	N×10
8KB R_data	4	40
32KB R_de	N	N×10
32KB R_data	8	80

Lock structure size

The coupling facility lock structure contains two parts. The first part is a *lock table* used to determine if there is inter-DB2 R/W interest on a particular *hash class* (resources that hash to a particular place in the lock table). The second part is a list of the update locks that are currently held (sometimes called a *modify lock list*). IRLM determines how the space of the lock structure is divided between these two parts, and the total size of the lock structure must be large enough to prevent performance problems.

IRLM reserves space for “must complete” functions (such as rollback or commit processing) so that a shortage of storage does not cause a DB2 subsystem failure. However, if storage runs short in the lock structure, there can be an impact on availability (transactions are terminated), response time, and throughput. See “Monitoring DB2 locking” on page 214 and “Changing the size of the lock structure” on page 219 for more information.

Specifying the lock entry size

The LOCK ENTRY SIZE field of installation panel DSNTIPJ determines how much space is needed for lock contention control information (that is, individual entries in the *lock table*). The LOCK ENTRY SIZE of the first IRLM to join the group determines the lock entry size for the lock table. The default is 2 bytes, which is probably the number you want, unless you are immediately creating a data sharing group of seven or more members.

By restricting each lock entry to two bytes, you maximize the number of entries in the lock structure. This can help avoid *false contention*, described in “Avoid false contention” on page 207.

Storage estimate for the lock structure

For installation planning purposes, the initial size of the lock structure is based on how much updating you do. Table 7 gives you size values to start with.

Recommendation: Choose a value that is a power of 2. With a power of 2, IRLM allocates half the lock structure for the lock table and the other half for the modify lock list. (The modify lock list is the part of the lock structure most susceptible to storage shortages.) If you do not specify a power of 2, the allocation is not as predictable, although IRLM does try to keep the allocation as close to 1:1 as possible.

Table 7. Recommendations for lock structure size

INITSIZE	SIZE	Condition
16 MB	32 MB	for light sharing with a low amount of updating, or for a single-member data sharing group
32 MB	64 MB	for medium sharing with a moderate amount of update activity
64 MB	128 MB	for a high amount of sharing with a lot of update activity

SCA size

The shared communications area (SCA) is a list structure in the coupling facility that contains exception information for objects in the database. log data set names, and BSDS names. A table for estimating the size of the SCA is shown in Table 8. The SCA size can be specified in 1KB increments.

Table 8. Estimating storage for the SCA

Site Size	Databases	Tables	INITSIZE	SIZE
Small	50	500	8 MB	16 MB
Medium	200	2000	16 MB	32 MB
Large	400	4000	32 MB	64 MB
Extra Large	600	6000	48 MB	96 MB

Running out of space in the structure can cause DB2 to come down. Because much of the space in the SCA is taken up with exception information, you reclaim space by correcting database exception conditions.

Changing structure sizes

You can change the size of structures by changing the CFRM policy and then rebuild by using the SETXCF START,REBUILD command of MVS. (This command does not work on group buffer pools that actively being duplexed.)

MVS attempts to reallocate a new instance of the structure in the same coupling facility, if that coupling facility is healthy. If there is not enough room, MVS looks at the preference list and uses the alternate coupling facility specified there. After the space is allocated, DB2 rebuilds the information into the new structure. Any transactions that need the structure must wait until the rebuild is complete. Thus, it is best to rebuild when other activity in the system is low.

Dynamically changing the structure size: If the affected structure is allocated in a coupling facility with CFLEVEL greater than 0, you can dynamically change the structure sizes up to a maximum limit specified on the CFRM policy using the MVS SETXCF START,ALTER command.

There are the following advantages to this method:

- It allows DB2 to access the structures while a change is taking place.
- It requires less coupling facility storage, because it does not have to allocate enough space for a whole new structure. It dynamically adds or deletes storage from the existing structure.
- It works on duplexed group buffer pools.

A disadvantage to this method is that it cannot be used to increase the size of the lock table portion of the lock structure. You must change the CFRM policy and manually rebuild it for that change to take effect. If the structure is deallocated and then reallocated, the size of the structure reverts to INITSIZE in the CFRM policy, and is not the size specified on the most recent MVS SETXCF START,ALTER command.

For more information, see:

- “Changing the lock structure size dynamically” on page 219 for the lock structure
- “Problem: storage shortage in the SCA” on page 166 for the SCA
- “Dynamic method” on page 256 for the group buffer pool

Estimating a value for the IRLM MAXCSA parameter

The requirements for IRLM lock storage are described in Section 2 of *DB2 Installation Guide*. If you specify PC=NO on the IRLM startup procedure, you control the amount of storage used for locks with the MAXCSA parameter of the IRLM startup procedure. (For more information about the PC parameter, see “Choosing a value for the PC parameter” on page 64.)

Recommendation: Use PC=NO on the IRLM startup procedure and to set the MAXCSA value at the high end of your estimates. IRLM does not take the storage unless it needs it, and you can change the amount of MAXCSA dynamically with the MVS command MODIFY irlmproc,SET,CSA. If you increase MAXCSA, you might need to increase the CSA value in SYS1.PARMLIB, too.

For data sharing, plan for additional storage because of additional data sharing locks called P-locks. These locks are held on open page sets and on database descriptors (DBDs), skeleton cursor tables (SKCTs), and skeleton package tables (SKPTs). Unlike transaction locks, storage for P-locks is held even when there is no transaction activity, and therefore they consume storage even with no transaction activity. See “P-locking” on page 227 for more information about P-locks.

Plan also for extra storage that IRLM needs to build retained locks in case other members fail. The variables you need to account for are shown in Table 9.

Table 9. Variables used to estimate additional IRLM storage for MAXCSA

Variable	Description	Calculation
I	Non-data-sharing MAXCSA	MAXCSA is ignored if PC=YES. For PC=NO, see Section 2 of *DB2 Installation Guide* for information about calculating this value.

X	P-locks	`N = (MAX_OPEN_DATA_SETS × 500)`
		`X = N + (N × .40)`

Note: The formula assumes that more than one P-lock might be held on a page set occasionally (such as for castout activity) and estimates about 40 percent for P-locks on the EDM pool objects and for short-lived page P-locks. If you know that your EDM pool has relatively few objects in it, you could use a lower percentage for that value. See Section 5 (Volume 2) of *DB2 Administration Guide* for more information about estimating the maximum number of open data sets, or use the value specified for the DSMAX subsystem parameter.		
Y	Ability to hold update retained locks for a failed member	Dependent on how update-intensive the workload is. Start with the following: `Y= .25X`

To calculate a MAXCSA, use the formula:

```
I + X + Y = MAXCSA
```

For example, suppose that your non-data-sharing IRLM storage estimate is 5 MB. If you estimate that this DB2 member could have as many as 8000 open data sets, you could calculate the IRLM storage as follows:

```
(8000 × 500) + 1600000 = 5.47MB
+
1 MB (approximate for retained locks)
+
5MB (non-data-sharing estimate)
-----------------------
Total IRLM storage = 11.47MB
```

Choosing a value for the PC parameter

Use the PC (program call) parameter of the IRLM startup procedure to specify where IRLM lock storage resides:

PC=YES Lock storage resides in IRLM private storage. The MAXCSA parameter is ignored.

PC=NO Lock storage resides in ECSA. Use MAXCSA to control the amount of lock storage.

Advantage of PC=YES: This option might be best for you if you do not have enough ECSA storage to avoid IRLM storage problems.

Advantages of PC=NO: PC=NO is superior in terms of performance, although the advantage is not as clear as it is in non-data-sharing. For data sharing case, the processor cost of the PC linkage is somewhat diluted because lock and unlock requests are propagated to the coupling facility. However, PC=YES can cause the “in IRLM” time to increase, thus possibly causing more IRLM latch contention.

For serviceability, PC=NO is preferred.

Set IRLM's priority high

Be sure to follow the guidelines documented in Section 5 (Volume 2) of *DB2 Administration Guide* for setting the priority of IRLM when using workload manager. If IRLM priority is too low, storage might not be freed as quickly, and IRLM might run out of storage.

Monitoring IRLM storage use

Use the MVS command MODIFY irlmproc,STATUS,STOR to see how much storage IRLM is using. You can see information about accountable storage (that counted against MAXCSA) and about storage that is not counted against MAXCSA. For more information about the syntax of the command, see Chapter 2 of *DB2 Command Reference*.

Increase IRLM storage for Sysplex query parallelism

Sysplex query parallelism uses IRLM Notify messages for the following functions:

- The coordinator uses them to communicate to the assistants.
- The assistants use Notify messages to communicate with the parallelism coordinator.
- The main use of these messages is to pass data between the assistants and the coordinator.

To use Sysplex query parallelism, make sure that you have enough ECSA to handle these messages.

Calculating storage for the assistants: To estimate the amount of extra storage required by IRLM on an assisting DB2 (ANY DB2 that receives query work from another DB2), estimate the number of parallel tasks that can run concurrently on the assisting DB2, and divide that by the number of members sharing the query work. Multiply the result by 32KB to get an estimate of the amount of extra storage needed on the assistants.

```
(numbers of queries × max concurrent tasks) / number of members) × 32KB
```

For example, assume you have a data sharing group in which all four members participate in processing parallel queries. If you have at most 10 queries executing concurrently, and the highest number of parallel tasks is approximately 40, the calculation is:

```
(10 × 40) / 4 = 100
100 × 32KB = 3MB of extra storage on assistant
```

To estimate the number of parallel tasks, you can look at EXPLAIN output or instrumentation records for the concurrent queries.

Calculating storage for the coordinator: Any member that can be a coordinator needs approximately 200 KB extra storage for messages that the coordinator sends to the assistants.

Storage for DB2 objects

In general, you can use the storage estimates in Section 2 of *DB2 Installation Guide* for your capacity planning. This section describes some additional information that is specific to data sharing.

Estimating storage for the EDM pool

The formula you use to calculate storage for the environmental descriptor manager (EDM) pool is described in Section 2 of *DB2 Installation Guide* . For data sharing, you might need to increase that storage estimate by about 10 percent because of the way DB2 cross-invalidates database descriptors (DBDs). This percentage is just an estimate; the actual amount of increase depends on how much you create, drop and alter objects in the data sharing group.

Cross-invalidating items in the EDM pool: DB2 does not have a backing EDM pool in the coupling facility for invalidating objects in the EDM pool (DBDs, cursor tables, and so on) because these objects are modified less frequently than database data. So, there is one EDM pool for each DB2 subsystem. When a DBD changes, perhaps because an object descriptor changed, DB2 uses XCF messages to notify other DB2s using that DBD that new transactions should not use that copy of the DBD. New transactions use the new DBD (which is read into the EDM pool). Thus, it is possible that one transaction is using the new DBD while the old one is still being used by the currently running transactions. In other words, more than one copy of a DBD can exist in the EDM pool.

Reducing the storage impact: For CREATE, ALTER, or DROP statements, the DBD is not modified until a COMMIT is issued. You can significantly reduce the number of EDM versions by issuing all those SQL statements within a single COMMIT scope. However, the exclusive lock on the DBD is held until the COMMIT.

Storage for reusing threads: One of the general recommendations for data sharing is to reuse threads whenever possible and to bind with the option RELEASE(DEALLOCATE). Depending on how often your threads get reused, this bind option can mean more storage is necessary for storing objects used by the plan. Plan for more EDM pool storage if you use RELEASE(DEALLOCATE) and if you reuse threads.

Recommendation: To achieve a good balance between storage and processor usage, use RELEASE(DEALLOCATE) for frequently used plans or packages. Use RELEASE(COMMIT) for those plans or packages that are infrequently used.

Planning to enable data sharing

Moving to data sharing is a two-step process, in most cases. First, migrate to Version 6 (or install Version 6), and then “enable” that Version 6 subsystem for data sharing. This process is described more fully in “Enabling DB2 data sharing” on page 87. (If you already have a data sharing group, see “Migrating an existing data sharing group to the new release” on page 96 for information about migrating that group.)

After data sharing is enabled, disabling data sharing is a very difficult process and one which should be avoided. See “Disabling and re-enabling data sharing” on page 107 for more information.

This section describes things you need to be aware of and to prepare for as you plan your move to data sharing:

- “Decide if merging is the right thing to do”
- “Connecting IMS and CICS” on page 68
- “Binding plans and packages if moving to a new machine” on page 68
- “Registering command prefixes and group attachment name” on page 68
- “Applications using CICSPlex SM” on page 70
- “Increasing the size of the BSDS” on page 73
- “Increasing the size of the SYSLGRNX table space” on page 74
- “Additional considerations for existing subsystem parameters” on page 74

See also “Tuning deadlock and timeout processing” on page 210 for information about tuning your timeout periods.

Decide if merging is the right thing to do

Although DB2 does not offer an automated way to “merge” catalogs, it is possible to merge existing DB2s into a data sharing group. Consider carefully a decision to merge existing DB2 subsystems.

Merging is a very complicated process. It involves not just the physical problem of moving data, it includes many other management issues, such as the following:

- Naming conventions for users, plans, packages, databases, tables, views, and so on
- Authorization techniques
- Backup and recovery conventions
- Availability practices

Before you consider merging existing DB2 subsystems into a single data sharing group, ask yourself the following question: **Why are the subsystems separate now?**

Reasons to not merge

If the subsystems are separated now because of a different set of user groups that do not need frequent access to each other's data, do not consider merging them.

For the same reasons that you most likely do not include test and production DB2s in a single MVS, we do not recommend merging test and production subsystems into a single data sharing group.

If you are trying to minimize the number of subsystems because you are running short of subsystem recognition characters, then using 8-character command prefixes offers relief.

If you have two existing subsystems, and each of those subsystems can grow into a separate group, availability is usually better by keeping those groups separate. Administration is simpler by keeping the groups split along the same lines as the users.

Reasons to consider merging

If the subsystems were split because of capacity constraints only, then merging might be a good idea, especially if the subsystems already share a common naming convention.

If the subsystems have or need a lot of common data and are using shared read only data, distributed access, or data replication to handle the problem of sharing the data, merging might be a solution. However, this might not be a good approach if the security needs of the two groups are very different. If you try to merge two subsystems with very different security needs, especially if a shared naming convention is not already in place for those separate subsystems, then merging them could be very difficult.

Connecting IMS and CICS

You must define IMS and CICS connections for each DB2 in the data sharing group. See Section 2 of *DB2 Installation Guide* for more information about connecting CICS and IMS to DB2.

Connecting CICS to DB2

The CICS-DB2 attachment facility lets you override the subsystem name on startup and with the INITPARM, eliminating the need for a second instance of the RCT if you wish to connect to a different DB2 subsystem.

Connecting IMS to DB2

For every member that is going to run IMS applications, make sure you attach IMS to that member. IMS must include a separate SSM member for every member DB2. See Section 2 of *DB2 Installation Guide* for more information about connecting DB2 to IMS.

Binding plans and packages if moving to a new machine

Running a plan or package on a data sharing group does not in itself require that you rebind that plan or package. However, if you are using a new machine that has different performance characteristics (from a 9021 711-based processor to the S/390 microprocessor cluster, for example), it is to your advantage to rebind plans and packages on the machine they will be running on. See “Access path selection in a data sharing group” on page 258 for more information.

Registering command prefixes and group attachment name

Use the IEFSSN*xx* parmlib member to register the 1- to 8-character command prefix for a member and the group attachment name for the group, as shown here:

```
ssname,DSN3INI,'DSN3EPX,prefix<,scope<,group-attach>>'
```

When you register the command prefix in the IEFSSN*xx* parmlib member, you also specify the *scope* of the prefix. We recommend that you choose a scope of Started (S), which allows you to have a single IEFSSN*xx* parmlib member to be used by all MVS systems in the Sysplex. It also simplifies the task of moving a DB2 from one system to another; you can stop DB2 on one MVS and start it up on another. There is no need to re-IPL the system.

For more information about information in the IEFSSN*xx* parmlib member, see information about job DSNTIJMV in Section 2 of *DB2 Installation Guide*.

Sample definitions

Here are sample definitions that might appear in the shared SYS1.PARMLIB:

```
DB1G,DSN3INI,'DSN3EPX,-DB1G,S,DB0G'
DB2G,DSN3INI,'DSN3EPX,-DB2G,S,DB0G'
DB3G,DSN3INI,'DSN3EPX,-DB3G,S,DB0G'
DB4G,DSN3INI,'DSN3EPX,-DB4G,S,DB0G'
```

With these definitions, you can start DB1G on MVS1 and that DB2 will be the only one in the Sysplex that can use -DB1G as its command prefix. However, because the DB2 is registered to MVS with a scope of S, it is possible to stop DB1G and restart it on another MVS without having to re-IPL any MVS system.

Changing the command prefix

To change the command prefix parameters, you must change the IEFSSN*xx* entry and re-IPL the host system. For example, if you want to change the command prefix scope from system to Sysplex-wide, and you want to register the prefix at DB2 startup, change the M in the entry to S before re-IPLing.

If you want to use multiple-character command prefixes, make sure your automation programs can handle multiple-character prefixes in messages before making that change.

Group attachment name

As shown in “Registering command prefixes and group attachment name” on page 68, specify the group attachment name in the IEFSSN*xx*member. You can let the DB2 installation process do this for you, or you can update the member yourself. We recommend that you specify the group attachment name at a convenient time, such as during a planned IPL.

Even if you have not yet enabled data sharing, the group attachment name is active after you IPL the system. This is not a problem. Until you are ready to move to data sharing, we recommend that you continue specifying the DB2 subsystem name in your TSO and batch jobs. When ready to move to data sharing, you can then change those jobs to specify a group attachment name without the need for an IPL.

Another option is to keep the group attachment name the same as the originating member's subsystem ID. By doing this, you can avoid changing the name in your jobs when you move to data sharing.

How DB2 chooses a subsystem name: When you submit a job on an MVS system, DB2:

1. Assumes that the name you specified on the DSN command is a subsystem name and attaches to that subsystem if it is started.
2. If *either* of the following is true:
   - The name on the DSN command is not defined as a subsystem
   - A subsystem by that name is not started and that subsystem's group attachment name is the same as the subsystem name

   then DB2 checks to see if the name is a group attachment name.

   a. If the name on the DSN command is a group attachment name, it constructs a list of DB2 subsystems that are defined to this MVS, and tries to attach to each one in order, until it finds a DB2 subsystem that is started on this MVS, or until it reaches the end of the list. DB2 always attaches to the first started one on the list—there is no load balancing.

To create the list, DB2 adds each subsystem when it goes through subsystem initialization. At IPL time, subsystems are initialized in the order in which they appear in the IEFSSN*xx* member. If you add a subsystem with the MVS SETSSI command, that subsystem is added to the list at that time.

b. If the name on the DSN command is not a group attachment name, then a “not started” message is returned.

Be careful when you begin moving to data sharing that your IEFSSN*xx* definitions are correct. This can be especially troublesome if you have extinct subsystems that are still defined but not used. For example, assume you have the following subsystem definitions on an MVS system. Notice that DB2P does not have the group attachment name specified in its definition:

```
DB2P,DSN3INI,'DSN3EPX,-DB2P,S'  ⟵Extinct subsystem
DB1G,DSN3INI,'DSN3EPX,-DB1G,S,DB2P'  ←Active subsystem
```

Because the jobs submitted on this MVS are trying to connect to the name DB2P, DB2 always tries to connect to subsystem DB2P. In most cases, DB2 doesn't use the group attachment name support, and a “not started” message is returned. To avoid this situation, include the group attachment name in DB2P's definition. If the correct group attachment name is present, DB2 does try to attach to DB1G after it discovers that DB2P is not started.

Applications using CICSPlex SM

CICSPlex® System Manager/ESA (CICSPlex SM) is a system-management tool that enables you to manage several CICS systems as if they were one. The dynamic transaction routing program supplied with CICSPlex SM balances the enterprise workload dynamically across the available application owning regions (AORs). CICSPlex SM enables you to manage a variable workload without operator intervention, while maintaining consistent response times. It can do this because it routes transactions away from busy regions and from those that are failing or likely to fail, giving improved throughput and concealing problems from end users.

In some situations, your DB2 applications must be sensitive to a “resource unavailable” condition. For example, assume a database is stopped for planned maintenance, and the application receives a -904 SQLCODE and ends normally. CICSPlex SM might continue to route work to that system because it appears to complete its work rapidly. This is sometimes called the “stormdrain” effect.

Another cause for stormdrain: When both of following conditions are true, the stormdrain effect can occur:

- The CICS attachment facility is down
- You are using INQUIRE EXITPROGRAM to avoid AEY9 abends

Again, because there hasn't been an abend, it appears as if work is completing rapidly at that subsystem.

Writing a CICS exit to avoid the stormdrain effect

The information under this heading, up to “Increasing the size of the BSDS” on page 73, is Product-sensitive Programming Interface and Associated Guidance Information about Customer Information Control System/Enterprise Systems Architecture (CICS/ESA) and DB2.

You can write a resource manager interface program exit, XRMIOUT, to avoid the stormdrain effect caused by a resource unavailable SQLCODE (-904). (This exit does not avoid the stormdrain problem caused by using INQUIRE EXITPROGRAM to avoid AEY9 abends.)

Using XRMIOUT, you can intercept the return from the resource manager. The exit can check whether:

- The resource manager is DB2.
- There is a -904 SQLCODE in the SQL communication area (SQLCA).

If these conditions exist, abend the transaction, instead of ending the transaction normally.

To determine if DB2 is the resource manager, compare `'DSNCSQL'` with the value stored at the address included with the UEPTRUEN parameter passed to XRMIOUT as shown in Figure 17.

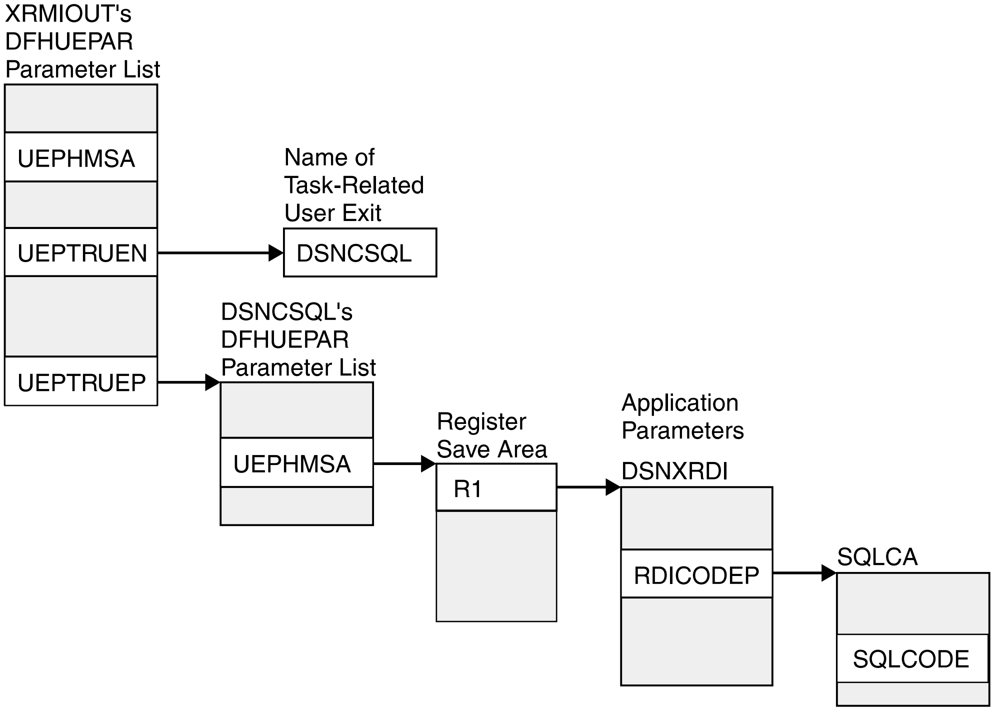


Figure 17. Pointers to resource manager name and SQLCODE

To find the SQLCODE:

1. Find UEPTRUEP in the DFHUEPAR parameter list passed to XRMIOUT. UEPTRUEP contains the address of the DFHUEPAR parameter list passed to DSNCSQL.
2. Find UEPHMSA in DSNCSQL's DFHUEPAR parameter list. UEPHMSA points to the register save area containing the application's registers.

3. Find register 1 in the register save area. Register 1 contains the address of the application parameters.

   The application parameters passed by the precompiler to DB2 are mapped by the DSNXRDI mapping macro. The mapping macro is contained in the data set library *prefix*.SDSNMACS.

4. Find RDICODEP in the DSNXRDI structure. RDICODEP contains the address of the SQL communication area (SQLCA).
5. Find the SQLCODE. The SQLCODE offset is documented in Appendix C of *DB2 SQL Reference*.

For more information about the XRMIOUT exit, see *CICS for MVS/ESA Customization Guide*.

CICS enhancement to avoid the stormdrain effect

The CICS Transaction Server for OS/390 Release 1, and subsequent releases, lets you avoid the stormdrain effect. With that release, you will not need to use XRMIOUT to check for resource unavailable conditions.

That follow-on release also gives you new options on the RCT TYPE=INIT macro that allow you to get the benefits of the INQUIRE EXITPROGRAM without the disadvantage of causing the stormdrain effect. Those options are STRTWT=AUTO and STANDBY=SQLCODE. For more information about these options, see Section 2 of *DB2 Installation Guide*.

Migrating transactions that have ordered dependencies

This section pertains only for those limited cases in which one transaction, called an *originating* transaction, updates DB2 data using INSERT, UPDATE, or DELETE, and then, *before completing phase 2 of commit*, spawns a second transaction that is dependent on the updates that were made by the first transaction. This type of relationship is referred to as “ordered dependencies” between transactions.

Statement of the problem

In some situations, the dependent transaction can encounter a “row not found” condition that did not occur in a non-data-sharing environment. The reason that the dependent transaction might start to encounter periodic “row not found” conditions after enabling data sharing is that in a data sharing environment the dependent transaction might run on a different DB2 member than the DB2 member on which the originating transaction is running, and because of the multi-system buffering effects that are present in data sharing (each DB2 member has its own local buffer pools), the uncomitted buffered pages that are updated by the originating transaction are not immediately “visible” to the dependent transaction when it runs on a different DB2 member. Even in a non-data-sharing environment, the dependent transaction would need to tolerate a “row not found” condition in cases where the originating transaction rolled back.

When the problem might occur

If all of the following conditions are true, it is possible that the dependent transaction might periodically encounter a “row not found” condition when attempting to access a row that was updated by the originating transaction:

- The originating transaction spawns the dependent transaction before its phase 2 of commit completes, and

- The dependent transaction runs on a different member than the originating transaction, and
- The dependent transaction is not bound with an isolation level of repeatable read, and
- The timing is such that the dependent transaction attempts to read the updated row before the originating transaction has completed phase 2 of commit.

Preventing the problem

To prevent these periodic “row not found” situations, consider using the IMMEDWRITE(PH1) or IMMEDWRI(YES) option of BIND/REBIND for a plan or package that spawns dependent transactions that might run on other members, or the IMMEDWRI subsystem parameter. DB2 immediately writes to the coupling facility any updated buffers that contain pages that belong to page sets or partitions that are dependent on group buffer pools. IMMEDWRITE(PH1) means that DB2 writes the page to the group buffer pool at or before phase 1 commit. IMMEDWRITE(YES) means that DB2 writes the page as soon as the buffer update completes. DB2 writes the data to one of the following structures or devices:

- The group buffer pool
- DASD for GBPCACHE NO group buffer pools or GBPCACHE NONE or GBPCACHE SYSTEM page sets

Subsystem parameter IMMEDWRI can override the value of the IMMEDWRITE bind option with which a plan or package is executed on a data sharing member. For information on which values of the subsystem parameter cause the bind option to be overridden, see “Other recommendations” on page 79.

The following are other alternatives to solving the order-dependent transaction problem:

- Ensure that the originating transaction does not schedule the dependent transaction until the originating transaction has completed phase 2 of commit.
- Run the dependent transaction with an isolation level of repeatable read.
- If the dependent transaction is currently running with an isolation level of cursor stability CURRENTDATA(NO), changing to use CURRENTDATA(YES) can sometimes solve the problem.
- Add statement retry logic to handle the return of a “row not found” condition.
- Run the dependent transaction on the same member as the originating transaction.

Increasing the size of the BSDS

Data sharing causes additional records to be written to the BSDS for member information. To avoid having the BSDS go into secondary extents, we recommend that you change the size of the primary space allocation to 180 records. This is necessary only for subsystems that have followed the IBM-recommended migration path from Version 3 to Version 6 without altering that size. New installations and member installations already do this for you.

To increase the space allocation for the BSDS, you must:

1. Rename existing BSDSs.
2. Define larger BSDSs with the original names.

3. Copy the renamed ones into the new ones.

All this can be done using access method services. To see the definition used for the BSDSs, see the installation job DSNTIJIN.

Increasing the size of the SYSLGRNX table space

The SYSLGRNX directory table space consists of the RBA ranges of when data sets are open for updating. Because there are more members opening and closing data sets, this table space is likely to grow with the addition of each new member to the data sharing group. And, if you choose to copy indexes, this table space can grow even more.

Consider increasing the frequency with which you remove rows from this table space, or increasing the size. To see the definition used for SYSLGRNX, see installation job DSNTIJIN.

To increase the space allocation for SYSLGRNX, you must use access method services:

1. Rename the existing SYSLGRNX data set.
2. Define a larger SYSLGRNX data set with the original name.
3. Copy the contents of the renamed data set into the new SYSLGRNX data set.

Additional considerations for existing subsystem parameters

If you are moving your DB2 processing to S/390 microprocessors, you might need to change some subsystem parameters:

- System checkpoint

  When you add a second member that runs on a smaller S/390 microprocessor to a data sharing group, consider tailoring the checkpoint frequency. You can reduce the number of log records written by the member running on the smaller machine so the checkpoint frequency is about the same as the member running on the larger machine. You do this by lowering the value on the CHECKPOINT FREQ field of installation panel DSNTIPN.

  Frequent checkpointing reduces DB2 restart time and thus reduces the amount of time data might be locked out from other DB2 members. It also reduces the amount of time it takes to recover a group buffer pool if it should fail.

  Be sure to consider also the impact of frequent checkpoints on how often DB2 changes data set status from R/W to R/O state. See “How DB2 Tracks Interest” on page 223 for more information.

- EDM pool size

  See “Estimating storage for the EDM pool” on page 66 for information about possibly increasing this value.

Chapter 4. Installing and enabling DB2 data sharing

The purpose of this chapter is to give you an overview of the process of making a data sharing group. Table 10 points you to the procedures you need to create or migrate to a data sharing group. The complete set of installation panels and steps are shown in *DB2 Installation Guide* .

Be sure to read “Choosing subsystem parameters” on page 76 for guidance on choosing specific subsystem parameters.

Table 10. Data sharing options

If you have this...	And you want this...	Read this...
No system	Version 6, data sharing	“Installing a new DB2 data sharing group” on page 84. (Use this procedure only in low-risk situations. It is best to migrate to Version 6 and then enable data sharing.)
A Version 6 non-sharing subsystem	The originating member of a data sharing group	“Enabling DB2 data sharing” on page 87.
One member in the group	More members in the group.	“Adding a new DB2 data sharing member” on page 89.
Separate DB2 subsystems	Merged DB2 subsystems into a single group.	“Merging existing DB2 data into the group” on page 90.
A Version 5 data sharing group	A Version 6 data sharing group.	“Migrating an existing data sharing group to the new release” on page 96.

The following tasks are also described here:

- “Renaming the DB2 subsystem” on page 85
- “Merging existing DB2 data into the group” on page 90
- “Testing the data sharing group” on page 94
- “Updating subsystem parameters for a member” on page 96

If you already have a Version 5 data sharing group, read this chapter for new information, and see “Migrating an existing data sharing group to the new release” on page 96.

For information about falling back, see “Falling Back and Remigrating” on page 105.

For information about disabling data sharing, which is not a recommended course of action, see “Disabling and re-enabling data sharing” on page 107.

And, in the event that you want to remove a data sharing member, either permanently or temporarily, see “Removing members from the data sharing group” on page 111.

Choosing subsystem parameters

Every member of a DB2 data sharing group must have its own unique load module for subsystem parameters (sometimes called DSNZPARM in a non-data-sharing environment).

The load module for subsystem parameters is built by job DSNTIJUZ and stored in the *prefix*.SDSNEXIT target library. Every member must use a different name for its subsystem parameters load module because the *prefix*.SDSNEXIT target library can be shared among all members of the data sharing group. The installation process requires that you provide the name of the load module for a member.

Recommendation: Name each member's load module using the convention DSNZP*xxx*, where *xxx* includes the number in the member name and the group identifier. For example, DB1G's subsystem parameters load module could be named DSNZP01G.

The subsystem parameters load module name for a member is an optional parameter on the EXEC statement in the JCL procedure used to start the *ssnm*MSTR address space. This is provided so that an operator (or automated operations) does not have to specify the subsystem parameter load module name when starting a DB2 member. The format for specifying the parameter on the EXEC statement of the *ssnm*MSTR JCL procedure is:

```
  //IEFPROC EXEC PGM=DSNYASCP,PARM='ZPARM(DSNZPxxx)',...
```

The scope and uniqueness of DB2 subsystem parameters

Even though the various subsystem parameters affect the operation of only a single DB2, some parameters must be the same on all the sharing subsystems. An example is the catalog alias name.

Other subsystem parameters must be unique for each member. An example of this is the name of the BSDS used by each DB2 subsystem.

Most subsystem parameters do not have to be unique. We offer recommendations for some of these parameters. In the tables in the following section, the parameter is indicated by the installation panel field name. However, some of the parameters do not reside in the DSNZP*xxx* load module, and cannot be changed through the installation or migration process.

Subsystem parameters that must be different on each DB2

These parameters must be different on every data sharing DB2 member in a group. These parameters must be specified when a member is installed.

Table 11 (Page 1 of 2). Subsystem parameters that must be different on each DB2

Subsystem parameter field name	Panel ID	Comment
Active Logs: COPY 1 PREFIX	DSNTIPH	Each subsystem writes to its own recovery log.

Table 11 (Page 2 of 2). Subsystem parameters that must be different on each DB2

Subsystem parameter field name	Panel ID	Comment
Active Logs: COPY 2 PREFIX	DSNTIPH	Each subsystem writes to its own recovery log.
Archive Logs: COPY 1 PREFIX	DSNTIPH	Each subsystem writes to its own recovery log.
Archive Logs: COPY 2 PREFIX	DSNTIPH	Each subsystem writes to its own recovery log.
Bootstrap Data Sets (BSDS): COPY 1 NAME	DSNTIPH	Each DB2 has its own BSDS.
Bootstrap Data Sets (BSDS): COPY 2 NAME	DSNTIPH	Each DB2 has its own BSDS.
COMMAND PREFIX	DSNTIPM	The command prefix used to route commands to this subsystem.
DB2 NETWORK LUNAME	DSNTIPR	Even if you do not use distributed database, this identifier is required to ensure that logical unit of work IDs (LUWIDs) are unique across the data sharing group.
DB2 PROC NAME	DSNTIPX	JCL procedure for the DB2-established stored procedures address space.
MEMBER IDENTIFIER	DSNTIPJ	The unique identifier for this IRLM.
MEMBER NAME	DSNTIPK	The member name for this DB2.
PARAMETER MODULE	DSNTIPO	The name of the subsystem parameter load module for this DB2.
PROCNAME	DSNTIPI	This is the name of the IRLM procedure that MVS invokes if IRLMAUT=YES.
RESYNC PORT	DSNTIP5	When using TCP/IP network protocols, this is the port used for resynchronization of two-phase commit processes.
SUBSYSTEM NAME	DSNTIPM	The DB2 subsystem identifier.
SUBSYSTEM SEQUENCE	DSNTIPM	The DB2 subsystem sequence number.
SUBSYSTEM NAME (IRLM)	DSNTIPI	Specifies the name of the IRLM subsystem associated with a particular DB2. This name must be unique within the Sysplex.
WORK FILE DB	DSNTIPK	The name of the work file database for this DB2.

Subsystem parameters that must be the same on every DB2

The following subsystem parameters must be the same for every member of the data sharing group.

Table 12 (Page 1 of 2). Subsystem parameters that must be the same

Subsystem parameter field name	Panel ID	Comment
CATALOG ALIAS	DSNTIPA2	Specifies the DB2 catalog alias name.
DATABASE PROTOCOL	DSNTIP5	Specifies the default protocol to use when an application program contains three-part names.
DB2 LOCATION NAME	DSNTIPR	Specifies the location name for the entire DB2 data sharing group. This name is required.
DRDA PORT	DSNTIP5	Specifies the DRDA port number for the entire DB2 data sharing group. This name is required if using TCP/IP network connections.
EXTENDED SECURITY	DSNTIPR	Determines the content of error codes returned to a client when a connection request fails because of security errors. Also allows users to change their passwords if their host passwords expire.
GROUP ATTACH	DSNTIPK	This is the group attachment name that allows TSO and batch programs, programs using CAF and RRSAF, and utilities to generically attach to any DB2 member of the group. The default is subsystem name of the first installed member of the group.
GROUP NAME	DSNTIPK	The name of the DB2 group.
IRLM XCF GROUP NAME	DSNTIPJ	The name of the IRLM group.
INSTALL DD CONTROL SUPT. (and related parameters)	DSNTIPZ	All members of the data sharing group must use the same set of data definition control registration tables, or unpredictable results can occur.
MINIMUM DIVIDE SCALE	DSNTIPF	Specifies whether to retain at least three digits to the right of the decimal point after any decimal division.

Table 12 (Page 2 of 2). Subsystem parameters that must be the same

Subsystem parameter field name	Panel ID	Comment
SITE TYPE	DSNTIPO	All members of the group must have the same value. All members on the remote site, after they need to become the local site, must have the same value (LOCALSITE).
TRACKER SITE	DSNTIPO	All members of the group must have the same value.
SYSTEM ADMIN 1 SYSTEM ADMIN 2 SYSTEM OPERATOR 1 SYSTEM OPERATOR 2	DSNTIPP	Specifies installation SYSADM and SYSOPR authorities.
TCP/IP ALREADY VERIFIED	DSNTIP5	Specifies whether incoming TCP/IP requests are accepted by DB2 without a password or RACF PassTicket. This option must be the same on all members or requesters will have inconsistent results.

Other recommendations

These parameters can be the same or different on separate members of the group.

Table 13 (Page 1 of 3). Recommended subsystem parameters

Subsystem parameter field name	Panel ID	Comment
DEALLOC PERIOD	DSNTIPA	This is the length of time during which an archive read tape unit is allowed to remain unused before it is deallocated. We don't recommend archiving to tape. If you *must*, however, we recommend that you specify 0 for this parameter unless you intend to run all RECOVER jobs from the same DB2 member. Specifying a deallocation delay means that the tape is not available to any other DB2 members until the deallocation time expires.
DEFAULT BUFFER POOL FOR USER DATA	DSNTIP1	Specifies the default buffer pool for any table spaces that are created without a specified default. For consistent results when creating objects on different members, make this default the same on all members.

Table 13 (Page 2 of 3). Recommended subsystem parameters

Subsystem parameter field name	Panel ID	Comment
DEFAULT BUFFER POOL FOR USER INDEXES	DSNTIP1	Specifies the default buffer pool for any indexes that are created without a specified default. For consistent results when creating objects on different members, make this default the same on all members.
DEVICE TYPE 1 DEVICE TYPE 2	DSNTIPA	This is the device type or unit name for storing archive logs. The recommendation is to archive the first copy of the log to DASD.
EDMPOOL STORAGE SIZE	DSNTIPC	This is the size of the environmental descriptor manager (EDM) pool in kilobytes. Whatever value is calculated from the installation CLIST, consider adding more storage because of the way data sharing updates database descriptors in the EDM pool.
READ COPY2 ARCHIVE	DSNTIPO	Specifies that the second copy of the archive log should be read first for restart and recovery. We recommend that this option be the same (either YES or NO) on all members of the group. Otherwise, the parameter as it is set on the member that owns the archive log data set determines which copy is used.
READ TAPE UNITS	DSNTIPA	This is the maximum number of dedicated tape units that can be allocated to read archive log tape volumes concurrently. We don't recommend archiving to tape. If you *must*, however, it is vital that you have enough tape units allocated to the DB2 doing the recovery to merge the archive logs from *all* members in the group that have updated the object being recovered. Thus, if there are 8 members in the group, make sure you specify at least 8 on this panel for each member. See “The impact of archiving logs in a data sharing group” on page 146 for more information about archiving to tape.

Table 13 (Page 3 of 3). Recommended subsystem parameters

Subsystem parameter field name	Panel ID	Comment
RECORDING MAX	DSNTIPA	Specifies the maximum number of archive logs to be recorded in the BSDS. We recommend that all members in a DB2 data sharing group use the maximum value of 1000. This makes it easier to transfer a workload from one member to another.
RETAINED LOCK TIMEOUT	DSNTIPI	Specifies a multiplier to apply to the timeout value for a connection when that connection is waiting for an incompatible lock held by another failed DB2 member. Locks held by failed DB2 members are called *retained* locks. ***Recommendation:*** If you have automatic restart, or some other restart automation that quickly restarts failed DB2s, choose a non-zero value for this field.
START IRLM CTRACE	DSNTIPI (or OR TRACE=YES on IRLM startup procedure)	Specifies that diagnostic traces be activated when IRLM is started. We recommend that this be activated for all members of the group, because the negligible performance overhead is outweighed by the benefit of easier problem resolution. Only those systems that have reached the limits of processor capacity should consider not automatically activating these traces.

An additional subsystem parameter that is not reflected on the installation panels that can be the same or different is IMMEDWRI, which is in macro DSN6GRP. This subsystem parameter determines the IMMEDWRITE bind option used for plans and packages that are run on the data sharing member:

NO
The IMMEDWRITE bind option that is specified for the plan or package is used. The default is NO.

PH1
The IMMEDWRITE(PH1) option is used for all plans and packages except those bound with IMMEDWRITE(YES). IMMEDWRITE(YES) is used for those plans and packages bound with that option.

YES
The IMMEDWRITE(YES) option is used for all plans and packages. This option might have some performance impact.

The IMMEDWRITE bind option controls when DB2 writes updated group buffer pool dependent pages to the coupling facility (or to DASD for GBPCACHE NONE|SYSTEM objects). DB2 can either write the GBP-dependent pages immediately, at or before phase 1 commit, or at or before phase 2 commit. For more information about the bind option, see *DB2 Command Reference*. For information on using the bind option to avoid “row not found” conditions for transactions that have ordered dependencies, see “Migrating transactions that have ordered dependencies” on page 72.

DSNHDECP parameters

There is a single DSNHDECP load module for each release of DB2 in a data sharing group. The application programming defaults contained within the DSNHDECP load module are considered global for the group. The load module is created during the installation of the DB2 group. It cannot be modified during the installation of a DB2 member.

The DSNHDECP parameter DECPSSID has a special meaning in a data sharing group. DECPSSID contains the group attachment subsystem name for the data sharing group This allows utilities, TSO attachment, RRSAF, and CAF applications to attach to any DB2 subsystem in the group.

Creating the DB2 data sharing group

Before enabling data sharing, read the following topics:

- “Recommended approach for moving to data sharing”
- “Sharing DB2 libraries” on page 83
- “Ensuring that installation jobs access the correct JCL procedures” on page 83
- “Establishing system affinity for installation jobs” on page 84

Recommended approach for moving to data sharing

Moving to data sharing is a big step. Plan this move carefully, because after you enable data sharing, it is very difficult to disable it. Disabling should only be considered if your long-term plans are to disable data sharing.

Before enabling data sharing, test other major new functions in the release on a single system, and then build and try a test data sharing group. When you are ready to begin a move to production, you must avoid having to fall back to the previous release.

Build and try a test data sharing group

Here is one approach to testing your data sharing group:

1. Convert indexes to type 2, if they are not converted already.

   You must convert all indexes to type 2 before migrating to or installing Version 6.

2. Install Version 6 as a single system, and test it with dummy data, or copies of production data, in order to test some of the new functions. Run old applications and begin new application development work.

   In the meantime, prepare the hardware and define the coupling facility structures to enable data sharing.

3. Enable data sharing on this test system.

   You will have a single-system data sharing group at this point, and this can help you find any initial configuration problems. Make sure old applications work in this environment.

4. Install additional members in the test group.

   Run applications from different subsystems in the group to fully exercise the group buffer pools and cross-system locking.

Move to production

When you are ready to move to production:

1. Migrate your existing Version 5 subsystem to Version 6, but don't use any of the new functions yet.
2. Start using new functions when you are sure the release is stable and you won't need to fall back.
3. Tune applications to contain the level of locking and lock contention rates.

   To reduce the effects of locking contention in a data sharing environment, it is best to first control locking costs in a non-data-sharing environment. This will give you a baseline from which to do further tuning after the move to data sharing. See Section 5 (Volume 2) of *DB2 Administration Guide* for information about reducing locking contention. See “Improving concurrency” on page 203 for information about reducing contention in a data sharing environment.

4. Enable data sharing on this originating subsystem, and run applications on this one-member data sharing group.
5. Install additional members as needed.

Sharing DB2 libraries

DB2 target and distribution library data sets can be shared among all members of a DB2 data sharing group. There is no need for each member of the group to have its own set of target and distribution libraries. Sharing these libraries reduces the effort to install and maintain the different members of the data sharing group. As described in “Administering a database” on page 26, sharing libraries can also help ensure that members are using the same exit routines.

Sharing libraries also simplifies the tasks of defining a new DB2 member to a data sharing group. The DB2 member installation process supports sharing of the libraries.

Ensuring that installation jobs access the correct JCL procedures

If you have more than one procedure library, ensure that your installation jobs access the right set of procedures by using a JCLLIB statement to specify the order in which procedure libraries are searched.

The JCLLIB statement looks like this:

```
//ddname JCLLIB ORDER=(library[,library...])
```

The JCLLIB statement must follow the JOB statement and precede the first EXEC statement in the job. You can have DB2 insert this statement in your JCL for you by entering it on installation panel DSNTIPY.

For more information on the JCLLIB statement, see *OS/390 MVS JCL Reference*.

Establishing system affinity for installation jobs

You must ensure that the installation jobs are run on the MVS system on which the appropriate DB2 subsystem is running. There are several MVS installation-specific ways to make sure this happens. These include:

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

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

  Where *cccc* is the JES2 name. You can specify an asterisk (SYSAFF=*) to indicate that the job should run on the system from which it was submitted.

- For JES3 systems, use the following JCL statement:

  ```
  //*MAIN SYSTEM=(main-name)
  ```

  Where *main-name* is the JES3 name.

OS/390 MVS JCL Reference describes the JCL statements shown above. You can edit the jobs manually, or you can enter the above statements on installation panel DSNTIPY and have DB2 insert these statements for you.

Your installation might have other mechanisms for controlling where batch jobs run, such as by using job classes.

To create a data sharing group, you add one member at a time. The first member (the *originating* member) can either be created as a new installation or enabled for data sharing from an existing Version 6 subsystem. **We strongly recommend that you use Version 6 before enabling data sharing.** This allows you to test Version 6 without the additional complexity of defining and tuning coupling facility structures. In addition, this approach helps you avoid falling back after data sharing is enabled. See “Migrating an existing data sharing group to the new release” on page 96 for information to help you plan a move to a data sharing environment.

Installing a new DB2 data sharing group

Use this procedure only in low-risk situations. The recommended approach is to migrate to or install a Version 6 subsystem, use it for a while, and then *enable* data sharing as described in “Enabling DB2 data sharing” on page 87.

However, if you decide to install and immediately enable data sharing on a new Version 6 subsystem, this new Version 6 subsystem becomes the originating member of the data sharing group. This member's DB2 catalog is used as the DB2 catalog for the data sharing group.

To install the new data sharing group:

1. On panel DSNTIPA1, specify:

   ```
   INSTALL TYPE ===> INSTALL
   DATA SHARING ===> YES
   ```

2. On panel DSNTIPP1, specify 1 to indicate the Group data sharing function.

3. On panel DSNTIPK, specify:

   ```
   GROUP NAME           ===> group name
   MEMBER NAME          ===> originating member name
   ```

   Verify that the *originating member name* is unique within your MVS Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the group name and member name you specify here.

4. Complete the installation panels, specifying parameters according to the guidelines in “The scope and uniqueness of DB2 subsystem parameters” on page 76.

5. Complete all installation steps, as described in Section 2 of *DB2 Installation Guide*.

6. Run the installation verification procedures (IVP), as described in Section 2 of *DB2 Installation Guide*.

Renaming the DB2 subsystem

If you want to rename your DB2 subsystem, it is best to do it before enabling data sharing. However, another convenient time to do this is during the process of enabling data sharing. Because DB2 must be shut down during the enabling process, you can perform the tasks necessary to rename the subsystem at the same time. This section is divided into two parts:

- “Tasks that require an IPL”
- “Tasks at enable time” on page 86

Renaming a subsystem is an activity that you should plan for very carefully. Because every installation has a different configuration, we cannot guarantee that this procedure is complete. If you are interested in changing the high level qualifier for data sets, see the procedure in Section 2 (Volume 1) of *DB2 Administration Guide*.

In this example procedure, we change the subsystem name for DB2P to DB1G to conform to the naming convention for data sharing that we use in this publication.

Tasks that require an IPL

If you choose to do any of the following tasks, you must IPL MVS to pick up the changes:

- Modify the IEFSSN*xx* member for this subsystem to include the group attachment name. To avoid having to modify JCL for all your jobs, use the existing subsystem name as the group attachment name. This step is necessary to ensure that there are no problems when the call attachment and TSO attachment facilities try to use the group attachment name.

  For example, if the existing IEFSSN*xx* member looks like this:

  ```
  DB2P,DSN3INI,'DSN3EPX,?'
  ```

  Change it to look like this:

  ```
  DB2P,DSN3INI,'DSN3EPX,?,S,DB2P'
  ```

- Add new IEFSSN*xx* definitions for the new names:

  ```
  DB1G,DSN3INI,'DSN3EPX,-DB1G,S,DB2P'
  DJ1G
  ```

Optionally, you can use the MVS command SETSSI to add the **new** IEFSSN*xx* statements without an IPL. You can do this now or later during the enabling process. Don't forget to add these names to the IEFSSN*xx* member before you IPL again.

Attention: The command prefix -DB1G is not a subset or superset of DB2P's command prefix. For example, ?DB1G is invalid in this context.

- Make sure that RACF definitions are in place to handle the new subsystem names. You will have to make changes to the following RACF definitions:
  - Add the correct names to the RACF router table.
  - Add the correct names to the started procedures table (ICHRIN03), if used.

Tasks at enable time

You can do the following tasks when you bring down the DB2 subsystem for the enabling procedure described in “Enabling DB2 data sharing” on page 87.

1. Define the correct profile names for the DSNR class.

2. Replicate existing PERMIT commands to allow users and groups to access the new profiles.

3. Complete the installation panels for enabling data sharing. (You cannot change the subsystem name on the installation panels.)

   On installation panel DSNTIPH, be sure to modify the archive prefix names to include the member name, like DB2PCAT.DB1G.ARCLG1. Specify the old subsystem name (DB2P) as the group attachment name.

4. Stop DB2 with MODE(QUIESCE).

5. Run job DSNTIJUZ to assemble and link-edit the new subsystem parameter data set (DSNZP*xxx*) and DSNHDECP.

6. Rename the BSDS and active log data sets with the new prefix, like DB2PCAT.DB1G.BSDS01.

7. Update the BSDS with the renamed log data sets. **Be sure to include the same RBA ranges as the original active log data sets.**

   a. Run the utility print log map (DSNJU004) to obtain the start and end RBAs.

   b. Use access method services to rename the log data set.

   c. Run the utility change log inventory (DSNJU003) to delete the active logs with the old names.

   d. Run the utility change log inventory (DSNJU003) to add the renamed active logs with the correct ranges. There is no need to add the archive log data sets, because they are eventually replaced by the archive log data sets you specified on panel DSNTIPH.

8. If necessary, increase the size of the BSDS, as described in “Increasing the size of the BSDS” on page 73.

9. If necessary, increase the size of the SYSLGRNX table space, as described in “Increasing the size of the SYSLGRNX table space” on page 74.

10. Rename the startup procedures. For example, change DB2PMSTR to DB1GMSTR. Don't forget to change the BSDS names to the new names in the *ssnm*MSTR startup procedure.

11. Rename the IRLM startup procedure, and take this opportunity to increase the value for MAXCSA. See “Estimating a value for the IRLM MAXCSA parameter” on page 63 for more information.
12. Make sure CICS can connect to the new subsystem. There are several ways of doing this, depending on which level of CICS you are running. Here are a couple of techniques:
    - Change the CICS RCT TYPE=INIT macro to use the new DB2 subsystem name, reassemble, and link-edit.
    - Modify the JCL for CICS to include the new subsystem name on the INITPARM (for CICS Version 4 and subsequent releases).
13. Make sure IMS can connect to the new subsystem.
14. Enter the command START DB2 and continue with the rest of the enabling process.

Enabling DB2 data sharing

After you migrate and test Version 6, enable this subsystem for data sharing. The enabling process simply allows this existing subsystem to be the originating member of a data sharing group; it does not allow you to change the subsystem name. (See “Renaming the DB2 subsystem” on page 85 for information about how to rename the subsystem.) This originating member's catalog is used as the DB2 catalog for the data sharing group.

The enabling CLIST tailors these jobs:

DSNTEJ0
DSNTEJ1
DSNTEJ1L
DSNTEJ1P
DSNTEJ1S
DSNTEJ1T
DSNTEJ2A
DSNTEJ2C
DSNTEJ2D
DSNTEJ2F
DSNTEJ2P
DSNTEJ2U
DSNTJE3C
DSNTEJ3P
DSNTEJ4C
DSNTEJ4P
DSNTEJ5A
DSNTEJ5C
DSNTEJ5P
DSNTEJ6
DSNTEJ6D
DSNTEJ6P
DSNTEJ6S
DSNTEJ6T
DSNTEJ6U
DSNTEJ61
DSNTEJ62
DSNTEJ7
DSNTEJ71
DSNTEJ73
DSNTEJ75
DSNTESA
DSNTESC
DSNTESD
DSNTESE
DSNTIJDE
DSNTIJEX
DSNTIJIC
DSNTIJFT
DSNTIJGF
DSNTIJIN
DSNTIJMV
DSNTIJSG
DSNTIJTM
DSNTIJVC
DSNTIJUZ

The enabling CLIST also edits the DB2I CLISTS.

Procedure: To enable data sharing:

1. On panel DSNTIPA1, specify:

   ```
   INSTALL TYPE ===> INSTALL
   DATA SHARING ===> YES
   ```

2. On panel DSNTIPP1, specify 3 to enable data sharing.
3. On panel DSNTIPK, specify:

   ```
   GROUP NAME           ===> group name
   MEMBER NAME          ===> originating member name
   ```

Verify that the *originating member name* is unique within your MVS Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the group name and member name you specify here.

4. Complete the installation panels, specifying parameters according to the guidelines in “The scope and uniqueness of DB2 subsystem parameters” on page 76.

5. Complete the following installation steps, as described in Section 2 of *DB2 Installation Guide*.

   a. Stop DB2 activity.

      See Migration Step 10 in Section 2 of *DB2 Installation Guide* for a detailed list of steps.

   b. Installation Step 1: Define DB2 to MVS: DSNTIJMV

   c. Installation Step 3: Define System Data Sets: DSNTIJIN

      For the enable process, DSNTIJIN only alters the current active log data sets to use SHAREOPTIONS (2 3).

   d. Installation Step 5: Define DB2 Initialization Parameters: DSNTIJUZ

   e. Installation Step 7: Record DB2 Data to SMF (Optional)

   f. Installation Step 9: Connect DB2 to TSO: DSNTIJVC

      This step is necessary only if you are using the group attachment name.

   g. Installation Step 12: IPL MVS

      This step is only necessary if you are changing the command prefix, or adding or changing the group attachment name.

   h. Installation Step 13: Start the DB2 Subsystem

   i. Installation Step 14: Define Temporary Work Files: DSNTIJTM

   j. Installation Step 17: Image Copy the DB2 Directory and Catalog: DSNTIJIC (Optional)

      This is an optional step. If you decide to do this, use the DSNTIJIC job that is generated during the installation or migration of the originating member.

      You do not need to make an image copy for the DB2 catalog or user data sets for recovery, because DB2 uses image copies made before you enabled data sharing.

   k. Installation Step 18: Verify the Installation

6. Optionally, run the installation verification procedures (IVP), or a subset of these, as described in Section 2 of *DB2 Installation Guide*.

   If you ran the complete set of IVP sample jobs after you migrated to Version 6, you probably don't need to run these jobs again. When you start the originating member, DB2 checks your coupling facility, group, and member definitions and verifies that data sharing is enabled. You can also verify that the group has been correctly established by issuing the DB2 DISPLAY GROUP command after the originating member has completed startup.

Adding a new DB2 data sharing member

You always add more members to the group as **new installations**. You cannot take an independently existing DB2 subsystem and merge it into the group. The new members begin using the DB2 catalog of the originating member.

DB2 does not have an automatic way to “merge” catalogs and resolve naming conflicts. If you have applications that are currently running on several existing DB2s, your migration plan might include procedures for moving the relevant data and applications from those DB2s onto one or more of the group members and for resolving any catalog naming conflicts that result. See “Merging existing DB2 data into the group” on page 90 for more information about this.

After you have installed a new data sharing group or enabled an existing subsystem for data sharing, you can add new data sharing members.

Jobs that the 'add member' CLIST tailors:

DSNTIJIN	DSNTIJMV	DSNTIJID	DSNTIJDE
DSNTIJTM	DSNTIJUZ	DSNTIJFT	DSNTIJGF

Procedure: To add a new data sharing member:

1. On panel DSNTIPA1, specify:

```
INSTALL TYPE ===> INSTALL
DATA SHARING ===> YES
.
.
INPUT MEMBER NAME  ===> originating member's output PDS member
OUTPUT MEMBER NAME ===> new member's output PDS member
```

2. On panel DSNTIPP1, specify 2 for adding a member.
3. On panel DSNTIPK, specify a new member name:

```
MEMBER NAME           ===> new member name
```

   Verify that the *new member name* is unique within your MVS Sysplex. Installation job DSNTIJMV edits the *ssnm*MSTR startup procedure with the member name you specify here.

4. Complete the installation panels, specifying parameters according to the guidelines in “The scope and uniqueness of DB2 subsystem parameters” on page 76.

   We recommend that you **rename the tailored SDSNSAMP** data set for each member. This data set contains tailored JCL for each member, including jobs used for disabling and re-enabling data sharing, if that should ever become necessary. If you do not rename it, it is overwritten when you install a new member. Do this by choosing a new name for the *prefix.*NEW.SDSNSAMP data set on installation panel DSNTIPT. For example, use *prefix.member_name*.SDSNSAMP.

5. Complete the following installation steps, as described in Section 2 of *DB2 Installation Guide*.

   a. Installation Step 1: Define DB2 to MVS: DSNTIJMV

   b. Installation Step 3: Define System Data Sets: DSNTIJIN

When you add a new member, DSNTIJIN does not define catalog and directory data sets. It does define the new BSDS and active log data sets to use SHAREOPTIONS (2 3).

c. Installation Step 4: Initialize System Data Sets: DSNTIJID

   When you add a new member, DSNTIJID does not initialize the catalog and directory data sets. It does add the new active log data sets to the BSDS.

d. Installation Step 5: Define DB2 Initialization Parameters: DSNTIJUZ

e. Installation Step 7: Record DB2 Data to SMF (Optional)

f. Installation Step 8: Establish Subsystem Security (Optional)

g. Installation Step 10: Connect IMS to DB2 (Optional)

h. Installation Step 11: Connect CICS to DB2 (Optional)

i. Installation Step 12: IPL MVS

   If you are using MVS Version 5 Release 2 or later releases, you can use the MVS command SETSSI to dynamically add the new DB2 and IRLM subsystems without having to IPL MVS. For example, the following two commands can be used to add subsystems for DB2 and IRLM to MVS3:

```
RO MVS3,SETSSI ADD,SUB=DB3G,INITRTN=DSN3INI,INITPARM='DSN3EPX,-DB3G,S,DB0G'
RO MVS3,SETSSI ADD,SUB=DJ3G
```

   Do take the time to add DB2 to the IEFSSN*xx* member so that it can be used on a subsequent IPL. For more information, see *OS/390 MVS Using the Subsystem Interface* for more information.

j. Installation Step 13: Start the DB2 Subsystem

k. Installation Step 14: Define Temporary Work Files: DSNTIJTM

l. Installation Step 17: Image Copy the DB2 Directory and Catalog: DSNTIJIC (Optional)

   This is an optional step. If you decide to do this, use the DSNTIJIC job generated during the installation or migration of the originating member.

m. Installation Step 18: Verify the Installation

6. Test the data sharing group, as described in “Testing the data sharing group” on page 94.

Merging existing DB2 data into the group

DB2 provides no way to merge existing DB2 data or subsystems into a data sharing group. Read “Decide if merging is the right thing to do” on page 67 before attempting to merge subsystems into a single data sharing group.

Merging subsystems

If, for some reason, you want to merge existing subsystems, you must do the following:

1. Choose a subsystem to be the originating member.
2. Move data and data definitions from the other DB2s to the originating member.

3. Add those other DB2 subsystems to the group using the new member install process.

Merging data

If you have an application that is now running on independent DB2 subsystems, you might decide that it is an application that will work well in a data sharing group. In that case, you must move the data and merge the catalog definitions for that application from the independent DB2s into the data sharing group. Because those independent DB2s still exist, you cannot reuse their subsystem names when installing new members into the data sharing group.

DB2 does not provide an automated way to move catalog definitions from one DB2 into the catalog of the data sharing group. If you have procedures and tools in place now to do things such as move applications from test to production, or to handle merging databases from enterprise reorganizations or mergers, those same procedures can be used to move applications into the data sharing group.

Existing distributed applications

If you move existing data to the data sharing group, it is likely that the location name of objects will change. Existing distributed applications that remotely reference that object by its three-part name must be changed to the new name, and any aliases on that table must also be dropped and recreated with the new location name.

Any application containing explicit SQL CONNECT statements that reference an old location name must be recoded. Any DB2 plan that uses an old location name for the CURRENTSERVER keyword must be bound again.

Procedure for moving data

The procedure described here is very difficult. It assumes you are going to change the catalog alias of the data sets to the alias of the data sharing group. If you want to keep the existing catalog alias, you must include steps to protect the data sets (such as by creating dummy data sets) so as not to overlay those data sets when you create the objects in the data sharing group.

In the following procedure, we use the term “target” for the DB2 subsystem you are moving data to (the data sharing group), and “source” for the DB2 you are moving data from.

1. Decide whether you are going to use DSN1COPY to move the data:

   The DSN1COPY utility copies the data and translates OBIDs. This is the simplest way, operationally, but translating OBIDs is an I/O-intensive process. Every record has the table OBID in its header, and this OBID must be translated.

   Another method can be used to avoid the heavy I/O used for translating OBIDs but is more complex. With this method, you use the existing data sets (making no copies), and the same table OBIDs as on the source subsystem. You must use the REPAIR utility to change the OBIDs for index spaces and table spaces in the header pages. REPAIR is also needed to reset the LEVELID of the page sets before you restart them on the target subsystem.

2. Choose a DB2 subsystem's catalog to be the “original” catalog for the data sharing group. In other words, this DB2 is the originating member of the group.

There are many considerations in choosing which DB2 to be the originating member of the group. For example, it probably makes sense to choose the member with the most database objects as the originating member so you can avoid moving as many objects as possible.

However, if all DB2s are pretty much equal, and if you are planning *not* to use DSN1COPY, also consider the log RBA values of the existing subsystems. Compare the end-of-log RBAs with the high order 6 bytes of the time-of-day clock timestamp on each of their systems (this is called the "truncated" timestamp). The usual case is that the RBA will be less than the truncated timestamp. In this case, any DB2 can be chosen as the originating member.

In the event that the current end-of-log RBA in any of the existing DB2 subsystems is higher than the 6-byte truncated timestamp value at the time you are ready to enable sharing, then either:

- Choose the DB2 subsystem that has the highest RBA as the originating member, or
- Use DSN1COPY with the RESET option to reset the log RBAs in each data and index page to 0 when you move databases from other DB2 subsystems to the data sharing group.

3. Resolve name conflicts among the objects and authorization IDs in the data sharing group.

4. Create the objects on the target subsystem.

   If you are using DSN1COPY with OBIDXLAT, you can enter the CREATE statements in any order. DB2 assigns new OBIDs for these objects.

   If you are not using DSN1COPY, or are using DSN1COPY without OBIDXLAT, you must query the DB2 SYSIBM.SYSTABLES catalog on the source DB2 subsystem to get the table OBIDs for tables that are within a database.

   When you run the SQL on the target subsystem, use the OBID clause on the CREATE TABLE statement to specify an OBID the same as the table OBID on the original subsystem. However, you must be sure that the OBID you specify is available. If an OBID is being used for another object within the database (say, for example, an index or referential constraint), DB2 won't allow you to create the table with the OBID you want.

   One way to guarantee the availability of OBIDs for all tables within a database is to defer the creation of all indexes and referential constraints until all tables are created. All CREATE TABLE statements must have the OBID clause to guarantee that they are assigned the correct OBID. If an explicit table space name is specified, then the CREATE TABLESPACE statement must come immediately before the first CREATE TABLE statement for that table space. This helps prevent a group of CREATE TABLESPACE statements from using up OBIDs that are needed for the tables.

5. On the source, run the utility REORG on any table spaces for which the following conditions are both true for a table in the table space:

   - A column has been added (ALTER TABLE ADD COLUMN) with no subsequent REORG
   - The columns are all fixed-length

   If you are unsure if a table meets this criteria, query SYSIBM.SYSTABLES for those tables in which CREATEDTS does not equal ALTEREDTS. However, there is no way to tell from the DB2 catalog whether the ALTER consisted of adding a column.

6. Stop the database on the target subsystem. If you are not using DSN1COPY, go to step 8.
7. Use DSN1COPY with the OBIDXLAT and RESET options to translate the OBIDs, to reset the level ID, and to copy the data sets.

   After you've completed this step, go to step 12.
8. Delete the data sets you created on the target subsystem when you ran the SQL to create the objects. Rename the source data sets to the high level qualifier of the target.
9. Record the object identifiers of the indexes and table spaces you just created on the target.

   Query the SYSIBM.SYSTABLESPACE catalog table to get the DBID and PSID of the table spaces. Query the SYSIBM.SYSINDEXES catalog table to get the DBID and ISOBID of the indexes.
10. Use the REPAIR utility to change the identifiers in the page set header page (or header pages, if you have a partitioned page set) to match the new identifiers on the target subsystem.

    The identifiers consist of two 2-byte fields: HPGDBID and HPGPSID. You must locate and replace these identifiers as follows:

    - For non-partitioned page sets, locate and replace the 4 bytes starting at `X'0C'` starting on page `X'0'`.
    - For partitioned page sets, locate and replace the 4 bytes starting at `X'0C'` on page `X'n...0'` where *n* is the partition number. The description of the REPAIR utility in Section 2 of *DB2 Utility Guide and Reference* contains more information about how to specify partition numbers.
11. Use the REPAIR utility with the LEVELID option to reset the level indicator of the page sets to a neutral value.
12. Start the database on the target subsystem for read-write access.
13. Optionally, drop the database objects on the source subsystem.
14. Make full image copies of all data. This is the earliest time to which data recovery can occur after the merge.
15. Run the RUNSTATS utility on the target.
16. All plans and packages on the source subsystem have to be bound on the target subsystem. Any plans and packages on the target subsystem that change because of name conflict resolution must be bound as well. All plans and packages will have to have the appropriate authorizations granted.

Testing the data sharing group

When you installed DB2, there were sample objects created in job DSNTEJ1. The DSNTESD member of *prefix*.SDSNSAMP contains SQL statements that refer to these objects. These SQL statements can be used to test group buffer pool caching, global lock serialization, and concurrency in the data sharing group. Do these tests after you have installed several data sharing members.

Test group buffer pool caching

Use the SQL statements in DSNTESD to verify that the group buffer pool operates correctly.

1. Run SPUFI on more than one data sharing member, using DSNTESD as the input data set. Specify AUTOCOMMIT=YES on the SPUFI panel.

   Run SPUFI on the different members serially, a few seconds apart if possible. (The runs must be close enough together to avoid having DB2 close the page set because of infrequent updates. The default amount of time between updates before DB2 switches the page set from read-write to read-only is 10 minutes, as specified on the RO SWITCH TIME field of installation panel DSNTIPN. RO SWITCH CHKPTS can also cause the data set to switch to read-only. It is a number of consecutive checkpoints.

   By running the SQL statements serially, DB2 detects inter-DB2 R/W interest on the table space and index and uses the group buffer pool.

   Verify that ITEM_COUNT increases by 5 after each run.

2. After running SPUFI on more than one member, issue the following command to see if the table space and index are using the group buffer pool.

   ```
   -DB1G  DIS DB(DSN8D61A) SPACENAM(DSN8S61S,XPARTS) LOCKS
   ```

   If the P-lock state is IX or SIX, then the table space and index are group-buffer-pool-dependent, as they should be.

3. Issue the following command to display the statistics for GBP0:

   ```
   -DB1G DIS GBPOOL(GBP0) GDETAIL
   ```

   In the group detail statistics, look for non-zero values in the READS and WRITES values of the display. This indicates that DB2 is using the group buffer pool successfully for caching.

Test global lock serialization

Use the SQL statements in DSNTESD to verify that locks are acquired and released correctly across multiple data sharing members.

1. Run SPUFI, using DSNTESD as input, on *member 1*. Specify AUTOCOMMIT=NO.

   Because you have inserted data into DSN8610.PARTS but have not committed, *member 1* holds global locks.

2. Run SPUFI, using DSNTESD as input, on *member 2*. Specify AUTOCOMMIT=NO.

   Because *member 1* holds global locks, *member 2* must wait to perform the insert.

3. In less than a minute, commit on *member 1*. (If you wait too long to commit, *member 2* will experience a lock timeout.)

   The global locks should be released, and *member 2* should be able to proceed. Verify that ITEM_COUNT has increased by 5.

Test concurrency

Use the SQL statements in DSNTESD to test concurrency within the data sharing group.

1. Run SPUFI concurrently on different data sharing members. Specify AUTOCOMMIT=YES.

   Global locking ensures that inserts to DSN8610.PARTS are coordinated across data sharing members.

2. Verify that ITEM_COUNT increases by 5 each time the run completes successfully.

Test Sysplex query parallelism

There is no sample procedure to test Sysplex query parallelism, but there is a way you can use your own data to confirm that a single query can be processed on more than one member of the data sharing group. Choose an existing query that you know uses CP parallelism, such as a SELECT COUNT(*) for a table in a large partitioned table space, and use the following procedure that forces DB2 to split the query across multiple DB2s:

1. Decide which DB2 will be the query coordinator for your test, and make sure the COORDINATOR field on installation panel DSNTIPK is set to YES for that DB2.

2. Make sure that all assistants have the ASSISTANT field on installation panel DSNTIPK set to YES.

3. Make sure your statement does not include one of the restrictions listed in Section 5 (Volume 2) of *DB2 Administration Guide*.

4. Run EXPLAIN on the statement.

   An X in the PARALLELISM_MODE column of the PLAN_TABLE output indicates that this statement can be split across multiple DB2s.

5. Set buffer pool allocation thresholds on the DB2s that you want considered as possible assistants:

   ```
   ALTER BUFFERPOOL (BPn) VPSIZE(z) VPSEQT(100) VPPSEQT(100) VPXPSEQT(100)
   ```

   Ensure that the VPSIZE is large enough to support parallel processing. Start with at least 50 buffers on each query assistant.

6. Make sure that accounting trace class 3 is active on the coordinating DB2 subsystem.

7. Run the query.

8. Inspect the IFCID 0003 trace record. Field QWA01RBN corresponds to the number of assisting DB2s. This value should be greater than 0.

Updating subsystem parameters for a member

There are no group-wide subsystem parameters that can be updated. To update subsystem parameters for an existing DB2 member within the DB2 data sharing group, specify the following installation options:

```
INSTALL TYPE         ===> UPDATE
```

The created installation job creates new parameters for the DB2 member.

Migrating an existing data sharing group to the new release

Migrating a data sharing group requires careful planning:

1. Read the information about migration considerations in Section 2 of *DB2 Installation Guide*.
2. Read the information in "Considerations for mixed releases in a data sharing group."
3. Make a plan to migrate the data sharing group over as short a time as possible.
4. Apply the fallback SPE (and its prerequisite APARS) to the Version 5 load library before attempting to migrate any member of the group. Stop and restart each member to pick up the change.
5. Follow the procedure in "Procedure to migrate the data sharing group" on page 103. Refer to Section 2 of *DB2 Installation Guide* for detailed information about migration. **You must complete the migration of the first member of the data sharing group to Version 6 before starting any other members at Version 6.** To prepare for fallback, keep the subsystem parameter load module used by Version 5.

Considerations for mixed releases in a data sharing group

DB2 allows the data sharing group to remain available while you migrate members of the group to the newest release. However, it is best to plan the migration during periods of low activity in the group because the DB2 catalog is unavailable for a brief time during the migration of the first member.

Recommendation

There is no single recommendation that can be made for all DB2 installations. The purpose of coexistence is to allow for migration of individual data sharing members so that your applications can continue to access DB2 data while the members are being migrated. However, you must weigh the benefit of improved availability against the operational costs of running in coexistence mode: many new functions are not available, and there are some system management issues to consider.

If you do not require continuous availability, it is easier to shut down the group for the migration to avoid the coexistence environment. If you need to run in coexistence mode, make a plan to migrate the members within as short a time as possible so that you can take advantage of the new functions available in the current release and minimize the operational complexity.

Determining the release of the DB2 group

When the first data sharing member starts Version 6, the entire group is now considered to be at that level, even though not all individual members of the group have migrated. This means that the *group level* is Version 6. You can see the group level by issuing the DISPLAY GROUP command as shown in Figure 18.

```
- DSN7100I -DB1G DSN7GCMD
- *** BEGIN DISPLAY OF GROUP(DSNDB0G ) GROUPLEVEL(610)
-                                      GROUP ATTACH NAME(DB0G)
- ---------------------------------------------------------------------
- DB2                                    DB2 SYSTEM    IRLM
- MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
- -------- --- ----   --------  -------- --- --------  ----   --------
- DB1G       1 DB1G   -DB1G     ACTIVE   610 MVS1      DJ1G   DB1GIRLM
- DB2G       2 DB2G   -DB2G     ACTIVE   610 MVS2      DJ2G   DB2GIRLM
- DB3G       3 DB3G   -DB3G     ACTIVE   510 MVS3      DJ3G   DB3GIRLM
- DB4G       4 DB4G   -DB4G     FAILED   610 MVS4      DJ4G   DB4GIRLM
- ---------------------------------------------------------------------
- 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(DSNDB0G )
- DSN9022I -DB1G DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 18. DISPLAY GROUP command shows group and member release level

Determining the function level of the IRLM group

IRLM communicates coexistence information by using function levels. A function level is an ever- increasing number that each IRLM can use to tell other IRLMs in the group what level of function it supports. The group function level is the minimum of the individual IRLM function levels for all IRLMs that can coexist. Any IRLM that tries to join a data sharing group is prevented from doing so by active members that cannot coexist with the new IRLM's function level.

When the function level for the group changes, that change is serialized by IRLM with lock structure rebuilds. In most cases, however, the lock structure does not actually do a full rebuild. The first phase of rebuild is enough to quiesce the work and cause the function level change to occur. These “partial” rebuilds take place when an IRLM joins or leaves the group and if that activity causes the group function level to change. For example, if the IRLM group is currently at function level n, and the IRLM member that wants to join the group is at n-1, the partial rebuild occurs to lower the group function level. Conversely, if the lowest level member leaves the group, the partial rebuild might occur if the group can coexist at a higher function level.

To display the function levels, enter the following command:

```
MODIFY irlmproc,STATUS,ALLI
```

You get output like that shown in Figure 19 on page 98:

```
 DXR103I LRLM STATUS IRLMID=007
IRLMS PARTICIPATING IN DATA SHARING GROUP FUNCTION LEVEL=013
IRLM_NAME IRLMID STATUS LEVEL SERVICE MIN_LEVEL MIN_SERVICE
  JRLM      005    UP    015  PQ15290    012      PN90337
  KRLM      006    UP    013  PN92893    006      IRLM2.1
  LRLM      007    UP    014  PN09381    006      IRLM2.1
```

Figure 19. Determining IRLM function levels

The IRLMs shown in Figure 19 are at group function level 13, which is the lowest level of any of the individual members (KRLM). The MIN_LEVEL field shows the minimum level with which this IRLM can coexist. MIN_SERVICE indicates the service or release that corresponds with that MIN_LEVEL.

Call attachment and TSO attachment coexistence

While you are in a coexistence environment, you can attach to either release of DB2 with your existing TSO logon procedures or JCL. After you migrate all members of the group to the latest level of DB2, be sure to update those procedures and jobs to point to the latest level of DB2 load libraries.

Avoiding automatic rebinds

When developing your migration plan, keep in mind that new functions introduced in this release are not available to members of the group who have not yet migrated. Thus, it is best to either:

- Migrate all members to the new release before attempting to use new utilities or commands, or any new options. Do not allow members to run any applications that include new SQL function until all members have migrated to the new release.
- Don't allow packages and plans bound on Version 6 to execute on members that have not yet migrated. Or, do not allow plans or packages to be bound on Version 6.

  This serves two purposes. First, if those Version 6-bound plans or packages are using new functions, you can avoid the application errors that occur if the plan or package tries to execute an SQL statement that is not allowed in Version 5. Second, it avoids the automatic rebind that occurs when any plan or package that is bound on Version 6 is run on Version 5. It also avoids the automatic rebind that occurs when a Version 6-bound plan or package that was automatically rebound on Version 5 is later run on Version 6.

If it is not possible to enforce on which member a plan or package runs, consider how you want to handle binds and automatic rebinds while two releases are coexisting. One approach is to disallow all binds and disable all automatic rebinds on the Version 6 subsystem. The other approach is to disable only those automatic rebinds that occur on Version 6 in step 3 of the following scenario:

1. Plan or package is bound on Version 6
2. Next, the plan or package is run on Version 5 (automatic rebind occurs on Version 5)
3. Next, the plan or package is run on Version 6 (automatic rebind occurs on Version 6)

   Because DB2 perceives this scenario as a fallback and remigration scenario, the autobind that occurs in step 3 is called a “remigration” rebind.

By disallowing the automatic rebind in step 3, you are avoiding the thrashing that can occur by having the plan or package rebind every time it runs on a member of a different level.

Disallowing all binds: You can do the following on the Version 6 member to avoid automatic rebinds:

- Specify NO on the AUTO BIND field of installation panel DSNTIPO. This disables *all* automatic rebinds on the Version 6 member for any reason. This means you cannot run a plan or package on a Version 6 subsystem if it has gone through the following scenario:

  1. Bind on Version 6.
  2. Run on Version 5. This causes an automatic rebind on that Version 5 subsystem.
  3. Try to run on Version 6. This returns a -908 SQLCODE (SQLSTATE '23510') because DB2 must autobind the plan or package on Version 6 before running it there.

- Use the resource limit facility to disallow BIND operations. Do this by inserting rows in the resource limit specification table (RLST) to set RLFFUNC to “1” and RLFBIND to “N.” This ensures that nobody binds plans or packages on Version 6.

  Here is an example of an INSERT statement for the RLST that disallows all BIND operations for all authorization IDs (except those with installation SYSADM or installation SYSOPR) for all packages and plans:

  ```
  INSERT INTO authid.DSNRLSTxx
    (RLFFUNC,RLFBIND) VALUES('1','N');
  ```

After all the DB2 members in the data sharing group are running at the current release, enable automatic rebinds again by setting AUTO BIND=YES, and allow bind operations by changing the RLST accordingly or by stopping the resource limit facility using the STOP RLIMIT command.

Disallowing only the automatic remigration rebind: To avoid the automatic remigration rebind in the case described in step 3under “Avoiding automatic rebinds” on page 98, specify COEXIST for the AUTOBIND field on installation panel DSNTIPO of the Version 6 members. This means that automatic rebind occurs on Version 6 in the following circumstances:

- The plan or package is marked invalid
- You migrate to a future release, bind a plan or package on that release, and then run the plan or package on Version 6.

When all members are at Version 6, you do not need to change the COEXIST value. The behavior is the same as if you had specified AUTOBIND YES.

Recommendations for BIND

If DSN is under Version 6 and if the DB2 member that is named in the DSN command is Version 5, a BIND or REBIND subcommand that uses any of the following options is rejected:

- OPTHINT (non-blank) for BIND PLAN or PACKAGE
- OPTHINT (any value) for REBIND PLAN or PACKAGE
- DBPROTOCOL(DRDA) for BIND PLAN or PACKAGE
- DBPROTOCOL (any value) for REBIND PLAN or PACKAGE
- The DYNAMICRULES values of DEFINEBIND, DEFINERUN, INVOKEBIND, or INVOKERUN for BIND and REBIND PACKAGE
- PATH for BIND and REBIND PLAN or PACKAGE
- PATHDEFAULT for REBIND PLAN or PACKAGE
- REBIND TRIGGER PACKAGE

To avoid problems, make sure the DB2 subsystem named in the DSN subcommand matches the load libraries that are used for the DSN command.

Recommendations for Utilities

Until all members of the data sharing group are running at the current release, avoid using any of the utilities functions available in Version 6. However, as long as you use utility options that are supported in Version 5, utilities can attach to either a Version 5 or Version 6 subsystem.

The utilities batch module (DSNUTILB) is split into two modules: a release-independent module (still called DSNUTILB) and a release-dependent module (DSNUT510 or DSNUT610). To operate in a mixed-release data sharing environment, you must have both DSNUT510 and DSNUT610 available to the utility jobs that operate across the data sharing group. Ways of doing that are in “Changing STEPLIB in DSNUPROC” and “Cross-Copy into load libraries.”

Changing STEPLIB in DSNUPROC: The recommended way of making the release-dependent modules available for utility jobs is to change the STEPLIB in DSNUPROC to include the other release as in the following example:

```
//DSNUPROC PROC LIB='DSN610.SDSNLOAD',
//         SYSTEM=DSN,
//         SIZE=0K,UID='',UTPROC=''
//DSNUPROC EXEC PGM=DSNUTILB,REGION=&SIZE,
//         PARM='&SYSTEM,&UID,&UTPROC'
//STEPLIB  DD   DSN=&LIB,DISP=SHR;
//         DD   DSN=DSN510.SDSNLOAD,DISP=SHR <– coexistence
//SYSPRINT DD   SYSOUT=*
//UTPRINT  DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//*DSNUPROC PEND        REMOVE * FOR USE AS INSTREAM PROCEDURE
```

Cross-Copy into load libraries: Another approach, not recommended for long-term use, is to cross-copy the release-dependent modules into the load libraries of the other release. So, copy DSNUT510 into the Version 6 load libraries, and copy DSNUT610 into the Version 5 load libraries. The problem with this approach is that you must repeat this procedure every time you apply maintenance to these modules. Thus, as with coexistence in general, this approach is only for short-term use.

Here is some sample JCL to do the cross-copy:

```
//CROSCOPY PROC D610TPRE='DSN610',
//             D510TPRE='DSN510',
//             RGN=4096K,SOUT='*'
//* *******************************************************************
//*    FOR EXECUTION OF IEBCOPY - DB2 POST-INSTALLATION             ***
//* *******************************************************************
//COPY     EXEC PGM=IEBCOPY,REGION=&RGN
//SYSUT3   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT4   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//* **************  DB2 TARGET LIBRARIES  *****************************
//*
//D510LOAD DD  DSN=&D510TPRE..SDSNLOAD,DISP=OLD
//D610LOAD DD  DSN=&D610TPRE..SDSNLOAD,DISP=OLD
//*
//         PEND
//COEXIST  EXEC PROC=DSNTIJCO
//SYSIN    DD *
  COPYMOD INDD=((D610LOAD,R)),OUTDD=D510LOAD
   SELECT MEMBER=(DSNUT610)
  COPYMOD INDD=((D510LOAD,R)),OUTDD=D610LOAD
   SELECT MEMBER=(DSNUT510)
```

## Recommendation for group restart

If a group restart is necessary while the data sharing group is running with mixed releases, we recommend that you issue the START command only for Version 6 members. Do not start the Version 5 members until the Version 6 members have completed forward log recovery. This recommendation is because if a Version 5 member performs the group restart for a Version 6 member, it adds pages to the logical page list during the peer-forward recovery phase when it tries to apply redo log records against a release-dependent object.

## Recommendation for SPUFI

When you migrate the first member of the data sharing group to Version 6, you run DSNTIJSG which rebinds SPUFI in Version 6. Binding SPUFI in Version 6 causes SPUFI to be unavailable to the Version 5 members. If you attempt to run an SQL statement in a data sharing member at Version 5, expect messages that indicate an unavailable resource.

Coexistence considerations for specific functions

The purpose of this section is to clarify how some of the new Version 6 functions work (or not) when there are mixed releases in the data sharing group. These considerations are similar to those you have to understand for the fallback environment. Objects that are frozen in fallback are generally not accessible from a Version 5 member. See the fallback considerations section of Section 2 of *DB2 Installation Guide* for more information.

New page sizes: Table spaces that use an 8-KB or 16-KB buffer pool are not available to Version 5 members. You cannot issue the ALTER or DISPLAY BUFFERPOOL commands for those buffer pools from a Version 5 system.

New built-in functions: DB2 uses the new CURRENT LOCALE LC_CTYPE special register when evaluating the TRANSLATE, UPPER, and LOWER scalar functions.

New GBPCACHE options: Table spaces or indexes that are defined with GBPCACHE NONE or GBPCACHE SYSTEM are not available to Version 5 members. Table spaces that are defined in a buffer pool defined with GBPCACHE NO are also not available to Version 5 members if they are group buffer pool–dependent.

TRACKMOD options: If you try to use COPY from a Version 5 member for a table space defined with TRACKMOD NO, the result will always be a full image copy. If you change from TRACKMOD NO to TRACKMOD YES and then try to copy from a Version 5 member, the first copy will be a full image copy.

New resource limit facility columns for predictive governing: If you populate the new columns in the resource limit facility table, those columns are ignored on a Version 5 member. There is no predictive governing on a Version 5 members.

Populating DSN_STATEMNT_TABLE: You cannot use EXPLAIN to populate the DSN_STATEMNT table from a Version 5 member.

Duplexed group buffer pools: Do not start duplexing until all members of the group are at the level of DB2 and OS/390 code that allows them to be duplex-enabled, and until the group buffer pool is allocated in a coupling facility with coupling facility control code level 5 or higher. No DB2 members can connect to a duplexed group buffer pool if they are not duplex-enabled.

If any downlevel members are connected to a group buffer pool, duplexing cannot start for that group buffer pool until all downlevel members are disconnected from the group buffer pool.

Postponed backout processing: A Version 5 member cannot request limited backout processing. However, if you enter a DISPLAY DATABASE command on a Version 5 subsystem, it is possible for that display to show the advisory restart pending (AREST) state of page sets that can result when Version 6 members use postponed backout processing.

Similarly, if you issue a DISPLAY GROUP command on a Version 5 subsystem, a member that is both active and has indoubt URs is displayed as ACTIVE (they are displayed as 'A I' on a Version 6 subsystem).

The meaning of the 'Q I' status is slightly different for Version 5 and Version 6 members in the display. For a Version 6 member, it can indicate that postponed abort URs are present.

Optimization hints: Although a Version 5 member can use a Version 6 PLAN_TABLE, it cannot use any hints that you've added to the PLAN_TABLE. If a static statement is bound using optimization hints, that statement can be executed on a Version 5 member as long as that statement has no other release dependencies. The static statement can also run on another Version 6 member that does not have optimization hints enabled.

For dynamic statements, hints are only used if the subsystem that the statement is running on has enabled optimization hints.

Changing partitioning values: You cannot issue the ALTER INDEX statement with the VALUES clause from a Version 5 member. Also, while a table space has any partitions in REORG pending, that entire table space cannot be accessed from a Version 5 member. However, you can run REORG TABLESPACE from the Version 5 member to turn off the REORG pending state and to rebalance the partitions.

Data sets that use extended addressability: The following objects are not available from a Version 5 member:

- Partitioned table spaces with a DSSIZE greater than 4 GB
- Partitioning indexes for those partitioned table spaces
- Nonpartitioning indexes created with a PIECESIZE value greater than 4 GB

Nonpartitioning indexes of greater than 128 pieces As of Version 6, your nonpartitioning index can grow to 254 pieces (from a previous limit of 128). As long as that nonpartitioning index is not for an EA-enabled table space, you can still access a nonpartitioning index of more than 128 pieces.

Change RECOVER INDEX to REBUILD INDEX: As of Version 6, you must use the syntax REBUILD INDEX to build an index from data. RECOVER INDEX is used to recover an index from an image copy of the index and with log record updates. Install APAR PQ09842 on your Version 5 subsystems and change utility JCL to use the REBUILD INDEX syntax so that you can run the JCL on any subsystem on the group and to avoid confusion over the fact that RECOVER INDEX works differently on Version 5 than it does in Version 6.

Sysplex query parallelism: If a plan or package that uses Sysplex query parallelism is bound on Version 6, then DB2 will not distribute queries to Version 5 subsystems. If the plan or package is bound on Version 5, though, queries can be distributed to both Version 5 and Version 6 subsystems.

Revoking SYSADM or SYSCTRL in Version 5: If SYSADM or SYSCTRL grants privileges on new objects in Version 6, you cannot revoke SYSADM or SYSCTRL from a Version 5 subsystem. This operation is disallowed on Version 5 because the revoke cascades, and a DB2 Version 5 is not aware of the new objects or privileges in Version 6. You must issue the revoke from a Version 6 subsystem.

Restriction on MODIFY RECOVERY: You cannot run the MODIFY RECOVERY utility from a Version 5 member for indexes that are defined as COPY YES.

Procedure to migrate the data sharing group

Jobs that the migration CLIST tailors: Jobs marked with an asterisk (*) are modified only when you migrate the first member.

DSNTIJEX*
DSNTIJIC*
DSNTIJFT
DSNTIJFV
DSNTIJGF
DSNTIJSU*
DSNTIJIN*
DSNTIJMV
DSNTIJSG*
DSNTIJVC*
DSNTIJTC*
DSNTIJTM*
DSNTIJUZ

The DB2I CLISTs and sample jobs also get edited for the first member to migrate.

Before migrating to Version 6, you must have the fallback and coexistence SPE (and its prerequisite APARs) installed on the Version 5 load library. Version 6 members cannot start if any one of the active Version 5 members do not have the SPE applied. Similarly, if a Version 6 member is started, a Version 5 member cannot start unless it has the fallback and coexistence SPE applied.

Attention: Follow these directions carefully. The first member of the data sharing group uses DSNTIDXA as its input member name. Subsequent members **must** use a previous member's output member as its input member name.

To migrate the data sharing group:

1. For the **first** member to migrate, specify the following on panel DSNTIPA1:

   ```
   INSTALL TYPE ===> MIGRATE
   DATA SHARING ===> YES
   .
   DATA SET NAME(MEMBER)==>
   this member's output member from Version 5
   .
   INPUT MEMBER NAME  ===> DSNTIDXA
   OUTPUT MEMBER NAME ===> this member's output PDS member
   ```

2. On panel DSNTIPP2, specify 1 to indicate that this is the first member of the group to migrate.
3. Complete the installation panels. Specify parameters according to the guidelines in "The scope and uniqueness of DB2 subsystem parameters" on page 76.
4. Complete all migration steps, as described in Section 2 of *DB2 Installation Guide*.
5. Migrate the **next** member of the group. Specify the following on panel DSNTIPA1:

   ```
   INSTALL TYPE ===> MIGRATE
   DATA SHARING ===> YES
   .
   DATA SET NAME(MEMBER)==>
   this member's output member from Version 5
   .
   INPUT MEMBER NAME  ===> first member's output PDS member
   OUTPUT MEMBER NAME ===> this member's output PDS member
   ```

6. Specify 2 on panel DSNTIPP2 to indicate that this is not the first member of the group to migrate.
7. For this next member, complete the following migration steps, as described in Section 2 of *DB2 Installation Guide*.
   a. Migration Step 6: Connect DB2 to TSO

      You do not have to run job DSNTIJVC for the second and subsequent members.

   b. Migration Step 7: Connect IMS to DB2 (Optional)

   c. Migration Step 8: Connect CICS to DB2 (Optional)

   d. Migration Step 9: Stop Version 5 Activity

   e. Migration Step 11: Define DB2 Initialization Parameters

Job DSNTIJUZ only contains a subset of the steps contained in the first migrating member's DSNTIJUZ job. DSNHDECP is not reassembled for each subsequent migrating member.

f. Migration Step 12: Establish Subsystem Security (Optional)

g. Migration Step 13: Define DB2 Version 6 to MVS

   Job DSNTIJMV contains a subset of the steps contained in the first migrating member's DSNTIJMV job. The steps to update the IEAAPFxx and LNKLSTxx members, and the steps to update the language procedures are not included when subsequent members migrate.

h. Migration Step 15: IPL MVS

   This step can be done before migrating DB2 if you have made the appropriate updates to the MVS libraries.

i. Migration Step 16: Start Version 6

j. Migration Step 23: Verify Version 6

k. For each subsequent member migration, repeat all steps beginning with step 5 on page 104.

Falling Back and Remigrating

Which procedure you use to fall back to Version 5 depends on whether Version 5 was enabled for data sharing or not when you migrated to Version 6.

- If you are falling back from Version 6 data sharing to Version 5 data sharing, use the procedure in “Falling Back to Version 5 Data Sharing.”
- If you are falling back from Version 6 data sharing to Version 5 non-data-sharing, see “Falling back to Version 5 non-data-sharing” on page 106.

Falling Back to Version 5 Data Sharing

Before falling back to Version 5, you must have the fallback and coexistence SPE and its prerequisite APARs installed on the Version 5 load library. When the first member falls back, you are running in coexistence mode, so be sure to read “Considerations for mixed releases in a data sharing group” on page 96. For complete information about falling back to Version 5, see Section 2 of *DB2 Installation Guide*.

You can do the fallback procedure on one member of the data sharing group at a time. Other members can continue to run while another member is falling back.

1. Use the ALTER GROUPBUFFERPOOL command to redefine GBPCACHE NO group buffer pools to GBPCACHE YES. If you do not alter the group buffer pools to GBPCACHE YES before falling back, connections receive a resource unavailable condition.
2. If Version 5 is not duplex-enabled for group buffer pools, you must return group buffer pools to simplex mode before falling back. If you do not return group buffer pools to simplex mode before falling back, connections receive a resource unavailable condition. Check the CFRM policy and make sure that any specifications for DUPLEX(ENABLED) are changed to DUPLEX(ALLOWED) or DUPLEX(DISABLED). For a CFRM policy of

DUPLEX(ALLOWED), use the MVS command SETXCF STOP,DUPLEX to return any duplexed group buffer pools to simplex mode. When you activate a CRFM policy of DUPLEX(DISABLED), the policy will automatically stop duplexing.

3. Then
4. Stop DB2 on the member that is falling back.
5. Reactivate Version 5 code for that member (DSNTIJFV).
6. Reconnect TSO, IMS, CICS to Version 5.
7. Start Version 5:
   a. Enter the command START DB2
      1) Check for indoubt units of recovery
      2) Check for outstanding restrictive states
8. Verify Version 5.
9. Repeat steps 4 through 8 for each member of the data sharing group.
10. When the last member is falling back, run job DSNTEJ0 to free plans and packages and drop objects created by the Version 6 sample programs.

Falling back to Version 5 non-data-sharing

If you are falling back from Version 6 to a Version 5 non-data-sharing DB2, you must do two procedures:

1. Disable data sharing as described in “Procedure to disable data sharing” on page 108. When you complete this procedure, you have a single member, the *surviving* member.
2. Fall back to Version 5, as described in Section 2 of *DB2 Installation Guide* .

Take special care when the survivor is not the originating member: If you disable data sharing so that the surviving member is not the originating member, be sure to complete the following tasks when you fall back:

1. Tailor the DSNTIJFV job to refer to the procedures used by the surviving member.
2. Modify the *ssnm*MSTR procedure of the surviving member, to include the correct BSDS names.
3. Modify the *ssnm*SPAS procedure to include the correct SUBSYS name. Modify the *ssnm*WLM procedures, if any, of the surviving member to include the correct DB2SSN parameter.
4. Rerun job DSNTIJUZ to update DSNZP*xxx* and DSNHDECP to refer to the surviving member.

Remigrating

If you followed the procedure in “Falling Back to Version 5 Data Sharing” on page 105, simply remigrate each member of the data sharing group as described in Section 2 of *DB2 Installation Guide*.

If you followed the procedure in “Falling back to Version 5 non-data-sharing,” remigrate the member as described in Section 2 of *DB2 Installation Guide*, and then re-enable data sharing, as described in “Re-enabling data sharing” on page 110.

Disabling and re-enabling data sharing

Disabling data sharing is a complex procedure. Do not make a disabling procedure part of your contingency plans for handling recovery situations. For temporary bypasses to data sharing problems, you are much better off moving to one-way data sharing. Some situations can be resolved by doing group restart.

The disabling procedure is included here in case:

- You are in the process of falling back to a non-data-sharing release of DB2, as described in “Falling back to Version 5 non-data-sharing” on page 106.
- You have made a strategic decision to move off data sharing
- One-way data sharing is not working

Do not attempt to disable data sharing without a thorough understanding of the process.

After you have disabled data sharing, only *one* DB2 from the data sharing group can access the previously shared data. That DB2 is called the *surviving* member.

Disabling data sharing

This section describes:

“Summary of disabling procedure”
“Procedure to disable data sharing” on page 108
“Data recovery after disabling DB2 data sharing” on page 110.

Summary of disabling procedure

The procedure to disable data sharing ensures that the most recent versions of all pages are externalized from the group buffer pool to DASD because DB2 does not use the group buffer pool after data sharing is disabled. You must ensure that data is written to DASD, or else **you will lose data when you start DB2 after disabling data sharing.**

You must also ensure that there is no need to recover data from information contained on other members' logs after you disable data sharing, as described in “Data recovery after disabling DB2 data sharing” on page 110. **Those logs are not available to the surviving member after you disable data sharing.** To prevent the survivor from applying inconsistent updates during recovery processing, a cold start is required to disable data sharing.

If you are planning to re-enable data sharing for this group, do not change any group-wide information in the surviving member's BSDS. This includes the catalog alias name and the database password. It also includes the DDF name and password information, even if you are not going to use DDF when you re-enable. If you change any of this information, you will have to change that value in every member's BSDS before you start the group.

Attention: Do not attempt to go through the installation process to re-enable data sharing after you have disabled it. You must use the procedure described in “Procedure to re-enable DB2 data sharing” on page 110.

Procedure to disable data sharing

The procedure to disable data sharing, and return to the Version 6 non-sharing environment is as follows:

1. Decide which member is going to be the surviving member of the group.
2. Stop all members by entering the following command for each member of the data sharing group:

   ```
   -DB1G STOP DB2 MODE(QUIESCE)
   ```

3. Start the surviving member in maintenance mode by using the following command:

   ```
   -DB1G START DB2 ACCESS(MAINT)
   ```

4. Make sure data is consistent. Enter the following commands from the surviving member of the group. Do not go on to the next step until all problems are resolved.
   - DISPLAY GROUP

     Make sure the STATUS column contains the word QUIESCED for all but the surviving member, which contains the word ACTIVE. If it does not contain QUIESCED, then you must take the following actions, depending on the value in the STATUS column:

     – If the member is FAILED, you must restart the member and stop it successfully.

     – If the member has castout problems, indoubt units of recovery, or outstanding resynchronization problems, you must start that member in maintenance mode and fix the problem.
   - DISPLAY UTILITY(*)

     If there is any remaining utility work for any member of the group, you must restart that member with ACCESS(MAINT) and either terminate the utility, or let it finish.
   - DISPLAY DATABASE(*) SPACENAM(*) RESTRICT

     If there are any restricted table spaces or index spaces (such as write error ranges, recovery pending status, or logical page list entries), recover them from the surviving member.
5. Stop the surviving member by using the following command:

   ```
   -DB1G STOP DB2 MODE(QUIESCE)
   ```

6. Stop any IRLMs that have not stopped, using the following command:

   ```
   STOP irlmproc
   ```

7. Dismantle the data sharing group.
   a. Enter the following command to display the structures for this data sharing group:

      ```
      D XCF,STRUCTURE,STRNAME=grpname*
      ```

   b. For all structures that are still allocated (STATUS:ALLOCATED) and still have connections (which appear as FAILED PERSISTENT) enter the following command to force the connections off those structures:

      ```
      SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL
      ```

c. Delete all the DB2 coupling facility structures by using the following command for each structure:

```
SETXCF FORCE,STRUCTURE,STRNAME=strname
```

d. Edit the JCL in job DSNTIJGF to point to the correct BSDS data sets.

DSNTIJGF is a change log inventory job that sets up the surviving member for a cold start. **Attention: Do not change the hex values that appear in the change log inventory CRESTART control statement. They are not real RBA values.**

e. Run job DSNTIJGF.

After you run this job, do not try to restart any of the non-surviving members. None of those members can start successfully.

8. Change the IRLM procedure to SCOPE=LOCAL.

9. Start the surviving DB2 subsystem with ACCESS(MAINT). Specify the old DSNZP*xxx*, from the non-data-sharing environment. If the surviving member is not the originating member, then you must reassemble the surviving member's subsystem parameters specifying the subsystem parameter DSHARE=NO in the invocation of the DSN6GRP macro. Also, comment out all steps from the DSNTIJUZ job except for those that reassemble and link-edit the subsystem parameters.

When you start DB2 after having run DSNTIJGF, you are required to respond with Y to a cold start prompt (message DSNJ246I on the MVS console).

This is a cold start, because DB2 increases the log RBA to a value higher than any LRSN used while sharing data. From now on, your RBAs will look like LRSNs.

10. Edit and run job DSNTIJFT, if necessary, to ensure that the surviving member's work file database is DSNDB07.

The surviving member must use DSNDB07 as its work file database. If the work file database for the surviving member is not DSNDB07, drop that work file database and run job DSNTIJFT.

11. Verify that the surviving member works by running a subset of the Version 6 installation verification sample jobs.

See the step for “Verifying Your Version 6 Subsystem” in Section 2 of *DB2 Installation Guide* for a list of these jobs.

12. To establish a new recovery point, take a full or incremental image copy or non-DB2 backup of all data. Run job DSNTIJIC to image copy the DB2 catalog and directory.

We recommend that you do this step as soon as possible after data sharing is disabled. See “Data recovery after disabling DB2 data sharing” on page 110 for more information about recovery.

13. Stop and restart DB2 for normal unrestricted access.

Data recovery after disabling DB2 data sharing

After data sharing is disabled, you cannot recover to the current point or to a previous point in time if that recovery depends on any portion of the log made before you disabled data sharing. Therefore, if there are any updates to the table space between the time of the copy and the time you disabled data sharing, DB2 does not let you use that copy as the basis for recovery. This is why we recommend that you create a new recovery point as soon as possible after you have disabled data sharing.

Using the group attachment after disabling data sharing

After disabling data sharing, you can continue to use the group attachment name. There is no need to change this to the surviving member's subsystem ID.

Re-enabling data sharing

Perform a subset of the original procedure to enable data sharing. **You cannot use the enabling process of installation to re-enable data sharing**.

The following output from the original data sharing enabling procedure remains intact after disabling data sharing and does not need to be recreated or re-specified:

- Data sharing subsystem parameters (output from the CLIST processing when enabling data sharing)
- XCF definitions
- Coupling facility definitions
- RACF definitions
- DB2 catalog and directory

Procedure to re-enable DB2 data sharing

The procedure to re-enable data sharing is as follows:

1. Edit the JCL in job DSNTIJGF.

   DSNTIJGF is a change log inventory job that sets up the BSDS of the surviving member for data sharing.

2. Run job DSNTIJGF.
3. Change the IRLM procedure to SCOPE=GLOBAL.
4. Start the surviving DB2 subsystem with the subsystem parameters used when data sharing was originally enabled.

   During startup, you will be asked to start all other members that were not quiesced at the time you disabled data sharing. You must start all these members.

5. Edit and run job DSNTIJFT on the surviving member to recreate the work file database for data sharing.

   See the directions in the prologue of job DSNTIJFT for information about editing this job to re-enable data sharing.

Removing members from the data sharing group

One of the features of DB2 data sharing is incremental growth, being able to add members to an existing group. However, there might be a situation in which you want to remove members from the group permanently or temporarily. For example, assume your group does the job it needs to do 11 months of the year. However, you get a surge of additional work every December that requires you to expand your capacity. It is possible to quiesce some members of the group for those 11 months. Those members are “dormant” until you restart them.

The same principle is used to “remove” a member of the group forever. Make sure a member is quiesced cleanly, and that member can remain dormant forever. In effect, it is removed from the group.

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

What data sets to keep

When you quiesce a member, you must keep the log data sets until such time as they are no longer needed for recovery (other members might need updates that are recorded on that 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. However during group restart, you must expect the following message:

```
DSNR020I -DB2G csect-name START MEMBER DB1G, OR REPLY 'NO' or QUIESCED'
```

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

```
DSNR030I -DB2G csect-name WILL CONTINUE WITHOUT THE DB1G 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 WTOR messages above)
- Update your group restart procedure to ensure that operators know how to respond to the DSNR020I and DSNR030I messages.

Procedure to quiesce a member

In summary, to quiesce a member of the group, you must:

1. Stop the DB2 you are going to quiesce. Our example assumes you want to quiesce member DB3G.

   ```
   -DB3G STOP DB2 MODE(QUIESCE)
   ```

2. From another member, enter the following commands:

```
-DB1G DISPLAY GROUP
-DB1G DISPLAY UTILITY (*) MEMBER(member-name)
-DB1G DISPLAY DATABASE(*) RESTRICT
```

See step 4 on page 108 for more information about using these commands. 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 do optional step 4 to create a disaster recovery archive log, start the quiesced member with ACCESS(MAINT).

   ```
   -DB3G 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. Optionally, 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:

   ```
   -DB3G ARCHIVE LOG
   ```

5. Stop DB2 again with MODE(QUIESCE).

   ```
   -DB3G STOP DB2 MODE(QUIESCE)
   ```

Chapter 5. Communicating with a data sharing group

A data sharing group can be a powerful server in your client/server environment. The group can be part of an SNA network, a Transmission Control Protocol/Internet Protocol (TCP/IP) network, or part of a network that uses both protocols. The data sharing group has a single-system image to requesting applications, whether those requests are coming in through TCP/IP or SNA network protocols. Requesting applications use the LOCATION NAME of the data sharing group to direct their SQL requests to that group.

A data sharing group can support many more threads than a single DB2 subsystem. The DDF thread limit for a DB2 group is “*n* × 150000,” where *n* is the number of DB2 subsystems in the group. Thus, a group with 16 DB2 members can support 2400000 DDF threads.

Before Reading this Chapter: Be sure to read Section 3 of *DB2 Installation Guide*.

Included in this Chapter: The following topics are described in this chapter:

An overview of the ways to access a DB2 data sharing group

SNA network alternatives: Each DB2 subsystem within the data sharing group has a unique DB2 NETID.LUNAME value. There are two ways to configure a DB2 for distributed access to the DB2 group using SNA:

- The requester associates a given location name with a list of LU names at the requester location. This is called *member routing* because the requester defines a list of LUs for the data sharing group members. This is the recommended method of support unless your requester configuration does not support it.
- Use VTAM® -supplied support for generic resources, which allows a generic LU name for the group. We call this setup *group-generic* because a single LU name serves for the entire group.

TCP/IP network: TCP/IP is supported only for access using DRDA protocols. When using TCP/IP, all members of the data sharing group use the same DRDA port number used to receive incoming SQL requests, but each member must have a resynchronization port number that is unique within the Sysplex. In the event of a failure, this unique number allows a client to reconnect to the correct member so units of work requiring two-phase commit can be resolved.

Clients connect to a data sharing group using a domain name. (It is possible to connect using an IP address, but this is not recommended because it does not work if DB2 is restarted on a different CPC.) After a client has connected to a member of the group the first time, that client receives a list of all eligible members in the data sharing group and subsequently can connect to any eligible member. This is functionally equivalent to member routing, but the requester does not have to define a name for each member of the data sharing group. See “TCP/IP workload balancing for DB2 data sharing” on page 119 for more information about balancing workloads in a TCP/IP network.

Mixed SNA and TCP/IP network: You can have the group send or receive requests using either or both network protocols. It can receive TCP/IP requests if a DRDA port and resynchronization port are defined for each member (any member that does not have this information cannot receive TCP/IP requests). When sending a request, DB2 uses the LINKNAME column of the SYSIBM.LOCATIONS catalog table to determine which protocol to use, as shown in Figure 20. If the LINKNAME value is found in the SYSIBM.IPNAMES table, TCP/IP is used. If it is not, then the SYSIBM.LUNAMES table is checked. If the value is found in SYSIBM.LUNAMES, then SNA is used.

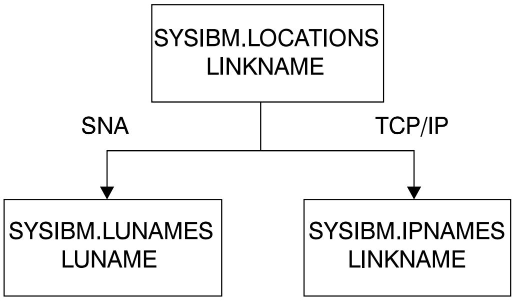


Figure 20. The LINKNAME column of SYSIBM.LOCATIONS determines which protocol to use

Attention: A requester cannot connect to a given location using both SNA and TCP/IP protocols. For example, if SYSIBM.LOCATIONS specifies a LINKNAME of LU1, and if LU1 is defined in both the SYSIBM.IPNAMES and SYSIBM.LUNAMES table, TCP/IP is the only protocol that is used to connect to LU1 from this requester. (TCP/IP is never chosen for access using DB2 private protocol, so it is possible to switch between SNA and TCP/IP.)

SNA alternatives

A given DB2 subsystem can support both member routing and group-generic, but only one can be used for a given partner. Use member routing whenever possible.

Using a single location name with a list of LU names (member routing)

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.

Requirements: The following requesters can use member routing:

- DB2 for MVS/ESA Version 4 or a subsequent release
- DB2 Connect™ Version Enterprise Edition with *Sysplex workload balancing*. DB2 Connect Version 5 Release 2 is the minimum level required, but Version 6 is recommended for workload balancing.

  To enable DB2 Connect EE for Sysplex workload routing, enter the SYSPLEX parameter in the correct position in the DCS directory. See *DB2 UDB Administration Guide Volume 1: Design and Implementation*.

For member routing, the data sharing group can run in goal mode or compatibility mode; incoming requests are distributed to members of the data sharing group based on their capacity

Using a generic LU name for the data sharing group (group-generic)

You can use VTAM's support for generic resources and have a generic 8-character name to represent a group of VTAM LU names. For ease of migration, we recommend choosing the LU name of the originating member of the data sharing group as the generic LU name. For example, generic name LUDB2A might represent three DB2 subsystems in the group whose real LU names are LUDB2AR, LUDB2B, and LUDB2C. (In this example, the “real” LU name for DB2A is renamed to LUDB2AR.)

Requesters outside the DB2 group set up their communications directories to refer to DB2 by a generic name, and VTAM selects the real DB2 LU name to be used by the requester. VTAM makes this choice based on the number of active DDF sessions or the result of a user-written VTAM or MVS workload manager exit routine.

After a requester is associated with a particular LU in the data sharing group, all future requests from that requester are sent to the same member of the data sharing group until all connections between the two LUs are terminated. In addition, if the connection between the client and server is enabled for two-phase commit processing, the mapping between the client and data sharing LUs are preserved until you issue the DB2 command RESET GENERICLU.

Comparing the SNA alternatives

Table 14 describes the differences between using the group-generic setup and member routing. “Communicating with a release of DB2 before Version 4” on page 116 describes how communication works if you do not use one of these methods, or if you are communicating with an earlier release of DB2 that does not support these methods.

Table 14 (Page 1 of 2). Comparing group-generic and member routing

Consideration	Group-generic	Member-routing
A requester sends SQL to a DB2 group	For any requesting LU, all SNA sessions must be directed to one and only one DB2 subsystem in the group. If the requester supports multiple concurrent threads, all threads must be directed to the same member of the DB2 group.	Each requester can establish sessions with one or more DB2 subsystems in the group. If the requester supports multiple concurrent threads, the threads need not be directed to a single member of the DB2 group.
DB2 group member sends an SQL request to another server.	Only one of the DB2 subsystems in the group is known to the remote server by the generic LU name at any given point in time. Other members of the DB2 group can send SQL requests to the remote server, but these members of the DB2 group must be known by their real VTAM LU names rather than the generic name. (For a DB2 remote server, a blank LUNAME column in SYSIBM.LUNAMES can serve this purpose.)	Any member of the DB2 group can issue SQL requests to the target server.

Table 14 (Page 2 of 2). Comparing group-generic and member routing

Consideration	Group-generic	Member-routing
DB2 server workload balancing	All SQL requests from a given LU are directed to only one DB2 server in the group. That DB2 server is chosen when the first VTAM session is established. After all sessions are terminated, VTAM can choose a different DB2 server for future communications, unless the sessions require two-phase commit. If sessions require two-phase commit, the same DB2 server must be used.	SQL requests can be sent to different DB2 servers in the group. The DB2 server informs the requester of the desired LU to service future SQL threads. DB2 uses special support from MVS's workload management to determine which DB2 subsystems have the greatest available capacity.
Reconnection after a communications failure	If the session uses two-phase commit, VTAM must always connect the requester to the DB2 LU chosen for the first session established by VTAM. If that DB2 LU is unavailable, a communication error is returned. If the session uses single-phase commit, VTAM can connect the requester to any DB2 LU in the group.	The requester can connect to any LU in the group. If resynchronization must be performed due to previous communication or system failures, the resynchronization request is routed to the DB2 LU involved in the failure. This activity is performed asynchronously, and does not prevent new sessions from being established with other members of the DB2 group.
Software requirements for partner	Existing DRDA products are sufficient.	DB2 for MVS/ESA Version 4 or DB2 Connect Version EE 5.2, or subsequent releases of either of those products.
Network definitions at partner site	For remote requesters, only the generic LU name must be defined. For remote servers, the generic LU name and the real LU names for each member of the group must be defined. See “Configuring to use group-generic processing” on page 125 for more information.	At the remote server, no special definitions are required. At the remote requester, definitions must be created for each LU name in the DB2 group. See “Configuring to use member routing” on page 122 for more information.

Recommendation: Use member routing unless your requester does not support it. It provides better workload balancing and two-phase connection support.

Planning for DB2 restart on another MVS

With VTAM , you can use a model application program definition for DB2. With a model definition, you use wild card characters for the application name (LU name). For example, you might have a model definition that looks like this:

```
LUDB2*  APPL   APPC=YES, ATNLOSS=ALL, ...
```

If you have a model definition like that shown above, and if one of your DB2s, say LUDB2A, fails on MVS1 and is restarted on MVS2, VTAM can use the above model definition for LUDB2A. For more information about using model application program definitions, see *VTAM for MVS/ESA Network Implementation Guide*.

Communicating with a release of DB2 before Version 4

A release of DB2 before Version 4 can communicate with a data sharing group, but not with the flexibility offered in Version 4 and subsequent releases. (There is no TCP/IP access to the group from any release before Version 5.) In releases before Version 4, administrative overhead is increased and there is no dynamic workload balancing.

As you can see from Table 15 on page 117, one of the major restrictions in having a data sharing group request data from a DB2 server before Version 4 is that only one member of the data sharing group can communicate with that server. All requests for the server have to be routed to the member of the data sharing group that is communicating with the server of the early release.

Table 15. DDF behavior when communicating with releases of DB2 before Version 4

Consideration	DDF behavior
A requester sends SQL to the DB2 group	Each requester must be directed to one of the available DB2 LU names to gain access to the DB2 server. If the requester supports multiple concurrent threads, all threads must be directed to the same member of the DB2 group. If the chosen LU name is not available, a communication failure is returned.
A DB2 group member sends SQL to another server	If the server is a DB2 subsystem, only one LU in the DB2 group is allowed to send SQL to the DB2 server. If two or more DB2 subsystems in the group require access to the remote data, the installation must have some mechanism for ensuring that all applications requiring access run on a single member of the DB2 group. If the chosen LU name is not available, a communication failure is returned. If the server is not a DB2 subsystem, each DB2 system in the group can send SQL statements to the server.
DB2 server workload balancing	Because DDF is not using any special support for data sharing, you are responsible for balancing the data sharing group workload. This is achieved by assigning some number of requesters to each DB2 server in the group. No dynamic workload balancing can occur.
Reconnection considerations after a communication failure	The requester is statically assigned to a single member of the DB2 group. If that member of the DB2 group is unavailable, a communication error is returned.
Network definitions at partner site	Only a single DB2 LU name must be defined at the requester. You are responsible for determining which LU name should be used at each requester to achieve acceptable workload balancing.

DRDA access through TCP/IP

A requester can define a server location name that is associated with many IP addresses; it is simpler to associate a server location name with a domain name rather than the many IP addresses for the systems in the Sysplex. To allow a requester to connect to any member in the group, each member uses the same DRDA port number; 446 is the port number we recommend. Only one member on a particular CPC can claim that DRDA port number for the purpose of serving incoming DRDA requests.

For resynchronization, however, each member has its own unique resync port number, which allows that member to handle outstanding resync requests, even if the member is started on another CPC.

Requirements

Requesters must support DRDA level 3. To be eligible to service TCP/IP requests, the members of the DB2 data sharing group must run on OS/390 Release 3 or subsequent releases.

Planning for availability

We recommend using a virtual IP address (VIPA) on each CPC to minimize the impact of a network controller failure. If you have more than one network controller on a single CPC, and if you route users to a particular controller's address, that becomes a single point of failure for the connection.

If you use the VIPA, end users can connect to the VIPA instead of the IP address associated with any single network controller. If a network controller fails, TCP/IP can automatically route the user's data to a different path.

To have DB2 send the virtual IP address to DRDA clients, you can use two methods:

- Specify the VIPA as the first entry for the TCP/IP HOME statement. For example:

  ```
   HOME
     12.23.34.45  VIPA1
     12.23.34.46  CTC1
     12.23.34.47  T3172A
     12.23.34.48  T3172B
  ```

- Specify the VIPA on the TCP/IP PRIMARYINTERFACE statement. This overrides the ordering specified on the HOME statement. For example:

  ```
  PRIMARYINTERFACE VIPA1
  ```

Planning for resynchronization after failure

All DB2 members must have the same DRDA port number for incoming DRDA SQL requests. If DB2 is automatically restarted on another CPC, the TCP/IP on that system must be able to allow that DB2 to use the DRDA port. The best way to do this is to assign the DRDA port to every member that could conceivably start on a particular system. On each system, you can replicate the TCP/IP PORT statement shown here. By explicitly assigning these port numbers, you can prevent some other program from using DB2's port number.

```
PORT
⋮
  446 TCP OMVS
```

In OS/390 Release 4, you can specify the PORT statement as follows, which reserves the DRDA port just for the DB2 DDF address space:

```
PORT
⋮
  446 TCP DB1GDIST
  446 TCP DB2GDIST
  446 TCP DB3GDIST
  446 TCP DB4GDIST
```

Only one of the DB2 members associated with a given DRDA port number is selected by TCP/IP to process the incoming requests. The other member can fulfill any resynchronization responsibilities it has using its unique resync port number. However, don't forget to register each member in the domain name server as described in “Registering names in the domain name server” on page 129.

The chosen DB2 member receives all of the DRDA server's workload for that TCP/IP instance, which leaves the other DB2 members with no TCP/IP server threads for DRDA. This is transparent to the DRDA clients if the DB2 member processing the TCP/IP requests doesn't reach the MAX REMOTE CONNECTED thread limit. If the MAX REMOTE CONNECTED limit is reached, the connection request of the DRDA client is rejected.

After you resolve a failure situation, it is a good idea to move the DB2 member to its original IP address.

TCP/IP workload balancing for DB2 data sharing

Consider the following methods for balancing the workload of DRDA clients connecting to a DB2 data sharing group:

- Enter the IP address of the desired DB2 member in each client's TCP/IP configuration. This approach has some drawbacks:
  - Balancing client workloads in this fashion is very labor-intensive.
  - The DB2 member's IP address changes if it is restarted on another CPC, so this approach doesn't work if your automatic restart configuration restarts DB2 on another CPC.
  - It doesn't adjust to changes in the work load.
- Take advantage of the Sysplex workload balancing information built into the DRDA architecture. If the requester is DB2 Version 5 or DB2 Connect Enterprise Edition Version 5.2, or subsequent releases of those products, it can use this workload balancing information to spread connections across the available members in the data sharing group.

  To enable DB2 Connect EE for Sysplex workload routing, enter the SYSPLEX parameter in the correct position in the DCS directory. See the README file shipped with the enabled version of DB2 Connect for more information.

  Other DRDA application requesters might also provide this capability; refer to the documentation associated with your DRDA requester product to determine whether this approach can be used. If you cannot use DRDA workload balancing, use the domain name server to balance the workload.
- Use the domain name server (DNS) to balance the workload.

  This approach, sometimes called *connection optimization for a Sysplex*, is recommended for single-user workstation connections, such as DB2 Connect Personal Edition Version 5.1 or subsequent releases. It's also useful for directing the first connection through a gateway.

  The steps for using the DNS to balance the workload are, in summary:

  1. Make sure that all members of the data sharing group are running in workload manager goal mode.
  2. Set up your DNS with a bootfile entry for the Sysplex that includes the cluster keyword. The high level qualifier for the bootfile entry must match the Sysplex name, which you can obtain from the MVS COUPLE*xx* data set. An example bootfile entry is:

```
;  /etc/named.boot
;
;       boot file for name server
;
;
; type        domain                 source file or host
;
directory     /etc/dnsdata
primary       sysplex1.ibm.com       named.wlm.for     cluster
primary       183.65.121.in-addr.arpa named.rev
primary       0.0.127.in-addr.arpa   named.lbk
cache         .                      named.ca
options query-log
```

For more information about setting up the DNS to handle Sysplex domain workload balancing, see *OS/390 eNetwork Communications Server: IP Configuration*.

3. WLM registers the DB2 group and server domain names to the domain name server. In OS/390 Release 4 (with APAR PQ06340), and subsequent releases, WLM provides information to the domain name server that identifies which IP address has the most capacity. For more information, see “Registering names in the domain name server” on page 129.

Recommendation: For the highest availability and workload balancing, use DNS workload balancing and DRDA Sysplex routing together where possible. Because the Sysplex routing information is returned only after the first connection is made to the host, the DNS can be used to route that first connection to an available member with the most capacity. After that first connection is made, the client can use DRDA Sysplex routing to choose where subsequent connections are made.

Defining a DB2 data sharing group in an SNA network

When you have decided whether to use group-generic or member routing for a particular partner, you must specify communications definitions for the group and for any partner LUs. To illustrate, assume a data sharing group whose original member is already defined to the partner LUs shown in Figure 21 on page 121. The data sharing group has a location name of DB2A (it is using the original member's location name so that existing applications do not have to be changed). As described in “Communicating with a release of DB2 before Version 4” on page 116, requesters of DB2A's data can access its data only through the DB2 defined as LUDB2A. If LUDB2A is unable to service requests, the requesters have to change their communication definitions to connect to a different member of the data sharing group.

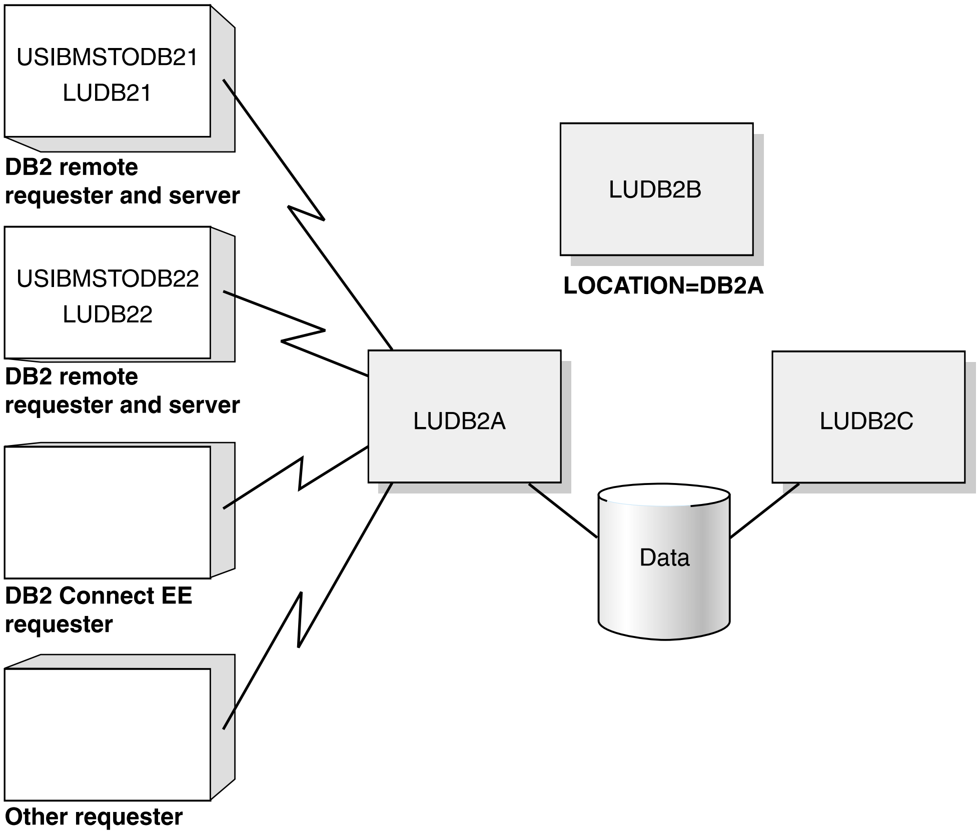


Figure 21. Example configuration before distributed group support is enabled. Access is limited to a single member of the data sharing group.

Example configuration with enhanced distributed support

Now assume that you want the configuration shown in Figure 22 on page 122. In that configuration, both remote DB2s and DB2 Connect EE are implementing member routing; the other DRDA requester is using group-generic processing because it does not have support for member routing.

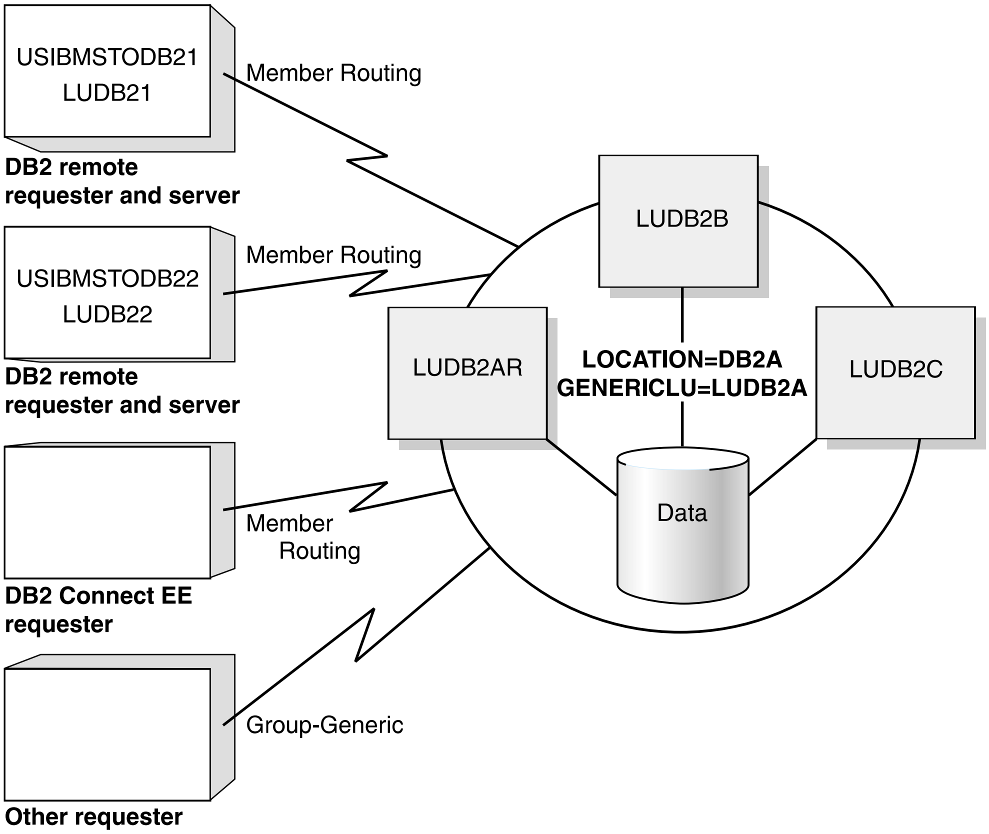


Figure 22. Configuration after enabling group-generic or member routing support. All members of the group can potentially service requests.

Configuring to use member routing

This section describes how to update the CDBs of DB2 requesters to use member routing as shown in Figure 23.

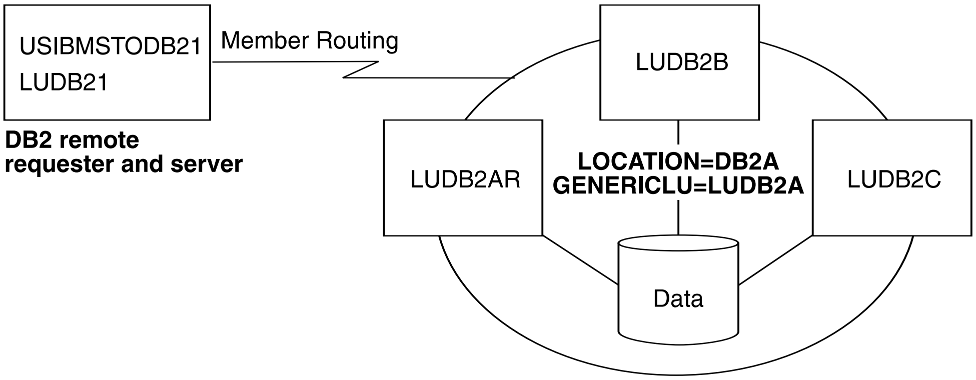


Figure 23. DB2s using member routing.

Communications database for a DB2 requester

To indicate that you want DB2 to use a list of LU names for a data sharing group, you must use a token, which we call the link name, that corresponds a list of LUs in the SYSIBM.LULIST table with a single LU definition in the SYSIBM.LUNAMES table.

A DB2 requester's SYSIBM.LUNAMES table: Assume that you choose a link name of LUGROUP. You must enter that name in the SYSIBM.LUNAMES table of the requester. The real LU names must not be included in SYSIBM.LUNAMES. To enable member routing, USIBMSTODB21's SYSIBM.LUNAMES table must look like Table 16. (The blank row is optional, but recommended.)

Table 16. A DB2 requester's SYSIBM.LUNAMES table to enable DDF support

LUNAME	...	GENERIC
LUGROUP		
(blank)		

A DB2 requester's SYSIBM.LOCATIONS table: Now, you must map the data sharing location name to the link name. USIBMSTODB21's SYSIBM.LOCATIONS table looks like Table 17.

Table 17. A DB2 requester's LOCATIONS table for member routing. Not all columns are shown.

LOCATION	LINKNAME	TPN
DB2A	LUGROUP	

General-use Programming Interface

Use the following SQL statements to populate SYSIBM.LUNAMES and SYSIBM.LOCATIONS:

1. This statement inserts the link name (LUGROUP) into the SYSIBM.LUNAMES table:

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

2. The following statement updates the LINKNAME column in SYSIBM.LOCATIONS to point to LUGROUP:

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

3. The following statement must be used to delete the existing LU name from the SYSIBM.LUNAMES table. You enter the real LU names for the DB2 members in the SYSIBM.LULIST table in a later step.

   ```
   DELETE FROM SYSIBM.LUNAMES
     WHERE LUNAME='LUDB2A';
   ```

End of General-use Programming Interface

A DB2 requester's SYSIBM.LULIST table: When DB2 acts as a requester, DB2 makes use of a communications database table called SYSIBM.LULIST. This table makes it possible for you to specify more than one LUNAME value for any given server. Populate this table only if DB2 is acting as a requester of data that resides in a data sharing group. You can specify a subset of members in list; those members that are not included in the list are not considered as servers for member routing.

The table is used by a DB2 requester as follows:

- When access is requested to a particular remote location, the SYSIBM.LOCATIONS table is searched to find a matching row. The LINKNAME column of that row identifies the corresponding rows in the SYSIBM.LUNAMES table.
- If one or more rows exist in the SYSIBM.LULIST table for the specified LINKNAME, the LUNAME values of the SYSIBM.LULIST rows represent the available network destinations (LUNAMEs) for the LOCATION. The values of the columns in the SYSIBM.LUNAMES row (SECURITY_IN, SECURITY_OUT, USERNAMES, and so on) apply to each of these destinations.

  Similarly, the rows (if any) in the SYSIBM.USERNAMES and SYSIBM.MODESELECT tables apply to each of the destinations. A blank row in the requester's SYSIBM.LUNAMES table lets DB2 use without error whatever list of LUs the data sharing group server returns to the requester when the requester makes a request to the data sharing group.
- If no matching row exists in SYSIBM.LULIST, the LUNAME column of the SYSIBM.LUNAMES table provides the VTAM LUNAME of the remote LOCATION.

SYSIBM.LULIST has the following columns:

LINKNAME CHAR(8)
The value of the LINKNAME column of SYSIBM.LOCATIONS with which this row is associated. This is also a value in column LUNAME for some row in table SYSIBM.LUNAMES. The values of the other columns of that row of LUNAMES (SECURITY_IN, SECURITY_OUT, USERNAMES, and so on) apply to the LU identified in column LUNAME of this row of SYSIBM.LULIST.

LUNAME CHAR(8)
The VTAM LU name of the remote database system. This LUNAME must not exist in the LUNAME column of SYSIBM.LUNAMES.

A DB2 requester, such as USIBMSTODB21, wants a SYSIBM.LULIST table that looks like Table 18 on page 125.

Table 18. A DB2 requester's SYSIBM.LULIST table

LINKNAME	LUNAME
LUGROUP	LUDB2AR
LUGROUP	LUDB2B
LUGROUP	LUDB2C

General-use Programming Interface

Use the following SQL statements to associate the LUNAMES of the all members of the group (LUDB2AR, LUDB2B, and LUDB2C) with DB2A (the location name of the data sharing group) by creating a new LUNAME (LUGROUP) that is composed of a list of the real LUNAMES.

```
INSERT INTO SYSIBM.LULIST
   VALUES ('LUGROUP', 'LUDB2AR');

INSERT INTO SYSIBM.LULIST
   VALUES ('LUGROUP', 'LUDB2B');

INSERT INTO SYSIBM.LULIST
   VALUES ('LUGROUP', 'LUDB2C');
```

End of General-use Programming Interface

When updates to SYSIBM.LULIST take effect: Changes to SYSIBM.LULIST take effect as follows

- If DDF has not yet tried to communicate with a particular location, rows added to LULIST take effect when DDF attempts to communicate with the remote location.
- If DDF has already attempted communication with a particular location, rows added to LULIST take effect the next time DDF is started.

Configuring to use group-generic processing

This section describes how to set up the data sharing group to service requests from a DRDA requester, as shown in Figure 24 on page 126. “DB2 for OS/390 remote requester and remote server definitions” on page 128 describes how you would set up a DB2 for OS/390 to request and serve using the group-generic setup, but in most cases, you will want to use member routing when both partners are DB2 for OS/390 (or MVS/ESA).

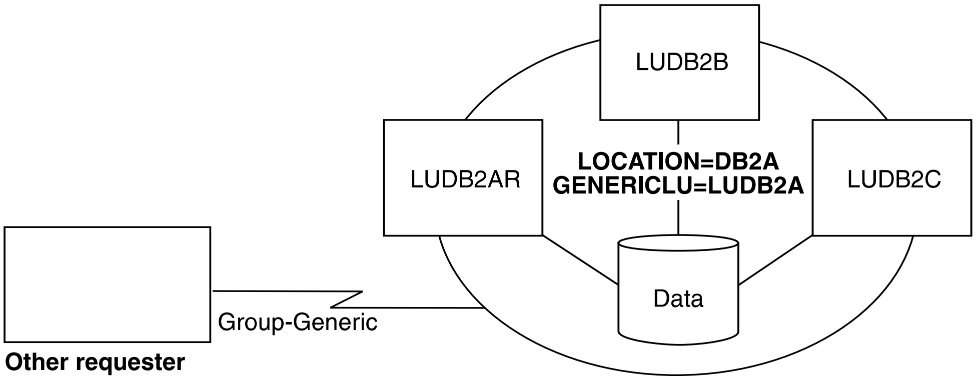


Figure 24. DRDA Requester using Group-generic processing.

To set up the group for group-generic processing, specify the generic LU name on an installation panel for each member of the group. If the group is to be used as a server (the most likely configuration) then you must specify Y in column GENERIC for the data sharing group's SYSIBM.LUNAMES table entry for requesters that use the generic name to connect to the group. (A blank LUNAME will do fine for this purpose.)

You must also include information in the coupling facility for support of VTAM generic resources (the ISTGENERIC structure). For more information about using VTAM's generic resources, see *VTAM for MVS/ESA Network Implementation Guide*. To calculate storage for that structure, you also need information from *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide*.

Specifying the generic LU name for the group at installation time

To specify the 1- to 8-character generic LU name for the data sharing group, use the DB2 GENERIC NAME field on the DSNTIPR installation panel.

To avoid extensive changes to the communications directories of requesting systems, **we recommend that you choose the existing LU name of the originating member of the data sharing group**. Before starting communications, you must change the real LU name of this originating member by changing the value in the VTAM APPL statement and the LUNAME value in the BSDS. In our sample configuration, the originating member of the group has changed its LU name from LUDB2A to LUDB2AR.

Each member of a data sharing group must choose the same name. The generic LU name is stored in each member's BSDS.

If this DB2 is not a member of the data sharing group but is still running as part of an MVS Sysplex, you can still choose a generic LU name for the DB2. This might be useful, for example, during a transition when network names are being changed.

Updating the group's SYSIBM.LUNAMES table

Use the GENERIC column of the SYSIBM.LUNAMES communications database table to specify whether the VTAM generic LU name or the DB2 real LU name is used by DB2 to identify itself to a given remote server. **This column is used only when DB2 is initiating contact with a given partner LU.**

This column does not determine whether you use group-generic or member-specific access for a partner. The rows, or lack of rows, in SYSIBM.LULIST determine whether group-generic or member routing access is used for a particular partner.

The description of the GENERIC column is as follows:

GENERIC CHAR(1)

The values have the following meanings:

N or blank The real VTAM LU name of the DB2 subsystem is used for CNOS operations and SQL requests to the partner LU identified in this row.

Y The VTAM generic LU name is used for CNOS operations and SQL requests to the partner LU when DB2 initiates processing. The partner must be able to recognize the generic LU name. If one member of the data sharing group is already using the generic LU name for a given partner, another member of the group cannot, and the real LU name is used instead. This means if the data sharing group is requesting data from a system that only accepts generic LU names, all requests must be routed through one member of the data sharing group.

If the partner starts CNOS processing first, VTAM uses the name with which the partner connected, whether the real LU name or the generic LU name. Because the behavior is not always predictable, it is best if the system that is servicing requests from the data sharing group can accept either the generic LU name or the real LU name when group-generic processing is used.

The GENERIC column is ignored if the field DB2 GENERIC NAME, on installation panel DSNTIPR, is blank.

DB2A wants a SYSIBM.LUNAMES table that looks like Table 19. A value of 'Y' is inserted into the GENERIC column for all partners that use a generic LU name to communicate with the data sharing group.

Table 19. The data sharing group's SYSIBM.LUNAMES table for group-generic processing

LUNAME	...	GENERIC
(blanks)		Y

General-use Programming Interface

The following statement can be used to update the SYSIBM.LUNAMES table of the data sharing group:

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

End of General-use Programming Interface

DB2 for OS/390 remote requester and remote server definitions

Although we recommend using member routing when both partners are DB2 for OS/390, this section is included in case you must use group-generic processing.

For a DB2 **requester,** use the generic LU name to access a DB2 data sharing group. Place the generic LU name of the DB2 group in the LUNAME and LINKNAME columns of the CDB entries associated with the DB2 group's location. If you use the original member's LU name as the generic LU name, then no changes are necessary.

For a remote DB2 **server**, include the generic LU name but also include all the LU names of all members of the requesting DB2 data sharing group. You can use the default row (blanks for LUNAME) in the SYSIBM.LUNAMES table as we have shown here:

Table 20. A remote DB2 server's SYSIBM.LUNAMES table

LUNAME	...	GENERIC
(blank)		

Switching from group-generic to member routing

If you are currently using group-generic for your connections between DB2 for OS/390 (or MVS/ESA) partners, and you want to switch to member routing because of the additional benefits it brings, you must do the following procedure. This procedure is necessary because of two-phase commit connections that have a real LU name or generic LU name registered in the coupling facility. To switch which name gets used, you must unregister the name.

1. Shut down the network connections between DB2 and the partner system that you are unregistering from the coupling facility. Some ways to do this include using the VTAM command VARY NET,INACT,ID=*luname*, or entering the DB2 command STOP DDF.
2. From an active member of the data sharing group, enter the command RESET GENERICLU *luname*.
3. Modify the requester system's CDB to ensure that member routing is used. You do this by populating the SYSIBM.LULIST table, as described in “Configuring to use member routing” on page 122. Make sure the requesting DB2's GENERIC column of SYSIBM.LUNAMES contains N or blank for the row to this data sharing group server.
4. Enable network connections.

Defining a DB2 data sharing group in a TCP/IP network

This section gives an overview of one possible way to configure a data sharing group to operate in a TCP/IP network. See your network administrator and the information in Section 2 of *DB2 Installation Guide* to work out the details for your specific configuration.

Example TCP/IP configuration

The configuration in Figure 25 shows two DB2 for OS/390 requesters and one other DRDA requester. When a requesting application uses an SQL statement that specifies DB2A as the location name, the requester uses the LINKNAME column of SYSIBM.LOCATIONS to find out whether or not it is using TCP/IP to connect to the server known as DB2A.

Because the LINKNAME value appears in the SYSIBM.IPNAMES table, DB2 knows that it is using TCP/IP to send the DRDA request to the server. (The requester is using a domain name to represent the server. That way, only the domain name server must be updated with specific IP addresses.) The application is directed to the member of the group that is best able to service the request based on its current work load. For more information about workload balancing, see “TCP/IP workload balancing for DB2 data sharing” on page 119.

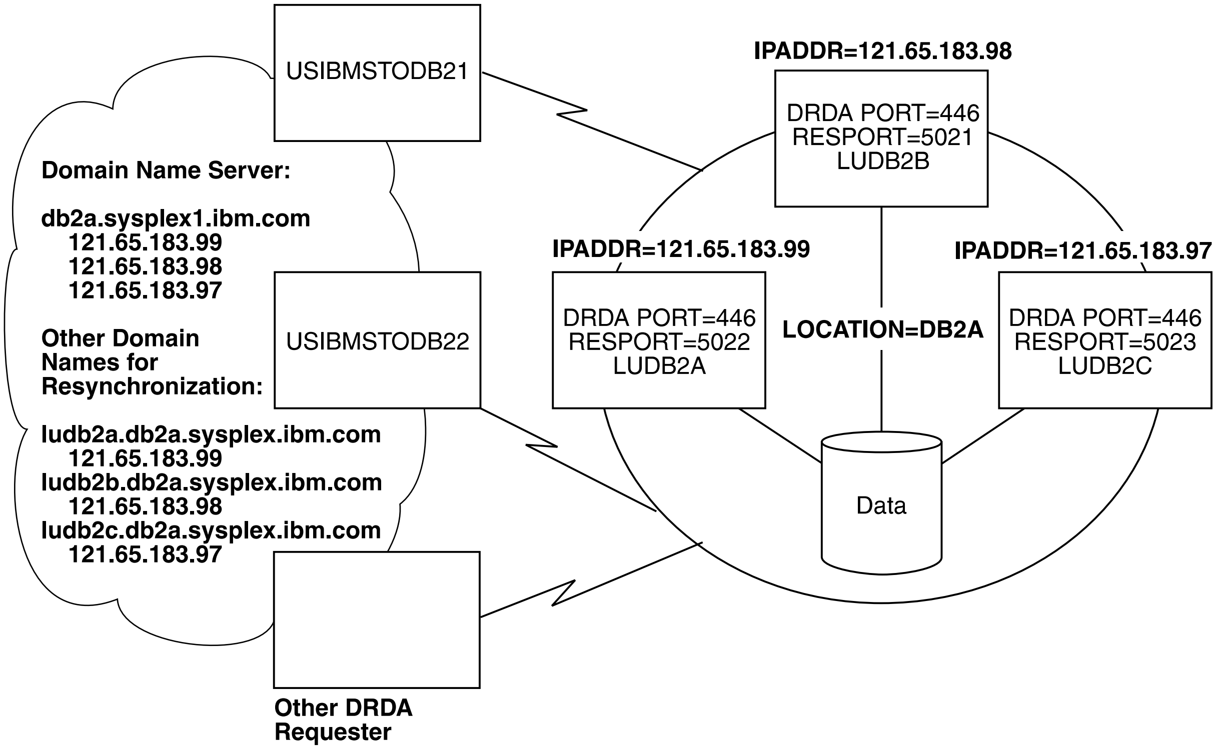


Figure 25. Example TCP/IP network configuration. Applications use the location name to direct requests to the group. The network addresses of the various members of the group are provided in the requester communications definitions. Use a domain name to simplify this process.

Registering names in the domain name server

To take advantage of workload balancing and correct resynchronization for two-phase commit processing if a DB2 is restarted on another MVS, you must register DB2 names in the domain name server (DNS). (In OS/390 Release 4 with APAR PQ06340 , workload manager registers these names for you automatically when you use configure the DNS for Sysplex routing. This configuration is called connection optimization in a Sysplex domain, as described in *OS/390 eNetwork Communications Server: IP Configuration*.) Register the following names:

- location.sysplex.domainname

This is the group name that represents the entire data sharing group. It contains the DB2 location name and the sysplex name (from the COUPLE*xx* data set, among other places). The domain name comes from the socket calls gethostid for the host address, and gethostbyaddr for the host name. DB2 removes the high level qualifier returned by the gethostbyaddr call and replaces it with the location name and sysplex name. You can verify that you have the correct name to register in the DNS by starting DDF. The name you must register appears in the 'domain name' field of the DSNL004I message.

- luname.location.sysplex.domainname

  This is the server name. Register one of these names for each member of the data sharing group. It includes the real LU name for the member. This name is used by DB2 to resolve indoubt threads.

Server definitions

Defining a data sharing group as a TCP/IP server does not require any CDB definitions.

Specify a generic LU name if you use RACF passtickets

If you use RACF PassTickets for security, define a generic LU name for the data sharing group by using the DB2 GENERIC NAME field of installation panel DSNTIPR, or use the change log inventory utility to do that. The generic LU name is used to generate the RACF PassTicket application names. See Section 3 (Volume 1) of *DB2 Administration Guide* for more information about using RACF PassTickets.

Specify the port numbers for the group

Use installation panel DSNTIP5 or the change log inventory utility to specify a DRDA port number on each member that accepts TCP/IP requests. Specify the same DRDA port number on all members of the group.

To resynchronize two-phase commit units of work, specify a unique RESYNC port number for each member of the group. (RESYNC port numbers must be unique within the Sysplex.) See *DB2 Utility Guide and Reference* for more information about the change log inventory utility.

The network administrator must register the port numbers on the TCP/IP that is associated with each member's MVS system.

Remote requester definitions

This section describes the CDB entries and the BSDS entries needed at the requesting DB2 for OS/390.

Remote requester CDB entries

To access the DB2A data sharing group server, the DB2 for OS/390 requester has a SYSIBM.LOCATIONS table that looks like Table 21 on page 131. The example specifies a generic LU name for the link name, because that generic LU name is used to generate the application name for the RACF PassTicket.

If you do not use RACF PassTickets, and you have no SNA communication at all, you can use any name as the link name; the link name relates rows in SYSIBM.LOCATIONS to rows in SYSIBM.IPNAMES.

Table 21. USIBMSTODB21's LOCATIONS table. Not all columns are shown.

LOCATION	LINKNAME	PORT
db2a	GENERICLU	446

General-use Programming Interface

Use the following SQL statement to populate SYSIBM.LOCATIONS.

```
INSERT INTO SYSIBM.LOCATIONS
            (LOCATION, LINKNAME, PORT)
      VALUES ('db2a', 'GENERICLU', '446');
```

End of General-use Programming Interface

If you prefer, you can use a case-sensitive *service name* instead of a hard-coded port number in the PORT column. This service name is another way to refer to a port number. For more information, see the customization information for the level of TCP/IP configuration that you are using.

The DB2 for OS/390 requester has a SYSIBM.IPNAMES table that looks like Table 22.

Table 22. SYSIBM.IPNAMES at the DB2 for OS/390 requester. Not all columns are shown.

LINKNAME	SECURITY_OUT	USERNAMES	IPADDRESS
GENERICLU	R	blank	db2a.sysplex1.ibm.com

General-use Programming Interface

Use the following SQL statement to populate SYSIBM.IPNAMES:

```
INSERT INTO SYSIBM.IPNAMES
            (LINKNAME, SECURITY_OUT, USERNAMES, IPADDR)
      VALUES ('GENERICLU', 'R', ' ', 'db2a.sysplex1.ibm.com');
```

End of General-use Programming Interface

Bootstrap data set entries

Use installation panel DSNTIP5 or the change log inventory utility to enter a DRDA port and resynchronization port into the BSDS. If you do not enter values for these ports, you cannot use TCP/IP.

The data sharing group as a requester

To use a data sharing group as a TCP/IP requester, you need no special configuration other than that shown in “Remote requester definitions” on page 130. If you use RACF PassTickets, you must use the name (either generic LU name or real LU name) that is defined at that server. Be aware, however, that only one member of the data sharing group can use the generic LU name as a requester.

Excluding a member from processing remote requests

Transparently to end users, you can exclude one or more members from DDF server processing while still letting it make DDF requests. To exclude a member from DDF server processing, set the MAX REMOTE ACTIVE option of installation panel DSNTIPE to zero for that member. The effects of setting this parameter to zero are:

- DB2 does not register the member's LU name with the VTAM generic LU name during DDF startup. Connections that use the generic LU name are directed to those members for which MAX REMOTE ACTIVE is greater than zero.
- DB2 does not register the member to MVS workload manager for member routing. The member can continue to use workload manager for setting priorities on work, but its name is not included in the list of available LUs that is sent to remote sites. Therefore, DDF server work is never routed to that member.
- DB2 does not listen on the DRDA SQL port, which means that SQL TCP/IP requests are accepted only by members for which MAX REMOTE ACTIVE is greater than zero.

Using the change log inventory utility to update the BSDS

Use the change log inventory utility (DSNJU003) to update the following information in the BSDS related to distributed data processing:

- generic LU name
- resynchronization port (for TCP/IP)
- DRDA port (for TCP/IP)

If you change the generic LU name or DRDA port for one member of the data sharing group, you must change it for all members. This requires that you stop and restart DDF to pick up the change.

See Section 2 of *DB2 Utility Guide and Reference* for more information about using DSNJU003.

Chapter 6. Operating with data sharing

Most data sharing operations are accomplished by means of commands to the licensed program DB2 for OS/390. This chapter describes:

Entering commands

This section describes the following:

Routing commands

Operations on an individual member of a data sharing group can be controlled from any MVS console by entering commands prefixed by the appropriate *command prefix*. For example, with the appropriate choice of command prefix, you can start a DB2 statistics trace on member DB1G by entering this command at any console in the Sysplex:

```
-DB1G START TRACE (STAT)
```

This routing of commands requires that the command prefix scope is registered as S or X on the IEFSSN*xx* parmlib member. For specifications of command prefixes, see "Registering command prefixes and group attachment name" on page 68.

Operations on certain objects are controlled by commands or command options that affect an entire group. These, also, can be entered from any MVS console. For example, and again with the appropriate choice of command prefixes and assuming DB1G to be active, you can start database XYZ by entering this command at any MVS console of the Sysplex:

```
-DB1G START DATABASE (XYZ)
```

Command scope

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

Many commands used in a data sharing environment affect only the DB2 for which they are issued. For example, a DISPLAY THREAD command displays only those threads that exist for 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 affects all members of the group. For example, a STOP DATABASE command, issued from any member of the group, stops that database for all members of the group. See *DB2 Command Reference* for information about the scope of each command.

Entering commands from an application program

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

Authorizing commands

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.

Receiving messages

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.

Starting and stopping DB2

To start members of a data sharing group, you must enter a START DB2 command for each subsystem in the group. If this is the first startup of 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 either issue the command from the MVS system you want to start DB2 on or route the command to that MVS. Here is an example of routing the command to the MVS on which DB1G is to be started:

```
ROUTE MVS1,-DB1G START DB2
```

After DB2 is started, all other commands can be issued from any MVS in the Sysplex, and the commands are routed to the appropriate DB2 subsystem.

Stopping DB2

Stop individual DB2 members using the STOP DB2 command as described in Chapter 2 of *DB2 Command Reference*. 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.

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

Normal Shutdown: Table 23 summarizes the information that you see after a normal termination:

Table 23. States of structures and connections after normal DB2 termination

Structure	Connections	Structure
SCA	None	Allocated
Lock	Failed-persistent	Allocated
Note: If a given DB2 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.		
Group buffer pools	None with CASTOUT(YES); Failed-persistent with CASTOUT(NO)	Unallocated with CASTOUT(YES); Allocated with CASTOUT(NO)
Note: If castout failure occurs during shutdown, group buffer pool connections show as failed-persistent, even though DB2 terminates normally.		

Abnormal shutdown: Table 23 summarizes the information that you see after an abnormal termination:

Table 24. States of structures and connections after abnormal DB2 termination

Structure	Connections	Structure
SCA	None	Allocated
Lock	Failed-persistent	Allocated
Note: If a given DB2 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.		
Group buffer pools	None with CASTOUT(YES); Failed-persistent with CASTOUT(NO)	Unallocated with CASTOUT(YES); Allocated with CASTOUT(NO)

Submitting work to be processed

The methods you use to submit work need not change for data sharing. However, you might find it to your advantage to use the group attachment name to direct jobs that are not submitted through CICS or IMS. This section describes more about the group attachment name and how it works.

Running CICS and IMS applications

There is no change in the process of running CICS and IMS applications for data sharing. You cannot use the group attachment for CICS and IMS applications because those transaction managers must be aware of the particular DB2 subsystem to which they are attached so they can resolve indoubt units of recovery in the event of a failure. See Section 4 (Volume 1) of *DB2 Administration Guide* for more information about running applications.

Using the group attachment name

Utilities and applications that use TSO, batch, or the RRSAF to connect to DB2 have two methods for specifying the DB2 to which they want to connect. The first method is to specify the particular subsystem name in the job for TSO and batch, RRSAF, and on an explicit CONNECT for CAF. The second method is to use the *group attachment name* instead of a specific subsystem name.

The group attachment name acts as a generic name for the DB2 subsystems in a data sharing group. It substitutes for the DB2 subsystem name running on the MVS the job was submitted from.

By using the group attachment name, the TSO user or batch job does not have to be sensitive to the particular subsystem, which makes it easier to move jobs around the Sysplex as needed.

The group attachment name is specified at DB2 installation on the DSNTIPK installation panel, which places the name in the IEFSSN*xx* member and in the DSNHDECP load module for the group. The group attachment name appears on the output from the command DISPLAY GROUP.

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

DB2I (DB2 Interactive) can also use the group attachment name.

For more information about submitting applications, see *DB2 Application Programming and SQL Guide*.

Submitting online utilities

When you submit a utility job, you must specify the name of the DB2 subsystem to which the utility is to attach, or you can specify the group attachment name. For example, the EXEC statement might look like the following:

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

Establishing affinity: If you don't use the group attachment name, the utility job must run on the MVS system where the specified DB2 subsystem is running. To do that, you must be sure that MVS runs the utility job on the appropriate MVS system. There are several MVS installation-specific ways to make sure this happens. These include:

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

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

Where *cccc* is the JES2 name. You can specify an asterisk (SYSAFF=*) to indicate that the job should run on the system from which it was submitted.

- For JES3 systems, use the following JCL statement:

  ```
  //*MAIN SYSTEM=(main-name)
  ```

  Where *main-name* is the JES3 name.

OS/390 MVS JCL Reference describes the above JCL statements. Your installation might have other mechanisms for controlling where batch jobs run, such as by using job classes.

Stopping and restarting utilities: Kunioka-Weiss ***add soft514** 10/08/99 *** paquet ***

In a data sharing environment, active utilities can be stopped (with the TERM UTILITY command) only on the DB2 subsystem on which they are active. You can terminate a stopped utility from any active member of the group. If a DB2 fails while a utility is executing, you must restart that DB2 on either the same or another MVS before stopping the utility. For remote site recovery from a disaster at the local site, utilities that were active at the local site can be terminated from any restarted member of the group at the remote site.

You can restart a utility on any member, but that member must be able to access all required data sets. We recommend that you define all work data sets used by utilities on shared DASD. Use the same utility (UID) to restart the utility. That UID is unique within a data sharing group. However, if a DB2 fails while a utility is executing, you must restart that DB2 on either the same or another MVS before restarting the utility.

Altering utilities: In a data sharing environment, the REORG utility can be altered (with the ALTER UTILITY command) only on the DB2 subsystem on which it is active.

Submitting stand-alone utilities

DB2 stand-alone utilities (such as DSN1COPY) run as MVS jobs that have no direct connection to DB2 services. Therefore, a DB2 system has no indication that one of these utilities is running.

In a data sharing environment, if a table space has inter-system R/W interest, then 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, then you must stop or quiesce the table space. Also, the data must not be in the RECP or GRECP status nor have any logical page list (LPL) entries. Use DISPLAY DATABASE with the RESTRICT option to find out if there are exception statuses for a given table space or index.

Monitoring the group

This section describes the commands you can use to do the following tasks:

“Obtaining information about the group”
“Obtaining information about structures and policies” on page 139
“Obtaining information about group buffer pools” on page 141
“Monitoring databases” on page 142

The section on monitoring databases includes information about the logical page list and how to clear entries from that list. It also includes information about detecting retained locks.

Obtaining information about the group

The information under this heading, up to “Obtaining information about structures and policies” on page 139 is General-use Programming Interface and Associated Guidance Information, as defined in Appendix B, “Notices” on page 265.

To obtain general information about all members of a particular group, use the DISPLAY GROUP command, shown here:

```
-DB1G DISPLAY GROUP
```

The command can be issued from any active member of the group, and displays the following output:

```
- DSN7100I -DB1G DSN7GCMD
- *** BEGIN DISPLAY OF GROUP(DSNDB0G ) GROUPLEVEL(610)
-                                      GROUP ATTACH NAME(DB0G)
- --------------------------------------------------------------------
- DB2                                    DB2 SYSTEM    IRLM
- MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
- -------- --- ----   --------  -------- --- --------  ----   --------
- DB1G       1 DB1G   -DB1G     ACTIVE   610 MVS1      DJ1G   DB1GIRLM
- DB2G       2 DB2G   -DB2G     ACTIVE   610 MVS2      DJ2G   DB2GIRLM
- DB3G       3 DB3G   -DB3G     ACTIVE   610 MVS3      DJ3G   DB3GIRLM
- DB4G       4 DB4G   -DB4G     FAILED   610 MVS4      DJ4G   DB4GIRLM
- --------------------------------------------------------------------
- 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(DSNDB0G )
- DSN9022I -DB1G DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 26. Output of DISPLAY GROUP Command

Figure 26 shows the following information:

- The DB2 group name and group release level, and the member names and release levels
- The group attachment name
- The IRLM subsystem names to which members are connected. For more information about the IRLM data sharing group, use the MVS command F *irlmproc*,STATUS,ALLI.
- The command prefix for each member

- The status of each member (ACTIVE, QUIESCED with or without additional conditions, or FAILED)
- The MVS system names where the member is running, or was last running in cases when the member status is not active
- The procedure names of the connected IRLMs
- The SCA structure size in kilobytes and the percentage currently in use
- Lock structure size and the percentage of the structure size in use. This percentage is based on the maximum percentage used of either the lock table or the modify lock list.

  The display also shows the following:

  - The maximum number of lock entries possible for the lock table and how many of those lock entries are currently in use. This number is an approximate value.
  - The maximum number of modify lock list entries and how many of those list entries are currently in use.

  For more information about the lock table and the list of modify locks, see "Avoid false contention" on page 207.

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. See Figure 36 on page 191 for an example.

Obtaining information about structures and policies

Use the MVS command D XCF,STR to display information about coupling facility structures and policy information. For more information about the D XCF,STR command, see *OS/390 MVS System Commands*.

Displaying all structures

The following command displays summary information about all structures:

```
D XCF,STR
```

The command produces the following output:

```
D XCF,STR
IXC359I  15.57.52  DISPLAY XCF
STRNAME            ALLOCATION TIME   STATUS
DSNCAT_GBP0      04/09/1999 15:57:47 DUPLEXING REBUILD NEW STRUCTURE
                                     DUPLEXING REBUILD
                                     REBUILD PHASE: DUPLEX ESTABLISHED

DSNCAT_GBP0      04/09/1999 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     04/09/1999 15:26:16 ALLOCATED
DSNCAT_SCA       04/09/1999 15:26:13 ALLOCATED
ISTGENERIC       04/09/1999 15:11:09 ALLOCATED
IXCLINKS              --       --    NOT ALLOCATED
LOCK2                 --       --    NOT ALLOCATED
RRSSTRUCT1            --       --    NOT ALLOCATED
```

Figure 27. Ouput from D XCF,STR command

Displaying information about specific structures

You can also display more 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 ESTABLISHEDPOLICY 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_V61A         02 00020001 UTEC277  V61ADBM1 002F ACTIVE NEW,OLD
 DB2_V61B         01 00010001 UTEC277  V61BDBM1 0033 ACTIVE NEW,OLD
```

Figure 28. Ouput from D XCF,STR,STRNAME Command

Obtaining information about group buffer pools

To obtain information about group buffer pools, you can use the MVS command D XCF,STR as described in “Displaying information about specific structures” on page 140. However, DB2 provides a DISPLAY GROUPBUFFERPOOL command that is useful for displaying statistical information about group buffer pool use.

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, there is only one set of connections for both instances of the group buffer pool. For example, if there are 3 connections to duplexed structure GBP0, there are just 3 connections, not 6 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.

See “Using the DISPLAY GROUPBUFFERPOOL command” on page 245 for more information about the DISPLAY GROUPBUFFERPOOL command.

Monitoring databases

Data sharing introduces the GRECP and LPL statuses. These statuses can appear on the output from the DISPLAY DATABASE command.

GRECP “Group buffer pool recovery pending.” The group buffer pool failed, 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 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 DB2 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 DASD because of a transient DASD problem.

For page sets or partitions that have LPL or GRECP status and aren't 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. For more information about removing LPL status, see “Recovering pages on the logical page list” on page 155.

Obtaining information about pages in error (LPL)

The logical page list (LPL) contains a list of pages (or a page range) that could not be read or written for some reason, such as 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 DASD or if locks are held by a failed subsystem, thus disallowing access to the desired page.

The LPL 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 LPL, it receives a “resource unavailable” SQLCODE. In order to be accessible, pages in the LPL must first have their logged changes applied to the page set.

General-use Programming Interface

To verify the existence of LPL entries, issue the DISPLAY DATABASE command. The LPL option of DISPLAY DATABASE can then be used to see the specific list of pages:

```
  -DB1G DIS DB(DSNDB01) SPACENAM(*) LIMIT(*) LPL ONLY
```

Output similar to the following is produced:

```
DSNT360I -DB1G
***********************************************************
DSNT361I -DB1G *  DISPLAY DATABASE SUMMARY
               *  GLOBAL LPL
DSNT360I -DB1G
***********************************************************
DSNT362I -DB1G    DATABASE = DSNDB01  STATUS = RW
               DBD LENGTH = 8000
DSNT397I -DB1G
NAME     TYPE PART STATUS             LPL PAGES
-------- ---- ---- ------------------ ------------------
DBD01    TS        RW,LPL,GRECP       000001,000004,00000C,000010
----                                  000039-00003C
SYSLGRNX TS        RW,LPL,GRECP       000000-FFFFFF
******* DISPLAY OF DATABASE DSNDB01  ENDED    ************************
DSN9022I -DB1G DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

If LPL entries exist, LPL recovery begins when you start the table space, index, or partition by using the START DATABASE command with ACCESS(RW) or ACCESS(RO).

End of General-use Programming Interface

Physical R/W errors: In previous releases 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 LPL. If recovery from the LPL is unsuccessful, the error is then recorded on the error page range.

Obtaining information about locks held during DB2 failure

When a lock is used to allow an object to be changed (this is called a *modify* lock), the lock is kept in a list in the coupling facility lock structure to allow for recovery in case a DB2 subsystem fails. If a DB2 subsystem fails, modify locks become *retained* locks, which means they are held until the failed subsystem is restarted. For more information about retained locks, see “Active and retained locks” on page 169.

To determine if there are retained locks, use the DISPLAY DATABASE command with the LOCKS option as shown here:

```
-DB1G 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 DB2G
```

Removing retained locks: A normal restart of DB2 resolves and removes retained locks held by that DB2 with the full data integrity control that DB2 restart provides. However, if for some reason you cannot get DB2 restarted and the failed DB2 has retained locks that are severely affecting transactions on other DB2s, 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 (LPL). Those pages are still inaccessible. However, this approach still has the advantage of removing any retained page set P-locks, which have the potential of locking out access to an entire page set. See Section 4 (Volume 1) of *DB2 Administration Guide* for more information about deferred restart.

- 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 will be looking at data whose status is unclear. See “Restarting a DB2 member with conditions” on page 177 for more information about how to do this.

Controlling connections to remote systems

This section describes how controlling DDF connections is changed for data sharing. The following topics are described:

“Starting and stopping DDF”
“Monitoring connections to remote systems” on page 145
“Resetting generic LU information” on page 145

Starting and stopping DDF

General-use Programming Interface

The distributed data facility (DDF) is controlled on a member basis. This gives you more control over DDF processing in the group. Say, for example, that you want to devote DB1G 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:

```
-DB1G STOP DDF MODE(QUIESCE)
```

To stop *all* DDF processing for the group, you have to enter the STOP DDF command for every member of the group. You might need to do this when, for example, you change the SYSIBM.LOCATIONS table of the CDB.

# To allow for completion of CREATE, ALTER, DROP, GRANT, or REVOKE
# operations from remote requesters, issue the STOP DDF MODE(SUSPEND)
# command.

End of General-use Programming Interface

Monitoring connections to remote systems

General-use Programming Interface

There are no new commands for monitoring connections to or from a data sharing group. The DISPLAY LOCATION and DISPLAY THREAD commands show information only for the member on which it is issued.

If your data sharing group is defined with a generic LU, you must use the *real* LU name for *luwid*, if you are requesting information by *luwid*.

When a remote DB2 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.

End of General-use Programming Interface

Resetting generic LU information

If you are using a generic LU name to connect to a member of the data sharing group using 2-phase commit, VTAM permanently records information in the coupling facility about which member of the DB2 group was involved in the communication. This permanently recorded information is required to guarantee that future VTAM sessions are always directed to the same DB2 group member, making it possible to provide access to the correct DB2 subsystem log for resolution of indoubt threads.

There might be times when you need to break that affinity between the group member and the other system. You would need to do this, for example, if you want to start using the member-specific method, or if you want to remove a member from the data sharing group.

General-use Programming Interface

To break this affinity, use the RESET GENERICLU command. The command must be issued from the member with the affinity to the particular LU. Here is an example that removes information about USIBMSTODB22 from DB1G:

```
-DB1G RESET GENERICLU(LUDB22)
```

Great care should be taken when using this command, because it can potentially cause the specified partner LUs to connect to different DB2 subsystems on later sessions. This can cause operational problems if indoubt threads exist at a partner

LU when this command is issued. This command can be issued from any member of the data sharing group.

End of General-use Programming Interface

For more information about using the RESET GENERICLU command, see Chapter 2 of *DB2 Command Reference*.

Establishing the logging environment

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 also contains information about other members' BSDS and log data sets, in case the SCA is not available.

Figure 29 illustrates a typical logging environment.

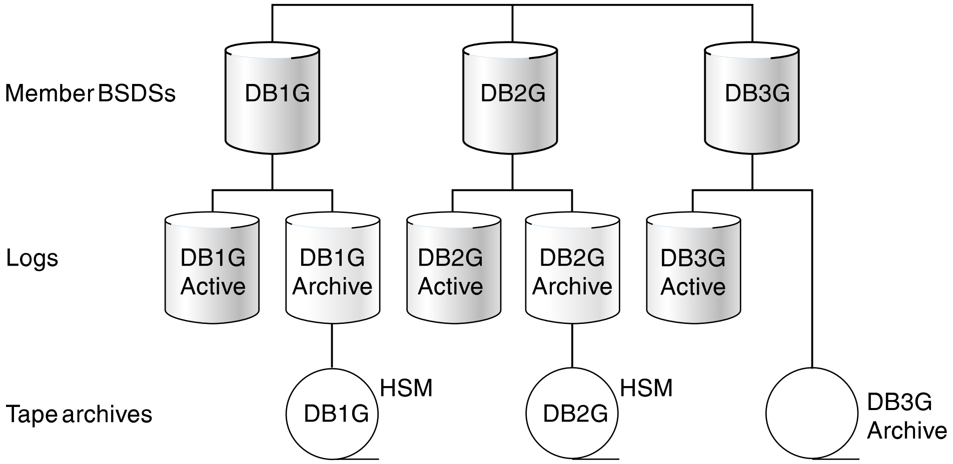


Figure 29. Member BSDSs and logs in the data sharing environment

The impact of archiving logs in a data sharing group

In data sharing, DB2's RECOVER utility needs log records from every member that has changed the object needing to be recovered. More information about how RECOVER uses these logs is described in “How recovery works in a data sharing group” on page 149. If the logs are archived, the impact on RECOVER depends on how the log data sets are archived:

- Archive to DASD without DFSMShsm to migrate the data sets from DASD to tape.

  There is no major impact on performance. But you need enough DASD to hold archive logs, and the DASD devices must be shared (accessible) by all DB2 members in a data sharing group. Because DFSMShsm™ or its equivalent is not used, you must manage DASD carefully to avoid running out of space.

- Archive to DASD with DFSMShsm.

DFSMShsm can do automatic space and data availability management among storage devices in a system. DFSMShsm can migrate the archive on DASD to less expensive storage (such as tape), and recall back to DASD 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 DB1G might need log data sets from DB1G, DB2G, and DB3G. DB1G 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 DB2 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**.

  Archiving to tape is not recommended because there can be negative consequences to not having enough tape units allocated: **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.

  If you must archive to tape, make sure 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 8 members, each member should specify at least 8 tape drives. You'll 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.

  Also, make sure you specify 0 for the DEALLOC PERIOD field 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 DB2, this suggestion does not apply.)

For data sharing, we recommend that you avoid using tape archive logs for data recovery.

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 latest image copy, the optional incremental copies, and the log records since the last incremental copy (or image copy, if there is no incremental copy). A RECOVER job needs no archived log records if all the log records since the last incremental image copy are still in active log data sets. This 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 DASD.

There are several ways to minimize the need to go to 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 31. Because each DB2 member can have up to 31 active log data sets, the total number of active log data sets is effectively increased by the number of DB2 members in a data sharing group.

The size of an active log data set is up to 2 gigabytes 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 volume. With the ever-increasing capacity of the new tape devices, the size of the active log can also increase. However, some of the increased capacity is due to a tape compression algorithm. We do not recommend using 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 it if there is compression.

(This is not to be confused with the 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 copies, the less chance there is that archive log records will be needed. Of course, this consideration needs to be weighed against the time it takes to make the incremental image copies and their effects on SQL transactions.

  See "Preparing for faster recovery" on page 150 for more information about improving recovery performance.

- **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 FREQ field of installation panel DSNTIPN to help you track when applications are not committing to the frequency set at your site.

Recovering data

This section describes the changes in data recovery that are required by data sharing, including data affected by the failure of the coupling facility or structures within the coupling facility.

The procedures for data recovery are fundamentally the same for data sharing as for non-data-sharing. Data sharing involves one catalog, but there are now many logs and BSDSs. In addition to DASD and cache controllers, a new medium, the coupling facility, is introduced. This adds a possible point of failure and necessitates appropriate procedures for data recovery. In planning for data sharing, it is important to consider having more than one coupling facility. Should a structure failure occur, recovery for the SCA and lock structure can proceed automatically if a second coupling facility is available.

The following topics are described in this section:

- "How recovery works in a data sharing group" on page 149
- "Preparing for faster recovery" on page 150
- "Using the RECOVER utility" on page 151
- "Recovering a data sharing group in case of a disaster" on page 152

- “Recovering pages on the logical page list” on page 155
- “Recovery from coupling facility failures” on page 155
- “Coupling facility recovery scenarios” on page 160

How recovery works in a data sharing group

This section describes how the recovery process works when a shared object needs to be recovered.

Determining the logs needed for recovery

Let's assume there are three members making updates to table space TS1 as shown in Figure 30.

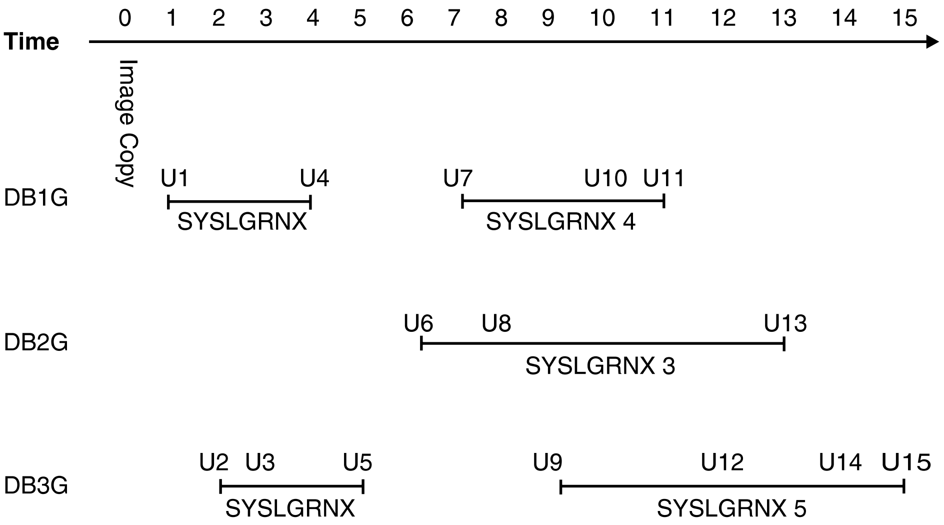


Figure 30. Three DB2 Subsystems Updating Table Space TS1

Here is the sequence of updates leading up to the time of recovery:

- DB1G updated TS1 between Time 1 and 4 (SYSLGRNX record 1) with updates 1 and 4 (U1 and U4 in the figure). DB1G updated TS1 again between Time 7 and 11 (SYSLGRNX record 4) with U7, U10, and U11.
- DB2G updated TS1 between Time 6 and 13 (SYSLGRNX record 3) with U6, U8, and U13.
- DB3G updates TS1 between Time 2 and 5 (SYSLGRNX record 2) with U2, U3, and U5. DB3G updates TS1 again between Time 9 and 15 (SYSLGRNX record 5) with U9, U12, U14 and U15.

Now, assume 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 above are selected to determine the log ranges of each system for the log scan. Updates U1 through U9 are applied in order.

Applying the log records

DB2 can access the logs of other DB2 systems in the group and merge them in sequence. Hence, the merge process needs a unique, group-wide identifier. The log record sequence number (LRSN), a 6-byte hex value derived from a store clock timestamp, is that identifier. Figure 31 shows the structure of the log record.

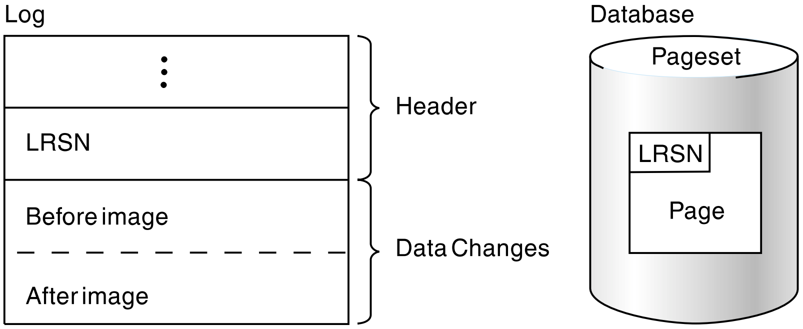


Figure 31. The Log and LRSN in the Data Sharing Environment. During recovery, DB2 compares the LRSN in log record with LRSN in the data page to determine whether the log record must be applied to the data on DASD.

The log record header contains a log record sequence number (LRSN). The LRSN is a 6-byte value that is equal to or greater than 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 DASD. If the LRSN in the log record is larger than the LRSN on the data page, the change is applied.

Preparing for faster recovery

| This section describes two ways to increase your recovery performance: more frequent image copies, and faster log apply.

Increase the frequency of copies

One way to prepare for a quicker recovery is to increase the frequency of copies. You might want to limit this activity by determining which table spaces most need this fast recovery. The following guideline is provided as a starting point to help you determine how often you must do incremental image copies. The assumption is that you are already familiar with using the COPY utility and with all the ramifications of using incremental versus full image copies, as described in Section 2 of *DB2 Utility Guide and Reference*. As with a single subsystem, doing frequent enough image copies can help you avoid going to the archive log for recovery.

Use the guideline below for each member of the data sharing group. Use the output of the print log map utility (DSNJU004) 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:

   ```
   time_interval = end_TIMESTAMP – start_TIMESTAMP
   ```

4. Calculate the interval at which to perform incremental image copies:

   ```
   interval of copy = time_interval * (n-1) / n
   ```

Where 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 percent. (30 percent 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.

   Be sure to periodically run MERGECOPY utility with incrementals. The RECOVER utility attempts to mount tape drives for all the incrementals at the same time. If it runs out of tape drives, it then switches to log apply. MERGECOPY merges what it can and then mounts more incrementals.

Enable fast log apply

Improve recovery times (and restart times) by providing enough storage for DB2's fast log apply process. This process is able to sort log records so that pages that are to be applied to the same page or same set of pages are together. Then, using several log apply tasks, it can apply those records in parallel.

Provide storage for this process using the LOG APPLY STORAGE field of installation panel DSNTIPL.

Using the RECOVER utility

Use the RECOVER utility to recover to currency or to a prior point in time. The details of RECOVER are described in Section 2 of *DB2 Utility Guide and Reference* .

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. Such problems with the 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 an 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 pending recovery 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 DASD.

Recovering a data sharing group in case of a disaster

This section presents an overview of how to recover a data sharing group at a remote site. To develop a procedure, you can use as a base the disaster recovery procedure documented in Section 4 (Volume 1) of *DB2 Administration Guide* . With a couple of exceptions, you must perform those steps for each member of the data sharing group. The following topics describe how to prepare for recovery of a data sharing group at a recovery site:

- "Configuring the recovery site"
- "What to send to the recovery site" on page 154

"Recovery procedure differences" on page 155 is the procedure you use to prepare the data sharing group at the recovery site for a group restart.

Using a Tracker Site for Disaster Recovery: 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 needed for takeover should a disaster occur 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. See Section 4 (Volume 1) of *DB2 Administration Guide* for more information about setting up and using a tracker site.

Configuring the recovery site

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 names of the members must be the same. The CFRM policies at the recovery site must define the coupling facility structures with the same names (although the sizes can be different).

You can run the data sharing group on as few or many MVS systems as you want.

The hardware configuration can be different at the recovery site, as long as it supports data sharing. 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.**

  This is the way the local site is most likely configured. You have a Parallel Sysplex containing many CPCs, MVS systems, DB2s. This configuration requires a coupling facility, the requisite coupling facility channels, and the Sysplex Timer.

  The advantage to this is you have the same availability and growth options that you had on the local site.

- **Run a single-system data sharing group.**

In this configuration, you centralize all your DB2 processing within a single, large CPC, such as an IBM S/390 9672 Rx5 or later CMOS processor. As Figure 32 on page 153 shows, you must *install* a multi-member data sharing group. After the group starts up, you shut down all but one of the DB2s and access data through that single DB2.

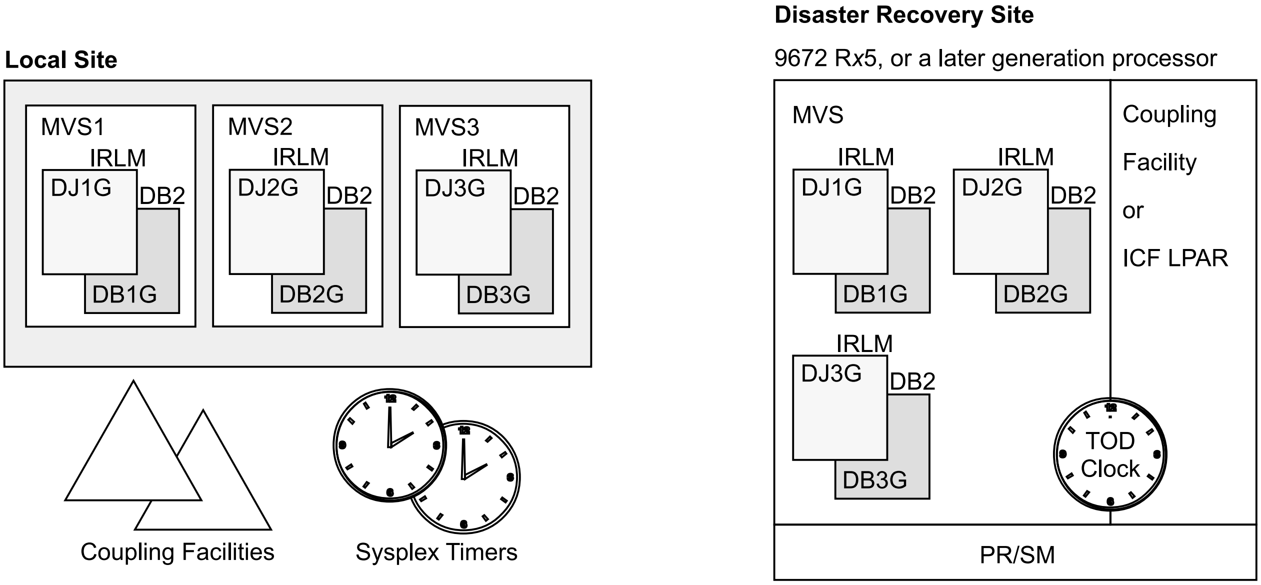


Figure 32. Example of local and disaster recovery site configurations. With this configuration, the recovery site can be a single-system data sharing group. After the data sharing group is started on the recovery site, all but one of the members can be stopped.

Obviously, you lose the availability benefits of the Parallel Sysplex, but the single-system data sharing group has fewer hardware requirements:

– The Sysplex Timer is not needed; the time-of-day clock of the CPC can be used.

– 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 R/W 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 so that DB2 can allocate them and 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.

For more information about ICF, see *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide*.

What to send to the recovery site

You must send the same information as documented for single-system remote recovery: logs and BSDSs, image copies, and so on. To prepare the logs for the remote site, you have two 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 you do 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 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 change log inventory utility on the remote site to truncate all logs to the same point in time.

  To determine the truncation value, you can 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 as a result of someone issuing 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 -DB1G DSNJOFF3 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 -DB2G DSNJOFF3 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 here, 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 your ENDLRSN value in the CRESTART statement of the change log inventory utility for each of the members at the remote site. All log records with a higher LRSN value are discarded during the conditional restart.

Attention: 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.

Recovery procedure differences

The procedure at the recovery site differs for data sharing in that there are extra steps for cleaning out old information in the coupling facility. Old information is in the coupling facility from any practice startups. In addition, you must prepare each subsystem for conditional restart rather than just a single system. For the detailed procedure, see Section 4 (Volume 1) of *DB2 Administration Guide*.

Recovering pages on the logical page list

In some cases, DB2 can automatically recover pages on the logical page list when group buffer pools are defined with AUTOREC(YES), the default. However, there are many situations where pages are put on the LPL that require you to do manual recovery. There are several ways to do this:

- Start the object with access (RW) or (RO). That 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 DSNI021I.

- Run the RECOVER utility on the object.

  The only exception to this is when a logical partition of a type 2 non-partitioned 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 LPL 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 above list work if there are retained locks held on the object. You must restart any failed DB2 that is holding those locks.

Recovery from coupling facility failures

Failures of the coupling facility can be classified into two main groups:

- Connectivity failures

  Connectivity failures can be caused by a problem with the attachment of the MVS 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 MVSs running.
  - Deactivation of the coupling facility partition.
  - Coupling facility control code failure.
  - Coupling facility CPC or LPAR failure.

- Structure failure

A structure failure is a rare event in which structures are damaged in some way but the coupling facility continues to operate.

This section also includes information about allocation failure and problems caused by not enough storage.

Preparing for structure and connectivity failures

Coupling facility failures can mean serious outages for users. Not having a lock structure or SCA causes the entire group to come down abnormally. Group buffer pool failure does not cause the group to come down, but it can still 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 on your end users. To best prepare yourself for both types of failures, you must have the following:

- An active system failure management (SFM) policy with system weights specified or APAR OW30814 installed.

  This is strongly recommended. Descriptions of failure scenarios in this section assume you have done this. If you have not, it is not possible to automatically rebuild 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, then those members disconnect from the group buffer pool.
- Alternative coupling facility information provided on the preference list of each of the structures in the CFRM policy.
- A REBUILDPERCENT value specified in the CFRM policy for all DB2-related structures. In general, it is best to have a low REBUILDPERCENT value specified 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, MVS normally uses the INITSIZE value of the CFRM policy on the alternate coupling facility to allocate storage. If MVS 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 DASD instead of rebuilding them into the alternate group buffer pool. For more information about how structure allocation works, see *OS/390 MVS Programming: Sysplex Services Guide*.
- For page sets requiring very high availability, use of group buffer pool duplexing.

For more information about planning, see “Coupling facility availability” on page 47.

Summary of failure scenarios

The tables in this section summarize connectivity and structure failure situations.

- “Connectivity failure for lock structure and SCA” on page 157
- “Connectivity failure for non-duplexed group buffer pools” on page 157
- “Structure failures” on page 159
- “Summary of failure scenarios for duplexed group buffer pools” on page 159

For more information about specific recovery scenarios, see “Coupling facility recovery scenarios” on page 160.

Connectivity failure for lock structure and SCA: Table 25 summarizes what happens when there are connectivity failures to the lock structure or SCA.

Table 25. Summary of connectivity failures for the SCA and lock structure

Situation	**Without Active SFM Policy OR weighted loss < REBUILDPERCENT**	**Operational Response**	**Active SFM Policy where weighted loss ≥ REBUILDPERCENT**	**Operational Response**
Lost connectivity to SCA	Each affected member: `DSN7501A 00F70600` DB2 comes down. Connection is deleted; structure remains allocated.	Options include: • Fix problem • Restart failed member on system that is connected to coupling facility • Manually rebuild onto another coupling facility.	Automatic rebuild `DSN7503I`	None needed.

Lost connectivity to lock structure.	Each affected member:
	`DXR136I 00E30105`
	DB2 comes down. Connection is failed-persistent; structure remains allocated.
		Same as SCA.	`DXR143I`
			Automatic rebuild
			`DXR146I`
				None needed.

Connectivity failure for non-duplexed group buffer pools: Table 26 on page 158 summarizes what happens when there are connectivity failures to the group buffer pools.

Table 26. Summary of connectivity failures for non-duplexed group buffer pools

Situation	Weighted loss < REBUILDPERCENT	Operational Response	Weighted loss ≥ REBUILDPERCENT	Operational Response
Lost connectivity from all connected members to a group buffer pool that is defined with GBPCACHE(YES)	Not applicable	Not applicable	Each affected member: DSNB303E DSNB228I Add pages to LPL, 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.
Lost connectivity from all connected members to a group buffer pool that is defined with GBPCACHE(NO)	Not applicable	Not applicable	Automatic rebuild. DSNB331I DSNB338I	None needed.
A subset of connected members lost connectivity to some or all group buffer pools.	Each affected member: DSNB303E DSNB228I DSNB313I rsn=LOSSCONN Quiesce applications that use the group buffer pool. Add pages to LPL if necessary. DSNB250E DSNB311I, DSNB312I Disconnect GBPx failed-persistent. DSNB309I	Options include: • Fix the problem. • Manually rebuild the structure onto another coupling facility. • Stop and restart DB2 on a system that is connected to the coupling facility. Enter START DATABASE commands to recover LPL entries.	Automatic rebuild. DSNB331I DSNB332I DSNB333I* DSNB338I	None needed.

Note: *Issued by the structure owner.

Structure failures: Table 27 summarizes what happens to each structure if there is a structure failure. For more information, see:

"Problem: group buffer pool structure failure (no duplexing)" on page 164
"Problem: lock structure failure" on page 164
"Problem: SCA structure failure" on page 164

Table 27. Summary of structure failures, by structure type

Failed structure	DB2 response	Operational response
SCA	DSN7502I Automatic rebuild DSN7503I	None needed.
Lock structure	DXR143I Automatic rebuild DXR146I	None needed.
Group buffer pool that is defined with GBPCACHE(YES)	DSNB228I DSNB314I rsn=STRFAIL Add pages to LPL, 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.

Summary of failure scenarios for duplexed group buffer pools

For duplexed group buffer pools, a failure response is the same for both structure failures and for lost connectivity.

Table 28 (Page 1 of 2). Summary of scenarios for both structure failure and lost connectivity for duplexed group buffer pools

Failure occurred for which structure?	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 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.

Table 28 (Page 2 of 2). Summary of scenarios for both structure failure and lost connectivity for duplexed group buffer pools

Failure occurred for which structure?	DB2 response	Operational response
Both (structure failure 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.

Coupling facility recovery scenarios

The following scenarios are described here:

- “Problem: all members have lost connectivity”
- “Problem: a subset of members have lost connectivity” on page 162
- “Problem: group buffer pool structure failure (no duplexing)” on page 164
- “Problem: lock structure failure” on page 164
- “Problem: SCA structure failure” on page 164
- “Problem: allocation failure of the group buffer pool” on page 165
- “Problem: storage shortage in the group buffer pool” on page 165
- “Problem: storage shortage in the SCA” on page 166
- “Problem: storage shortage in the lock structure” on page 167

Because some problems might require you to deallocate structures by force, we also include information about how to do that in “Deallocating structures by force” on page 168.

Problem: all members have lost connectivity

This scenario explains what you might see if there was a failure that is treated by MVS and DB2 as total loss of connectivity to the coupling facility. It assumes you have an active SFM policy or APAR OW30814 installed.

Symptom: Some or all of the following messages appear, depending on which structures DB2 tries to access:

```
DSNB303E -DB1G 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 gpbname 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
```

```
DXR143I irlmx REBUILDING LOCK STRUCTURE BECAUSE IT HAS FAILED OR AN IRLM
        LOST CONNECTION TO IT
```

System action: When all active members lose connectivity to the **SCA** or **lock structure**, these structures are rebuilt:

DSN7503I is issued for a successful rebuild of the SCA.
DXR146I is issued for a successful rebuild of the lock structure.

Important: If the lock structure or SCA cannot be rebuilt, the lost connectivity causes the members of the group to abend with abend code 00E30105 or 00F70600. If they cannot be rebuilt, such as if both coupling facilities are volatile and lose power, a group restart is required. Group buffer pools cannot be automatically recovered during group restart, and you have to recover those group buffer pools with START DATABASE commands.

To avoid situations in which group restart is necessary, put structures in nonvolatile coupling facility structures. See “Coupling facility volatility” on page 51 for more information.

For the lost connectivity to the **group buffer pools**, applications needing access to group buffer pool data are rejected with a -904 SQL return code and can see any of the following reason codes:

00C20204
00C20205
00C20220

- DB2 puts the group buffer pool in “damage assessment pending” status. The following message appears:

  ```
  DSNB304I -DB1G  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 DB2 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 MVS 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: The problem causing the loss of connectivity must be fixed. If the problem is not an obvious one (such as a power failure), call IBM service. After the problem is fixed, restart any failed members as quickly as possible to release retained locks.

- For the SCA or lock structure, if the automatic rebuild occurred normally, processing can continue while you wait for the coupling facility problem to be fixed.
- 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 LPL after recovery, issue START DATABASE commands for all affected page sets. You must issue separate START DATABASE commands in the following order.
  1. DSNDB01
  2. DSNDB06

Problem: a subset of members have lost connectivity

This scenario describes what happens if one or more members have lost connectivity to the coupling facility, but some systems are still connected. This might happen if a link is detached between a system and the coupling facility. **This scenario assumes that the combined system weights of the systems that have lost connectivity is less than that required to trigger an automatic rebuild of the structure; that is, less than the value specified on the REBUILDPERCENT parameter of the CFRM policy. Assume that your group buffer pool is not duplexed.** If the combined system weight is greater than or equal to that required to cause rebuild, automatic rebuild occurs. When DB2 rebuilds a 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 DASD, instead.

Operator intervention is usually not required unless DB2 is required to add pages to the LPL for some reason while the rebuild occurs. In this case, a START DATABASE command is needed recover the pages on the LPL.

Symptom: Some or all of the following messages appear, depending on which structures DB2 tries to access:

```
DSNB303E -DB1G csect-name A LOSS OF CONNECTIVITY
WAS DETECTED TO GROUP BUFFER POOL gbpname.
```

```
DSNB301E -DB1G csect-name GROUP BUFFER POOL
gbpname CANNOT BE CONNECTED
DB2 REASON CODE =  reason1
MVS IXLCONN REASON CODE =  xxxx0C06
```

```
DSNB228I csect-name GROUP BUFFER POOL gbpname
CANNOT BE ACCESSED FOR function
MVS IXLCACHE REASON CODE=reason
```

```
DSNB313I -DB1G csect-name GROUP BUFFER POOL gbpname
TO BE DISCONNECTED
REASON=LOSSCONN
LOSSCONN PERCENTAGE=percentage
```

```
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
```

```
DSNB311I csect-name DBNAME database SPACE NAME spacename
HAS PAGES IN THE LOGICAL PAGE LIST
```

```
DSNB312I csect-name DBNAME database SPACE NAME spacename
PARTITION part-number
HAS PAGES IN THE LOGICAL PAGE LIST
```

```
DSNB309I csect-name GROUP BUFFER POOL gbpname
HAS BEEN DISCONNECTED WITH
A REASON OF 'FAILURE'
```

```
DXR136I irlmx HAS DISCONNECTED FROM THE DATA SHARING GROUP
```

```
DSN7501A -DB1G csect-name SCA STRUCTURE sca-structure-name
               CONNECTIVITY FAILURE.
```

System action: For a loss of connectivity to the **SCA** or **lock structure**, DB2 abnormally terminates on the members affected by the loss of connectivity. Either abend code 00E30105 or 00F70600 is issued for those members.

For a loss of connectivity to the **group buffer pool**:

1. DB2 adds entries to the logical page list, if necessary.
2. Applications running on the members with the lost connectivity continue processing until the next COMMIT point. On the next attempt to access a GBP-dependent page set associated with a disconnected group buffer pool, applications receive a -904 SQLCODE (reason code 00C20204). For inflight units of recovery, the loss of connectivity is detected immediately, and the application receives a -904 SQLCODE (reason code 00C20220).
3. DB2 disconnects from the group buffer pool and issues message DSNB309I with a reason of FAILURE. The connection enters a failed-persistent state.

System programmer action: The problem causing the loss of connectivity must be fixed. If the problem is not an obvious one (a disconnected link, for example), call IBM service. After the problem is fixed, restart any failed members as quickly as possible to release retained locks.

Consider the following options:

- If connectivity was lost to the SCA or lock structure causing the member to fail, restart the failed member on another system that has connectivity to the structure.

  If connectivity is lost to a group buffer pool, DB2 continues to run. If you want to start the member on another system, you must stop DB2 before you restart it.
- Rebuild the structure manually onto another coupling facility using the MVS SETXCF command:

```
SETXCF START,REBUILD,STRNM=strname,LOC=OTHER
```

After you rebuild the group buffer pool, there might still be LPL entries for the page sets. To recover these, enter the command START DATABASE after receiving a DSNB311I or DSNB312I for the page set or partition. Issue the command from a DB2 member that is connected to the group buffer pool.

To avoid manual intervention next time, lower the REBUILDPERCENT value in the CFRM policy so that the next time you lose connectivity, DB2 can automatically rebuild.

Problem: group buffer pool structure failure (no duplexing)

Symptom: The following message appears on the console of the member who will be coordinating damage assessment:

```
DSNB314I csect-name DAMAGE ASSESSMENT TO BE TRIGGERED FOR
GROUP BUFFER POOL gpbname REASON=STRFAIL
```

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. Applications needing access to group buffer pool data are rejected with a -904 SQL return code and can see any of the following reason codes:

00C20204
00C20205
00C20220

See “Problem: all members have lost connectivity” on page 160 for a description of what DB2 does during automatic recovery.

System programmer action: Correct the coupling facility failure. For any page sets that were not automatically recovered by DB2, 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.

Problem: lock structure failure

Symptom: Locking requests are suspended until the lock structure is rebuilt. If the lock structure cannot be rebuilt, the following message appears:

```
DXR136I irlmx HAS DISCONNECTED FROM THE DATA SHARING GROUP
```

System action: If the structure cannot be rebuilt, all active members of the group terminate abnormally with a 00E30105 abend code.

System programmer action: See message DXR135E for the root cause of the problem and the corrective procedure.

Problem: SCA structure failure

Symptom: The following message appears:

```
DSN7502I -DB1G csect-name SCA STRUCTURE FAILURE,
ATTEMPT TO REBUILD IS IN PROGRESS.
```

DB2 suspends processing until the SCA is rebuilt, using information contained in DB2's memory.

If the SCA cannot be rebuilt, the following message appears:

```
DSN7504I -DB1G csect-name SCA STRUCTURE structure-name
               REBUILD UNSUCCESSFUL.
```

System action: If the rebuild is unsuccessful, all DB2s in the group terminate abnormally.

System programmer action: Check the termination code for the reason the rebuild was unsuccessful. Correct the problem and then restart the members of the group.

Problem: allocation failure of the group buffer pool

Symptom: The following message appears:

```
DSNB301E -DB1G 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, then there can be no inter-DB2 R/W activity 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, there can be no update activity 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. See “Changing the size of the group buffer pool” on page 256 for more information.
- Have the database administrator reassign some of the table spaces or indexes using that group buffer pool to a different group buffer pool.

Problem: storage shortage in the group buffer pool

Symptoms: The following message appears when the group buffer pool is 75% full:

```
DSNB319A -DB1G csect-name THERE IS A SHORTAGE OF SPACE IN GROUP
BUFFER POOL gbpname
```

If you don't do anything to relieve the shortage, the following message appears when the group buffer pool is 90 percent full:

```
DSNB325A -DB1G 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 the group buffer pool, 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. See “Group buffer pool size is too small” on page 248 for more information about such symptoms and how to avoid having writes to the group buffer pool fail because of a lack of storage.

System action: DB2 initiates castout processing if it isn't already in progress. DB2 then tries again to write to the group buffer pool. For simplexed group buffer pools, or for the primary of a duplexed group buffer pool, pages that cannot be written to the group buffer pool are added to the logical page list and message DSNB250E is issued.

If it is the secondary group buffer pool that is too full, DB2 does not add pages to the logical page list; instead, it takes the structure out of duplexing mode.

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 be used as a temporary solution.

Problem: storage shortage in the SCA

Symptoms: The following message appears:

```
DSN7505I -DB1G csect-name THERE IS A SHORTAGE OF FREE STORAGE
          IN SCA STRUCTURE sca-structure-name.
```

If you don't do anything to reclaim space, such as recovering pages from the LPL, the following message appears when the SCA is 90 percent full:

```
DSN7512A -DB1G 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: Do 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 (LPL).

2. Restart any failed DB2s.

If your actions do not free up enough space, or if this problem continues to occur, you have the following options, depending on what level of MVS and coupling facility you have.

- If all of the following conditions are true:

  – All members of the group are running with MVS Version 5 Release 2 or above
  – 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

  Then you can enter the following command to increase the size of the SCA (this example assumes the group name is DSNDB0G):

  ```
  SETXCF START,ALTER,STRNAME=DSNDB0G_SCA,SIZE=newsize
  ```

This example assumes 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:
  - Any member of the group is not running with MVS Version 5 Release 2 or above
  - 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 MVS command SETXCF START,POLICY to start the updated policy
  3. Use the following MVS command to rebuild the structure:

     ```
     SETXCF START,REBUILD,STRNAME=DSNDB0G_SCA
     ```

- If **all** members are down, and you cannot alter the SCA to a larger size, you must do the following:

  1. Delete the SCA structure by using the following command:

     ```
     SETXCF FORCE,STRUCTURE,STRNAME=DSNDB0G_SCA
     ```

  2. Increase the size of the SCA in the CFRM policy.
  3. Restart DB2 to rebuild the SCA using group restart, as described in “Group restart” on page 170.

Problem: storage shortage in the lock structure

Symptom: A DXR170I message indicates when storage reaches 50, 60, and 70% full. The following message appears at increasing thresholds starting at 80% full:

```
-DB1G 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 no DB2s are down and holding retained locks. Restarting any failed DB2s can remove the locks retained in the coupling facility lock structure and release that space.

If a failed DB2 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, as described in “Changing the size of the lock structure” on page 219.

Deallocating structures by force

In a few exceptional cases, you might need to deallocate a structure by force to get DB2 restarted. This might be necessary, for example, in the following cases:

- The structure is corrupted
- You know the structure does not exist but MVS thinks it does.

When you force off 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, you must use MVS 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
  ```

Restarting DB2 after termination

After a failure or after a normal shutdown of DB2, you can restart DB2 with the command START DB2. You can also choose to have DB2 automatically restart after a failure by using the automatic restart manager of MVS. See “Using MVS's automatic restart capability” on page 46 for more information.

During restart, DB2 resolves inconsistent states. Restart is changed for data sharing because of the following:

- Database exception states, which exist solely on the log in a non-data-sharing environment, are on both the SCA and the log in data sharing.
- 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 DB2s in the group start up again, but information from all nonstarting DB2s must be used to rebuild the lock structure or SCA.

In this section:

- “Normal restart for a data sharing member” on page 169
- “Group restart” on page 170
- “Phases of group restart” on page 171
- “Protecting retained locks: failed-persistent connections” on page 175

- “Postponing backout processing” on page 175
- “Restarting a DB2 member with conditions” on page 177
- “Deferring recovery during restart” on page 178

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. In this section we describe some additional information about locks, because 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 and retained locks

Active locks: When a DB2 member is active, the locks 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. For more information about locking mechanisms, see “Improving concurrency” on page 203.

Retained locks: The particular concern for availability is what happens to locks when a DB2 subsystem 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 they are released during restart. Retained locks are necessary to protect data in the process of being updated from being accessed by another active DB2 member of the group. See “Protecting retained locks: failed-persistent connections” on page 175 for information about the failed-persistent connections associated with retained locks.

DB2 has various types of retained locks. Among them are L-locks, page set P-locks, or page P-locks. As long as an incompatible lock is held by failed member, another member cannot lock the resource in a mode that 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 field of installation panel DSNTIPI; if the value is 0, they are immediately rejected.

In the case of a page set P-lock, it is conceivable that an entire page set could be unavailable (an X mode page set P-lock is retained if the page set was 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, **it is important to restart failed DB2s as soon as possible**, either on the same or another MVS.

Retained utility ID locks: When a member is running a utility, it is holding a lock on the utility ID (UID) for that utility. That lock, too, is retained should the member fail. This means you cannot restart a utility until that member is restarted and the retained lock is converted to an active lock.

When retained locks are reacquired or purged

During the restart process, DB2 will remove its retained locks in either of two ways:

- Convert the lock to active, called *reacquiring* the lock. This is what DB2 does for page set P-locks.
- Purge the lock. This is what DB2 does for page P-locks and for L-locks.

This process of reacquiring or purging locks can happen at different times in the restart process, depending on the type of retained lock as shown in Table 29.

Table 29. Restart processing for locks

Lock Type	Processing
Page set P-lock	Reacquired when page sets are opened for log apply. This generally happens during forward recovery or before restart completes.
Note: The earliest that a page set P-lock can become negotiable is at the end of forward recovery. See “Page Set P-Locks” on page 227 for more information about negotiation. If no log apply is needed, it can happen later, such as the end of restart processing.	
Page P-lock	Purged at the end of forward recovery.
L-lock	Purged at the end of restart processing.

If the DB2 requesting a lock which is incompatible with a retained lock has a nonzero value for RETAINED LOCK TIMEOUT, its applications can be suspended waiting for those retained locks. Those requests can go through as soon as the retained locks are purged or become 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.

Group restart

Group restart requires scanning the logs of each member to rebuild the SCA or retained lock information. This is why we recommend that you have an alternate coupling facility on which these vital structures can be automatically rebuilt in the event of a coupling facility failure. That automatic rebuild does not require the log scans that group restart does.

During group restart, all restarting DB2s update the SCA or lock structure from information contained on their logs. If you don't enter a START DB2 command for all members of the group, then the started DB2 subsystems carry out group restart on behalf of the nonstarting subsystems by reading their logs.

Although one DB2 can perform restart on behalf of the group, we recommend that you restart all of the nonquiesced members together, perhaps by using an automated procedure. This shortens the total restart time. Also, because retained locks are held for nonstarting DB2s, it is best for data availability to start all members of the group.

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.

Once the members are synchronized after forward log recovery, backward log read proceeds in parallel for the started subsystems.

Phases of group restart

The phases of 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 or lock structure is lost (or both), and whether information is needed from the logs of inactive members. Table 30 summarizes the phases, depending on which structure is lost.

Table 30. Summary of 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 recovery (rebuild locks)	Forward recovery (rebuild locks)
	Peer forward recovery (rebuild locks)
Backward Recovery	Backward Recovery

In the message output shown in this section, we are showing a group restart controlled by member DB3G on behalf of members DB2G and DB1G.

DB2 initialization

This phase verifies BSDSs, logs and the integrated catalog facility catalog. The RBA of the last log record is retrieved and logging is set to begin at the next CI following the last RBA. Also during this phase, DB2 determines if the lock structure or SCA is lost and needs to be recovered.

During initialization, you see messages similar to the following:

```
 $HASP373 DB3GMSTR STARTED
⋮
 DSNJ127I @DB3GDB2 SYSTEM TIMESTAMP FOR BSDS= 95.040 13:03:05.32
 DSNJ001I @DB3GDB2 DSNJW007 CURRENT COPY 1 ACTIVE LOG 753
 DATA SET IS DSNAME=DSNC410.THIRD.LOGCOPY1.DS01,
 STARTRBA=000000000000,ENDRBA=000000167FFF
 DSNJ001I @DB3GDB2 DSNJW007 CURRENT COPY 2 ACTIVE LOG
 DATA SET IS DSNAME=DSNC410.THIRD.LOGCOPY2.DS01,
 STARTRBA=000000000000,ENDRBA=000000167FFF
 DSNJ099I @DB3GDB2 LOG RECORDING TO COMMENCE WITH
 STARTRBA=000000010000

⋮
 $HASP373 DB3GDBM1 STARTED
```

Current status rebuild (CSR)

During current status rebuild, the SCA is rebuilt from the log by reading it forward from the last checkpoint. In addition, DB2 determines all outstanding units of recovery (UR) 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.

When a restarting member has completed its own CSR, it checks and waits for every other DB2 member to finish CSR. If there are non-starting DB2 subsystems, then peer CSR is performed.

During current status rebuild, you see messages similar to these (The phrase in parentheses is not part of the output.):

```
DSNR001I @DB3GDB2 RESTART INITIATED
DSNR003I @DB3GDB2 RESTART...PRIOR CHECKPOINT RBA=00000000DC4E
DSNR004I @DB3GDB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of current status rebuild for member DB3G)

DSNR021I @DB3GDB2 DSNRRGRC DB2 SUBSYSTEM MUST PERFORM
GROUP RESTART FOR PEER MEMBERS
```

Peer CSR (used only when some DB2s are not starting)

This activity is skipped unless it is necessary to perform group restart on behalf of non-starting members. (Peer CSR is not used when the non-starting DB2s are normally quiesced.)

A restarting DB2 can select an inactive member on which to perform peer initialization and peer CSR:

- If the SCA is lost, the restarting DB2 rebuilds SCA information from the information contained in the nonstarting member's logs.
- If the lock structure was lost, the restarting DB2 reacquires page set and partition P-locks (as retained locks) for the nonstarting member. Those locks are now retained locks.

When all members have completed current status rebuild, either by doing it on their own or by having a peer do it for them, the SCA has been rebuilt, and page set and partition P-locks have been reacquired.

During peer current status rebuild, you see messages similar to these:

```
 DSNR023I @DB3GDB2 DSNRRGRC GROUP RESTART INITIATED TO
 RECOVER THE SCA FOR GROUP MEMBER DB1G
 DSNR003I @DB3GDB2 RESTART...PRIOR CHECKPOINT RBA=00000201CC4E
 DSNR004I @DB3GDB2 RESTART...UR STATUS COUNTS
 IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
 DSNR024I @DB3GDB2 DSNRRGRC GROUP RESTART COMPLETED TO
 RECOVER THE SCA FOR GROUP MEMBER DB1G
(End of peer current status rebuild for member DB1G)

 DSNR023I @DB3GDB2 DSNRRGRC GROUP RESTART INITIATED TO
 RECOVER THE SCA FOR GROUP MEMBER DB2G
 DSNR003I @DB3GDB2 RESTART...PRIOR CHECKPOINT RBA=000000009C4E
 DSNR004I @DB3GDB2 RESTART...UR STATUS COUNTS
 IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
 DSNR024I @DB3GDB2 DSNRRGRC GROUP RESTART COMPLETED TO
 RECOVER THE SCA FOR GROUP MEMBER DB2G
 (End of peer current status rebuild for DB2G)

 DSNR022I @DB3GDB2 DSNRRGRC DB2 SUBSYSTEM HAS
 COMPLETED GROUP RESTART FOR PEER MEMBERS
 (End of peer processing)
```

Forward log recovery

In this phase DB2 applies log records and completes the database write operations which 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 than 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 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 there is a problem applying a log record for an object (such as if the DASD 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 every other DB2 member to finish. If there are non-starting DB2 subsystems, then peer forward log recovery is performed.

At the end of forward log recovery, the rebuild of the lock structure is complete.

During forward log recovery, you see messages similar to these:

```
DSNR005I @DB3GDB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR021I @DB3GDB2 DSNRRGRH DB2 SUBSYSTEM MUST PERFORM
GROUP RESTART FOR PEER MEMBERS
```

Peer forward log recovery (used only when some DB2s are not starting)

This activity is skipped unless it is necessary to rebuild lock information from information contained in inactive, non-quiesced 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 nonstarted member.

During peer forward log recovery, you see messages similar to these:

```
DSNR025I @DB3GDB2 DSNRRGRH GROUP RESTART INITIATED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB1G
DSNR003I @DB3GDB2 RESTART...PRIOR CHECKPOINT RBA=00000201CC4E
DSNR004I @DB3GDB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of peer partial current status rebuild for DB1G)

DSNR005I @DB3GDB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR026I @DB3GDB2 DSNRRGRH GROUP RESTART COMPLETED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB1G
(End of peer forward log recovery for member DB2G)

DSNR025I @DB3GDB2 DSNRRGRH GROUP RESTART INITIATED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB2G
DSNR003I @DB3GDB2 RESTART...PRIOR CHECKPOINT RBA=000000009C4E
DSNR004I @DB3GDB2 RESTART...UR STATUS COUNTS
IN COMMIT=0, INDOUBT=0, INFLIGHT=0, IN ABORT=0
(End of partial current status rebuild for member DB2G)

DSNR005I @DB3GDB2 RESTART...COUNTS AFTER FORWARD
RECOVERY
IN COMMIT=0, INDOUBT=0
DSNR026I @DB3GDB2 DSNRRGRH GROUP RESTART COMPLETED TO
RECOVER RETAINED LOCKS FOR GROUP MEMBER DB2G
DSNR022I @DB3GDB2 DSNRRGRH DB2 SUBSYSTEM HAS
COMPLETED GROUP RESTART FOR PEER MEMBERS
(End of peer forward log recovery for member DB1G)
```

Backward log recovery

At this point, all forward log applies have been performed and inflight, indoubt, and in-abort transactions are protected by locks. During this phase, DB2 completes recovery processing by reversing all changes performed for inflight and in-abort units of recovery.

Any updates that cannot be externalized to the group buffer pool or DASD cause the affected pages to be added to the logical page list.

Backward log recovery can occur in parallel for all the started subsystems. There is no peer backward log recovery; all members must be started to complete backward log recovery and release the locks held by in-flight and in-abort transactions.

During backward log recovery, you see messages like the following:

```
DSNR006I @DB3GDB2 RESTART...COUNTS AFTER BACKWARD
RECOVERY
INFLIGHT=0, IN ABORT=0
(End of backward log recovery for member DB3G)
DSNR002I @DB3GDB2 RESTART COMPLETED
DSN9022I @DB3GDB2 DSNYASCP 'START DB2' NORMAL COMPLETION
```

The backward log recovery messages for the other members do not appear until those members are actually started.

Protecting retained locks: failed-persistent connections

Do not delete failed-persistent connections to the lock structure.

IRLM uses failed-persistent connections to the lock structure to protect retained lock information that it holds on its own behalf or on the behalf of others. A group restart or a rebuild of the lock structure when another member is down can move the safeguarding responsibility of those retained locks to other IRLMs in the data sharing group.

For example, assume that two members of the data sharing group are down and hold retained locks. If the lock structure is rebuilt, but those members are not restarted, the retained locks for those failed members are now associated with the active IRLM that has the lowest member ID in the group. So, although you might think this IRLM cannot possibly hold any retained locks because the member was quiesced normally, it is entirely possible that that member is safeguarding retained locks for another member until that other member can restart.

In summary, do not ever delete failed-persistent connections to the lock structure (unless you are cleaning out old coupling facility information from a practice disaster recovery scenario).

Postponing backout processing

As described in Section 4 (Volume 1) of *DB2 Administration Guide*, you can postpone backout processing for inabort and inflight units of recovery (URs). The advantage to this is the ability to get DB2 up and accepting new work before handling the backout processing.

To summarize, there are two options that let you enable and control postponed backout. Both of these options are on installation panel DSNTIPN:

- LIMIT BACKOUT

  YES or AUTO enables postponed backout. AUTO is the recommended value, because you do not have to remember to issue commands to start the backout processing.

- BACKOUT DURATION

  A multiplier that indicates how much log to process for backout. This option is valid only when you have specified YES or AUTO for LIMIT BACKOUT.

For more information about these options, see Section 2 of *DB2 Installation Guide* .

Why postpone backout work in data sharing?

While a DB2 subsystem is restarting, it cannot accept new work. If a restarting subsystem 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 there is not enough capacity on the other system, 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 DB2 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 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 backout work pending is blocked by transaction locks that persist 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).

Advisory restart-pending status

To determine if a DB2 member has work pending, use the DISPLAY GROUP command. 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.

To narrow down whether the pending work is indoubt URs or postponed backout URs, enter DISPLAY THREAD TYPE(POSTPONED) and DISPLAY THREAD TYPE(INDOUBT).

- To recover indoubt URs, use the RECOVER INDOUBT command.
- To recover postponed abort URs, use RECOVER POSTPONED. (If you specify AUTO for LIMIT BACKOUT, DB2 recovers the postponed URs after restart, and you do not have to enter a command.)

To determine what objects have backout work pending, use the DISPLAY DATABASE command. Objects with backout work pending are displayed with a 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.

Utility access to objects in AREST status: Utilities are not restricted by the AREST status, but any write claims held by postponed abort URs on the objects in AREST status prevent draining utilities from accessing that page set.

Restarting a DB2 member with conditions

As described in Section 4 (Volume 1) of *DB2 Administration Guide*, 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.

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, this use of conditional restart is not recommended. It is safer and you have better availability, to reroute work to another DB2 or by postponing backout processing rather than suffering the total outage necessary for a conditional restart.

If you do use conditional restart, it is necessary to stop all members of the group other than 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.

This section describes two procedures:

- “Procedure for cold starting a member (STARTRBA = ENDRBA)”
- “Procedure for other conditional restarts (STARTRBA ≠ ENDRBA)” on page 178

Procedure for cold starting a member (STARTRBA = ENDRBA)

Use this procedure for cold start; that is, when the STARTRBA = ENDRBA on the change log inventory's CRESTART statement.

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 DB2 was connected.
3. Resolve all data inconsistency problems resulting from the cold start. For more information about how to do this, see Section 4 (Volume 1) of DB2 Administration Guide.
4. After you have resolved the data inconsistencies resulting from the cold start, you can start all the other members and restart this one without ACCESS(MAINT).

Procedure for other conditional restarts (STARTRBA ≠ ENDRBA)

Use this procedure when you are truncating the log but are not doing 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 cold start. For more information about how to do this, see Section 4 (Volume 1) of *DB2 Administration Guide*.
4. After you have resolved the data inconsistencies resulting from the conditional restart, you can start all the other members and restart this one without ACCESS(MAINT).

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 DASD 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 LPL will not be able to access that page until the object is restarted.

You have to use the START DATABASE command with the SPACENAM option to make those pages available after DB2 has restarted.

Deferring recovery does not change restart time significantly.

Starting and stopping duplexing for a group buffer pool

This section describes how you can start and stop duplexing for a particular group buffer pool.

Starting duplexing

When duplexing starts, there is a period in which activity to the group buffer pools is quiesced. Thus, it is best to start duplexing during a period of low activity in the system.

There are two ways to start duplexing for a group buffer pool:

- 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 group buffer pool is currently allocated, then MVS can automatically initiate the process to establish duplexing as soon as you activate the policy. If the group buffer pool is not currently allocated, then the duplexing process can be initiated when the group buffer pool is allocated.
- Activate a new CFRM policy with DUPLEX(ALLOWED) for the structure. If the group buffer pool is currently allocated, use the following command to start the duplexing rebuild:

  ```
  SETXCF START,REBUILD,DUPLEX,STRNAME=strname
  ```

  If the group buffer pool is not currently allocated, wait until it is allocated before starting the duplexing rebuild.

While duplexing is being established, and for the entire time duplexing is in effect, a display of the structure shows structure as being in DUPLEXING REBUILD. The rebuild phase is called DUPLEX ESTABLISHED, which means that duplexing is truly active for the structure. See Figure 65 on page 244 for an example.

Stopping duplexing

To stop duplexing, you must first decide which instance of the group buffer pool is to remain as the surviving simplexed group buffer pool. If you have a choice in the matter, it is preferable to use the primary one as the surviving one. The primary group buffer pool has intact page registration information, whereas to switch to the secondary one, all the page registration is lost. Therefore, all locally cached pages revert to an invalid state and must be refreshed from the group buffer pool or DASD.

If you want to switch temporarily to one of the structures as a simplexed group buffer pool, 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 be the same as before.
2. Use the SETXCF STOP,REBUILD command, specifying KEEP=OLD to revert to using the primary structure as the simplexed group buffer pool, or KEEP=NEW to switch to the secondary instance as the simplexed structure. For example, the following command reverts to using the primary instance as the simplexed group buffer pool:

   ```
   SETXCF STOP,RB,DUPLEX,STRNAME=strname,KEEP=OLD
   ```

If you do not plan on reestablishing duplexing for the group buffer pool in the near future, then activate a new CFRM policy specifying DUPLEX(DISABLED) for the structure.

If you want to do maintenance on a coupling facility that has primary or secondary group buffer pool instances allocated in it, then use the SETXCF STOP,RB,DUPLEX command specifying the target CF, as documented in “Shutting down the coupling facility.”

Shutting down the coupling facility

Make a plan for those cases where 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 your structures to another coupling facility before shutting it down. This section gives some recommendations for how to handle this event for DB2. For other structures in the coupling facility, see the appropriate product documentation.

Consider the following:

1. Prepare for the move:
   - Make sure that you have enough room on the alternate coupling facility for all structures you intend to move.
   - Make sure that the preference list for the group buffer pool, SCA, and lock structures contains the alternate coupling facility information.

2. Move simplexed structures to the **new** coupling facility:

   ```
   SETXCF START,REBUILD,CFNAME=newcf,LOC=OTHER
   ```

   That command rebuilds all structures that allow rebuild onto the alternate coupling facility.

3. For duplexed structures, issue the command to deallocate the duplexed structures on the **target** coupling facility:

   ```
   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), MVS 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, MVS avoids automatially 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 into that third coupling facility.

4. For populating a coupling facility that is newly brought into service or is being returned to service after a maintenance operation, use this command:

   ```
   SETXCF START,REBUILD<POPULATECF,CFNAME=cfname
   ```

   To return the coupling facility to service, perform the following steps:

   a. Evacuate the target coupling facility.

      ```
      SETXCF START,REBUILD,CFNAME=cfname
      ```

   b. Do the maintenance operation on the target coupling facility and bring it back into the sysplex.

   c. Repopulate this coupling facility with the structures that were in it before the coupling facility was brought down.

      ```
      SETXCF START,REBUILD,POPULATECF,CFNAME=cfname
      ```

      The POPULATECF command requires OS/390 Version 2 Release 6 or subsequent releases.

Chapter 7. Performance monitoring and tuning

One of the main objectives of the data sharing function of the licensed program DB2 for OS/390 is to increase processing capacity while using the lower cost S/390 Parallel Sysplex technology. Work load capacity is increased by allowing many DB2 subsystems to access shared DB2 databases 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, there are also actions you can take to reduce the performance cost of data sharing. The purpose of this chapter is to describe briefly how data sharing locking and buffer management work and to offer some guidance about how to improve performance in the group.

This chapter describes the following topics:

- "Monitoring tools" on page 182
- "Improving the performance of data sharing applications" on page 183
- "Improving the response time for read-only queries" on page 186
- "Improving concurrency" on page 203
- "Tuning group buffer pools" on page 220
- "Access path selection in a data sharing group" on page 258

Setting performance expectations

With data sharing, as you increase throughput by spreading work across more systems, you can also expect to see some processing increases because of the cost of managing and controlling data sharing locking, buffer management, and data set management. It is important to remember, however, that those increases are limited to that percentage of your workload that is accessing inter-DB2 shared R/W data.

In this section, we describe some additional activities that DB2 performs for data sharing, and to which address space you can expect to see the cost of those activities charged:

- *ssnm*MSTR

  Activities that occur under the DB2 master address space include commit processing after updates, inserts, and deletes; logging; backout processing; and archiving. With data sharing, writes to the group buffer pool and the global unlocking that occurs during commit processing happen under SRBs in this address space.

- *ssnm*DBM1

  Activities that occur under this address space include prefetching, space management, and deferred write . With data sharing, castouts, prefetch interactions with the coupling facility, P-lock negotiation, updates to SYSLGRNX, and group buffer pool checkpoints occur under SRBs in this address space.

- IRLMPROC

For data sharing, additional activities that occur under IRLM's 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

  For data sharing, additional activities that occur under the allied agent's address space include global lock requests and requests to read from the group buffer pool.

Monitoring tools

This section describes the following tools:

“Using resource measurement facility (RMF®) reports”
“Using DB2 trace”
“Using DB2 PM” on page 183

Using resource measurement facility (RMF®) reports

The Resource Measurement Facility Version 5 (RMF) provides single system and Sysplex views by reporting on collected resource usage data:

- The Sysplex Summary report provides an integrated view of the entire Sysplex on one screen.
- The Response Time Distribution report gives details about the distribution of response times on a Sysplex level with the capability of zooming into a single system that indicates problems.
- The Coupling Facility reports have information about storage allocation, structure activity, and subchannel activity to allow you to plan for better resource utilization. See Figure 52 on page 215 and Figure 66 on page 245 for examples of Coupling Facility Activity reports.
- The Shared Device report provides information about how DASD and tape resources are shared among the different systems in the Sysplex.

Using DB2 trace

DB2 writes trace records to help you monitor events in the DB2 group. Many trace classes now include information about locking and use of the group buffer pool.

A single DB2 trace does not gather information about all the members of the group. All the trace commands (such as START TRACE and MODIFY TRACE) take effect only on the DB2 subsystem on which they are issued; they do not affect tracing activity on any other DB2 subsystems. In addition, the statistics interval for each subsystem is set by each member and the interval is not synchronized across the members of the group.

To gather data about group activity, as when auditing access to a particular table or detecting locking contention from processes on different DB2 subsystems, you must merge trace records in time sequence. To do this, use the value of the Sysplex Timer store clock contained in the standard header. Every trace record from a member of a data sharing group includes a data sharing header, which gives the DB2 group name and member name. This header is mapped by mapping macro DSNDQWHA. For more information about the format of trace records, see Appendix D (Volume 2) of *DB2 Administration Guide*.

Using DB2 PM

DB2 PM can monitor the use of shared DB2 resources such as the catalog and directory, group buffer pools, and databases for entire data sharing groups as well as 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
- Locking
- Statistics
- Audit

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 DB2 PM, you can have processor times reported in service units. This allows times from different CPC models in the group to be normalized before they are reported.

See Figure 53 on page 216 and Figure 54 on page 218 for examples of DB2 PM reports.

Improving the performance of data sharing applications

Many of the things you currently do for a single DB2 to improve response time or reduce processor consumption also hold true in the Parallel Sysplex. For example, most of the recommendations for reducing locking overhead described in “Improving concurrency” on page 203 are the same for a single system.

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. This section describes ways to make these resource-intensive applications run at their best in the Parallel Sysplex. The topics are:

See “Improving the response time for read-only queries” on page 186 for information about using query parallelism in a data sharing group. That section also includes information about governing resources in a data sharing group.

General recommendation

We suggest that you take advantage of data partitioning and that you design for parallel processing whenever possible. DB2 can use parallel processing effectively only when the data is partitioned. See Section 5 (Volume 2) of *DB2 Administration Guide* for more information about parallel processing for queries and utilities.

Migrating batch applications to data sharing

When migrating a batch application to data sharing, make sure 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 the significant portion of the elapsed time of a long-running batch application is because of processor resource consumption and for which you are currently having scheduling problems, consider running more than one copy of the same program in parallel. Each copy can be run 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 a CPC in the group that has the fastest single-CP speed. This approach is recommended only if the application does not access the coupling facility at a high intensity.

To avoid DASD contention, make sure the data partitions are placed on DASD devices with separate I/O paths. Consider using the PIECESIZE option of CREATE or ALTER INDEX to give you more control over the data set size of nonpartitioning indexes. See Chapter 6 of *DB2 SQL Reference* for more information about the PIECESIZE option.

Designing table spaces for heavy insert and update activity

For more information about both options described here, see “Reducing space map page contention” on page 229.

The MEMBER CLUSTER option: If your application does heavy sequential insert processing from multiple members in 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 then 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. See “Reducing space map page contention” on page 229 for more information about the MEMBER CLUSTER option of CREATE TABLESPACE.

The TRACKMOD option: When an application is rapidly updating a single table space, there can be contention on the space map pages as DB2 tracks changes. DB2 tracks these changes to help make incremental image copies run faster. With TRACKMOD NO, DB2 avoids tracking changes and thus avoids contention on the space map. This option is not good 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.

Migrating online applications

The response times for online applications are usually not degraded solely by limited processing power. They can most effectively use the Parallel Sysplex to run many transactions in parallel. Any slight increase in transaction elapsed time caused by running on the S/390 microprocessors is offset by running transactions across many CPCs. This high level of parallelism can use the available capacity of the Sysplex to deliver equivalent or higher levels of throughput compared to running on a high-end processor.

The release of MVS and coupling facility level can affect performance of applications that use sequential or list prefetch. See “Prefetch processing” on page 232 for more information.

Running utilities

For most DB2 utilities, processor resource consumption represents the significant portion of the elapsed time when running on an early generation of S/390 microprocessors . Two of the most significant examples are:

LOAD with many columns
RUNSTATS with non-index column statistics

Consider using the SAMPLE option of RUNSTATS to reduce processing costs. The key to letting processor-intensive utilities perform well is to partition the data. For more information about running utilities on separate partitions of data, see *DB2 Utility Guide and Reference*.

Where partitioning the data is not an option, consider running utilities on a CPC with a faster single-CP speed.

Using the resource limit facility (governor)

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. For information about using the resource limit facility to control queries using Sysplex query parallelism, see “Setting limits using the resource limit facility” on page 200.

Controlling the RLST

The commands used to control the RLST (DISPLAY RLIMIT, START RLIMIT, and STOP RLIMIT) affect only the DB2 on which the command is issued. The same holds true if you start the RLST at DB2 startup by using a system parameter.

Dropping objects in the resource limit facility

While an RLST is active in any DB2 in the group, you cannot drop any object associated with any active RLST.

Restrictions for STOP and START DATABASE

While an RLST is active in any DB2, you cannot enter a STOP DATABASE command for a database or table space that contains the active RLST, nor can you start the database or table space with ACCESS(UT).

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 (CP). For complex queries, run the query in parallel within a member of a data sharing group, and, with Sysplex query parallelism, use the full power of the data sharing group to process individual complex queries on many members of the data sharing group. The information in this section is based upon information about query parallelism in Section 5 (Volume 2) of *DB2 Administration Guide*, but this section emphasizes how you can use the full power of the data sharing group to run your complex queries.

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 distribute parts of the query to be processed on other members of the data sharing group.

Terminology: The DB2 that is attached to the application that issued the query is the *parallelism coordinator*. A member who helps process one of these parallel queries is an *assisting DB2* or *parallelism assistant*.

How data is returned to the parallelism coordinator: Data is returned to the parallelism coordinator in one of two ways:

- By work file

  If the query requires the use of a work file (because the data needs to be sorted, for example), then the parallelism coordinator can access the data from the parallelism assistants' work files. Each assistant writes to its own work file, and the coordinator can read the results from the assistants' work files.

- By cross-system coupling facility (XCF) links

  For data that does not require the use of a work file, XCF is used. XCF traffic can be transported using channel-to-channel connections between the CPCs, or by using the coupling facility.

The follow topics are described in this section:

- “Planning for Sysplex query parallelism” on page 187
- “Enabling Sysplex query parallelism” on page 190
- “Monitoring and tuning parallel queries” on page 194
- “Disabling Sysplex query parallelism” on page 202

Planning for Sysplex query parallelism

This section describes the following tasks:

- "Configuring the systems"
- "Setting Workload Management Goals"
- "Designing the database" on page 190

Configuring the systems

Before you can use Sysplex query parallelism, you must configure the members of the data sharing group to allow it. In summary:

- The original query must originate from a member for which YES is specified for the COORDINATOR field of installation panel DSNTIPK. The COORDINATOR subsystem parameter controls whether this DB2 can send parallel tasks out to other DB2s in the data sharing group. If the COORDINATOR parameter is not YES at run time, the query runs within a single DB2. This is reported in the DB2 PM accounting report in Figure 43 on page 197 in field **C**.
- To be considered as candidates for assisting the parallelism coordinator, the other members must have YES specified in the ASSISTANT field of installation panel DSNTIPK. The ASSISTANT subsystem parameter controls whether this DB2 can receive parallel tasks from another DB2 in the data sharing group. If this DB2 is the coordinator for a particular query, then its ASSISTANT parameter is not relevant.
- Put work files on shared DASD that is accessible from 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. DB2 assumes at run time that all work files are in the same numbered buffer pool.
- Define group-wide goals for workload management for work that originates on a particular DB2 and for work that is processed by that DB2 on behalf of another. See "Setting Workload Management Goals."
- Buffer pools that can assist with 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 be greater than 0, too. See "Buffer pool threshold for parallelism assistants" on page 192 for more information.
- For optimal use of processor resources, only one DB2 on a given CPC should be used as an assistance. There is no benefit to splitting a query across multiple members on the same CPC.

Setting Workload Management Goals

It is important to define how you want MVS to handle the work for Sysplex query parallelism. The task of classifying work that runs on the parallelism coordinator is the same as in the past. However, you must also classify work that runs on the assistant DB2. If you do not classify DB2 as an assistant, the following outcomes can occur:

- If you are running in compatibility mode, the part of the query that runs on the assistant has the same priority as the DB2 database services address space. This is probably not what you want.

- If you are running in goal mode, the 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 that this DB2 assists, you must:

1. Define a service class that is used for queries that run on an assistant. This service class can have the same goals as those for the coordinator, but the name must be different unless you are running with OS/390 Release 3 or subsequent releases. Alternatively, you can apply modified goals to the assistants to take into account the numerous amount of enclaves per assistant for a single query. See "Example: setting goals for the parallelism assistants" on page 189 for information about defining a service class.
2. If you run in compatibility mode, use the IEAIPS*xx* PARMLIB member to indicate which performance group should control the query work. Also, use the IEAICS*xx* PARMLIB member to set the SRVCLASS parameter to the service class defined in the policy, and associate it with a performance group. The subsystem type you use is DB2.

   As described in "How period switches work on parallelism assistants," any work coming into an assistant starts in period 1.
3. Install the service definition and activate the service policy.

For more information about using workload management, see *OS/390 MVS Planning: Workload Management*.

How period switches work on parallelism assistants

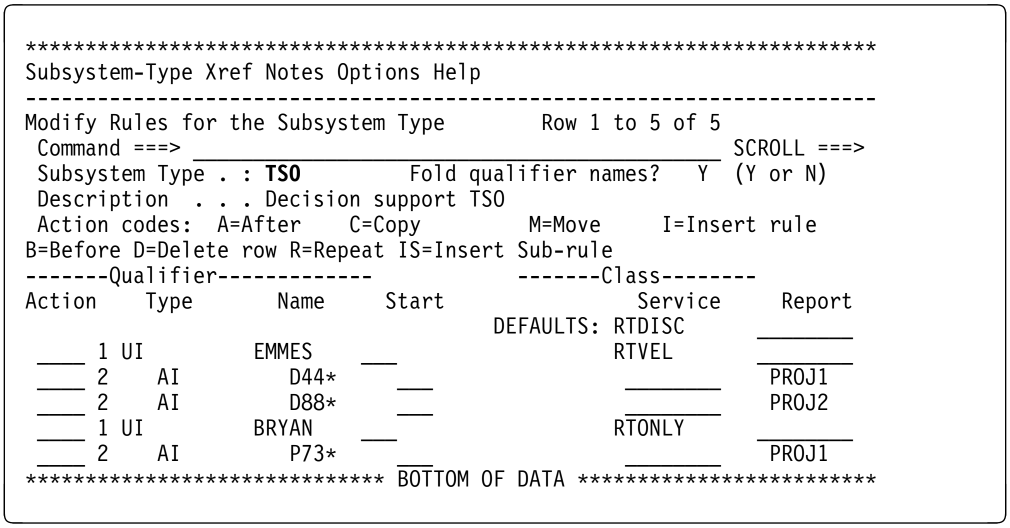
For work performed within a single system, work is governed by specifying *periods*. Each period has a service unit value and a priority attached to it. If you run in WLM compatibility mode, you must specify these values in the IEAIPS*xx* PARMLIB member. If you run in WLM goal mode, those values are assigned by the system. When a piece of work exceeds the service units for a given period, then a period switch occurs and the piece of work is switched to the next period, which probably has a lower priority attached to it. The dispatching priority is reevaluated after a period switch.

With Sysplex query parallelism, the query work for a particular query always starts out in the first period on each assistant with no accumulated service units.

Recommendation: Initially, we recommend that you classify work for the assistants using only one performance period. You can add more performance periods after you monitor the system.

Example: setting goals for the parallelism coordinator

Although this work has probably already been done for the coordinating DB2, this section illustrates the relationship between service classes for the parallelism coordinator and for the assistant. Figure 33 on page 189 is a an example of how you might classify complex queries that originate from TSO. Note that SUBSYSTEM TYPE is TSO.



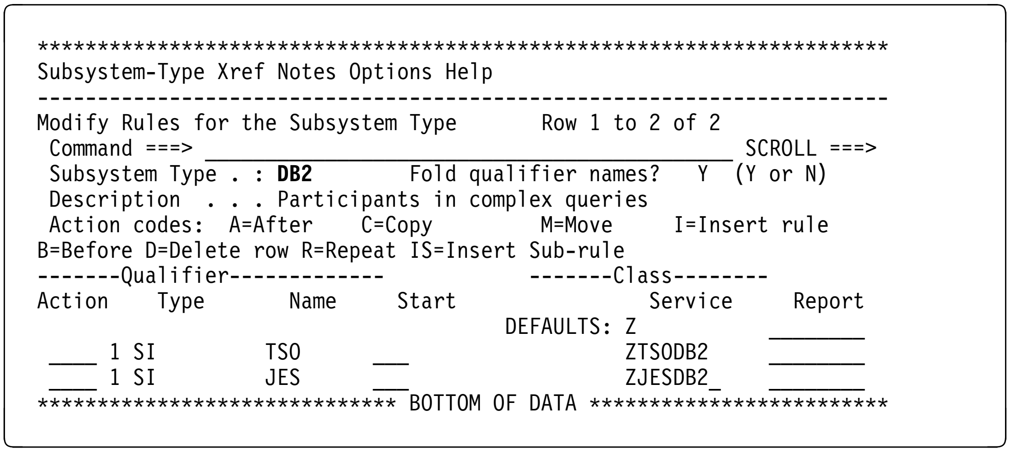
```
***********************************************************************
Subsystem-Type Xref Notes Options Help
-----------------------------------------------------------------------
Modify Rules for the Subsystem Type        Row 1 to 5 of 5
 Command ===> ____________________________________________ SCROLL ===>
 Subsystem Type . : TSO        Fold qualifier names?   Y  (Y or N)
 Description  . . . Decision support TSO
 Action codes:  A=After    C=Copy        M=Move     I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
-------Qualifier-------------           -------Class--------
Action    Type       Name     Start                Service     Report
                                       DEFAULTS: RTDISC      ________
 ____ 1 UI         EMMES    ___                  RTVEL       ________
 ____ 2    AI         D44*     ___                ________    PROJ1
 ____ 2    AI         D88*     ___                ________    PROJ2
 ____ 1 UI         BRYAN    ___                  RTONLY      ________
 ____ 2    AI         P73*     ___                ________    PROJ1
***************************** BOTTOM OF DATA *************************
```

Figure 33. Defining workload management information for the parallelism coordinator

The service classes shown in Figure 33 are used for queries that originate and run on a particular DB2. If you are running on a release of OS/390 before release 3, any part of the same query that is shipped to another DB2 must use a different service class name, because WLM cannot manage enclave work and address space work with the same name. Report classes can be the same.

Example: setting goals for the parallelism assistants

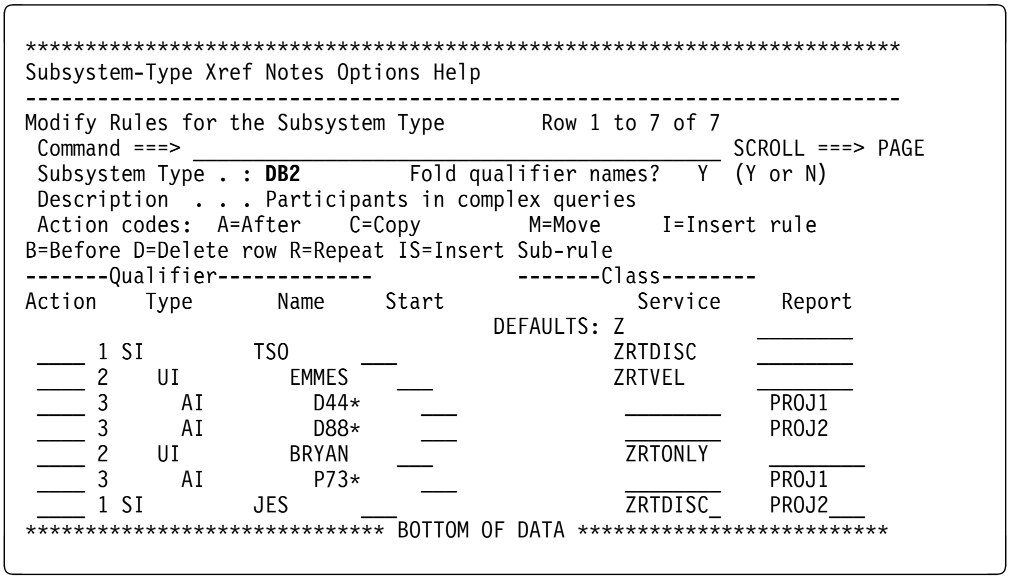
Figure 34 shows one way to classify work on the assistant. In Figure 34, all query work originating from a given environment (such as TSO) has the same goals, but each work environment has its own goals.



```
***********************************************************************
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-------------           -------Class--------
Action    Type       Name     Start                Service     Report
                                       DEFAULTS: Z           ________
 ____ 1 SI         TSO      ___                  ZTSODB2     ________
 ____ 1 SI         JES      ___                  ZJESDB2_    ________
***************************** BOTTOM OF DATA *************************
```

Figure 34. 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.

Figure 35 on page 190 shows another way that work can be classified. It more clearly illustrates the relationship between the work classified on the coordinator in Figure 33 and work classified on the 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-------------            -------Class--------
Action    Type       Name     Start                Service     Report
                                       DEFAULTS: Z           ________
 ____ 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 35. Classifying work for the parallelism assistants

Designing the database

Ideally, you want as much partitioning of table spaces as possible and as much separation of I/O as possible. Section 5 (Volume 2) of *DB2 Administration Guide* contains some detailed guidance on how to partition to achieve your performance goals.

Define table spaces and indexes with GBPCACHE CHANGED: Because it is unlikely that DB2s processing a large query will repeatedly read the same pages, there is no need to cache those pages in the group buffer pool. Define the relevant table spaces and indexes with GBPCACHE CHANGED (the default).

Enabling Sysplex query parallelism

This section describes how to enable parallelism within your application and within the system.

There is no change for how to enable parallel processing within your application. To **allow** parallel processing:

If query is:	**Use:**	**In:**
Static	DEGREE(ANY)	BIND or REBIND subcommand
	`PARALLEL DEGREE ==> ANY`	DB2I DEFAULTS panel for BIND or REBIND PACKAGE or PLAN
Dynamic	`SET CURRENT DEGREE = 'ANY'`	A previous SQL statement

Note: 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 CS or UR and use a LOCK TABLE *table-name* IN SHARE MODE statement before the query.

See Section 5 (Volume 2) of *DB2 Administration Guide* for more information about restrictions on queries.

Coordinator and assistant subsystem parameters

For a DB2 to be a coordinator of parallelism, you must specify YES in the COORDINATOR field of installation panel DSNTIPK. To allow a DB2 to assist with parallelism, you must specify YES in the ASSISTANT field 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.

Easy way to see coordinators and assistant parameters: The command DISPLAY GROUP now has a DETAIL option that allows you to see the COORDINATOR and ASSISTANT subsystem parameters for all active members of the group. For example, the following command:

```
-DB1G DISPLAY GROUP DETAIL
```

Displays output similar to Figure 36.

```
DSN7100I -DB1G DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNDB0G ) GROUPLEVEL(610)
                                     GROUP ATTACH NAME(DB0G)
--------------------------------------------------------------------
DB2                                    DB2  SYSTEM   IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL  NAME     SUBSYS IRLMPROC
-------- --- ----   --------  -------- ---  -------  ----   --------
DB1G       1 DB1G   -DB1G     ACTIVE   610  MVS1     DJ1G   DB1GIRLM
DB2G       2 DB2G   -DB2G     ACTIVE   610  MVS2     DJ2G   DB2GIRLM
DB3G       3 DB3G   -DB3G     QUIESCED 610  MVS3     DJ3G   DB3GIRLM
DB4G       4 DB4G   -DB4G     ACTIVE   610  MVS4     DJ4G   DB4GIRLM
--------------------------------------------------------------------
DB2      PARALLEL    PARALLEL
MEMBER   COORDINATOR ASSISTANT
-------- ----------- ---------
DB1G             YES        NO
DB2G             YES       YES
DB3G            ****      ****
DB4G              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(DSNDB0G )
DSN9022I -DB1G DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 36. DISPLAY GROUP with DETAIL option

See *DB2 Command Reference* for more information.

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 coordinator with processing parallel queries. Figure 37 shows the relationship among the various buffer pool thresholds.

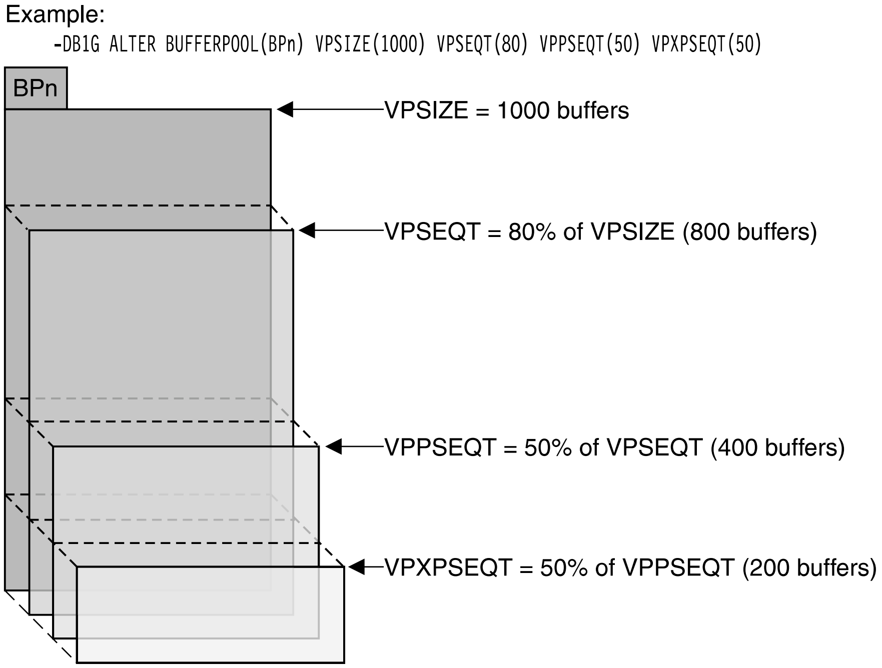

Example:
-DB1G ALTER BUFFERPOOL(BPn) VPSIZE(1000) VPSEQT(80) VPPSEQT(50) VPXPSEQT(50)
BPn
VPSIZE = 1000 buffers
VPSEQT = 80% of VPSIZE (800 buffers)
VPPSEQT = 50% of VPSEQT (400 buffers)
VPXPSEQT = 50% of VPPSEQT (200 buffers)

Figure 37. Relationships of buffer pool thresholds. The threshold 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: Let's look at how the buffer pool parallel allocation threshold values might be set up for two different configurations.

Figure 38 on page 193 shows how the online systems (IMS and CICS) are configured to send all of their parallel queries to the query processor subsystems on CPC7 and CPC8, but they do not assist other members.

- The COORDINATOR subsystem parameter is YES
- The ASSISTANT subsystem parameter is NO

The TSO and BATCH systems can run their own queries in parallel and can send query work to the query processors, but they do not assist other members.

- The COORDINATOR subsystem parameter is YES
- The ASSISTANT subsystem parameter is NO

The query processors have both ASSISTANT and COORDINATOR set to YES.

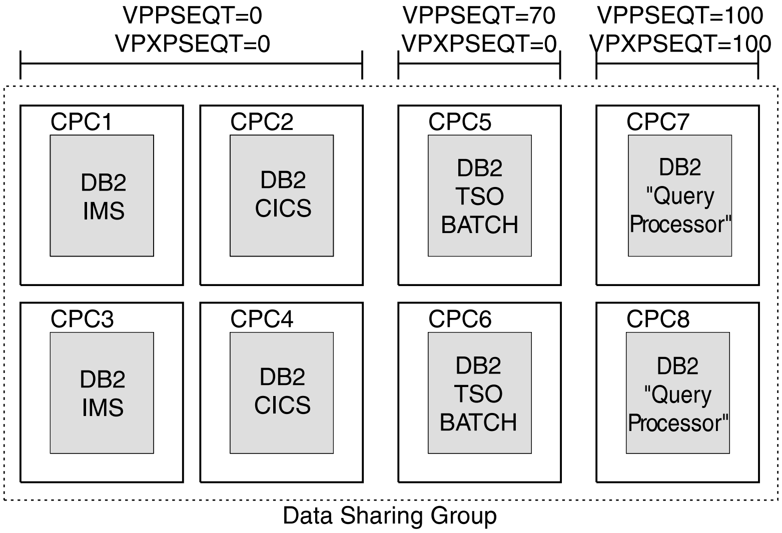


Figure 38. Separate query processor configuration. In this configuration, the transaction systems (IMS and CICS) are configured for maximum transaction throughput. They do not assist in processing queries for other members (ASSISTANT=NO), and no query parallelism takes place on those subsystems.

Figure 39 shows a data sharing group in which all members can coordinate and assist with parallel queries. All COORDINATOR and ASSISTANT subsystem parameters are set to YES.

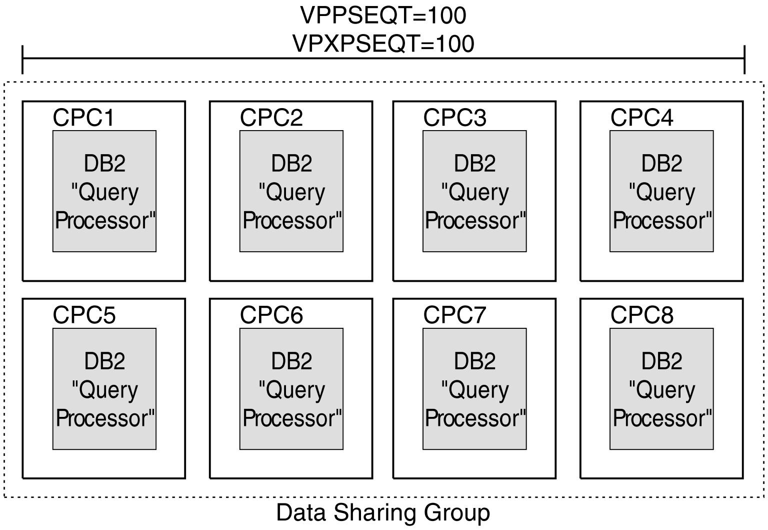


Figure 39. Dedicated query processor configuration. In this configuration, the entire data sharing group is a set of "query processors." The virtual pool sequential threshold (VPSEQT) is presumably set very high (perhaps 100).

Displaying the buffer pool thresholds: The DB2 command DISPLAY BUFFERPOOL displays all buffer pool thresholds, including the assisting parallel sequential threshold, as shown in Figure 40 on page 194.

```
-DB1G DISPLAY BPOOL(BP1)
DSNB401I -DB1G BUFFERPOOL NAME BP1, BUFFERPOOL ID 1, USE COUNT 76
DSNB402I -DB1G VIRTUAL BUFFERPOOL SIZE = 20000 BUFFERS
             ALLOCATED       =    20000   TO BE DELETED   =        0
             IN-USE/UPDATED  =    16345
DSNB403I -DB1G HIPERPOOL SIZE = 1000 BUFFERS, CASTOUT = YES
             ALLOCATED       =        0   TO BE DELETED   =        0
             BACKED BY ES    =        0
DSNB404I -DB1G THRESHOLDS -
            VP SEQUENTIAL       =100   HP SEQUENTIAL          = 75
            DEFERRED WRITE      = 85   VERTICAL DEFERRED WRT  = 80
            PARALLEL SEQUENTIAL = 50   ASSISTING PARALLEL SEQT=100
DSN9022I -DB1G DSNB1CMD '-DISPLAY BPOOL' NORMAL COMPLETION
```

Figure 40. Displaying buffer pool thresholds.. In this particular buffer pool, 50 percent 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.

Monitoring and tuning parallel queries

This section describes the following:

“Using DISPLAY THREAD”
“Using a performance monitor” on page 195
“How DB2 reports what it did” on page 195
“Monitoring processor use” on page 196
“Improving response time” on page 197
“Controlling resources used by parallel operations” on page 200

Using DISPLAY THREAD

Although DISPLAY THREAD is not a group-wide command, it can display information about parallel tasks associated with their originating task. If a thread is using Sysplex query parallelism, issuing a DISPLAY THREAD on any assisting DB2 gives you the thread token of the originating task and the query coordinator's member name. The status PT is used in the display to indicate parallel tasks. All parallel tasks are displayed immediately after their corresponding originating thread.

See Chapter 2 of *DB2 Command Reference* for information about the syntax of the command DISPLAY THREAD.

Display on the parallelism coordinator: This example shows an allied, nondistributed originating thread (TOKEN=30) that is established (plan allocated) along with all of its parallel tasks, which are running on DB2 members DB1G and DB2G. Because the originating thread is running on DB1G, it is the coordinating DB2.

```
- 17.08.44          -DB1G DISPLAY THREAD(*)
- 17.08.44 STC00090  DSNV401I -DB1G DISPLAY THREAD REPORT FOLLOWS -
- 17.08.44 STC00090  DSNV402I -DB1G ACTIVE THREADS -
- NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
- BATCH    T  *     1 PUPPYDML     USER001  DSNTEP2  0025    30
-          PT *   612 PUPPYDML     USER001  DSNTEP2  002A    35
-          PT *   545 PUPPYDML     USER001  DSNTEP2  002A    34
-          PT *   432 PUPPYDML     USER001  DSNTEP2  002A    33
-          PT *   443 PUPPYDML     USER001  DSNTEP2  002A    32
-          PT *   252 PUPPYDML     USER001  DSNTEP2  002A    31
- DISPLAY ACTIVE REPORT COMPLETE
- 17.08.45 STC00090  DSN9022I -DB1G DSNVDT '-DB1G DISPLAY THREAD' NORMAL COMPLETION
```

Figure 41. Display on coordinator

Display on the assistant: The PARALLELISM COORDINATOR field tells you which DB2 is the coordinator. The ORIGINATING TOKEN field tells you lets you associate a parallel task with its originating task on the coordinator.

```
- 17.10.12          -DB2G DISPLAY THREAD(*)
- 17.10.12 STC00044  DSNV401I -DB2G DISPLAY THREAD REPORT FOLLOWS -
- 17.10.12 STC00044  DSNV402I -DB2G ACTIVE THREADS -
- NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
- BATCH    PT *   641 PUPPYDML     USER001  DSNTEP2  002D     6
-  V443-PARALLELISM COORDINATOR=DB1G, ORIGINATING TOKEN=30
- BATCH    PT *    72 PUPPYDML     USER001  DSNTEP2  002D     7
-  V443-PARALLELISM COORDINATOR=DB1G, ORIGINATING TOKEN=30
- BATCH    PT *   549 PUPPYDML     USER001  DSNTEP2  002D     8
-  V443-PARALLELISM COORDINATOR=DB1G, ORIGINATING TOKEN=30
- BATCH    PT *   892 PUPPYDML     USER001  DSNTEP2  002D     9
-  V443-PARALLELISM COORDINATOR=DB1G, ORIGINATING TOKEN=30
- BATCH    PT *    47 PUPPYDML     USER001  DSNTEP2  002D     10
-  V443-PARALLELISM COORDINATOR=DB1G, ORIGINATING TOKEN=30
- DISPLAY ACTIVE REPORT COMPLETE
- 17.10.12 STC00044  DSN9022I -DB2G DSNVDT '-DISPLAY THREAD' NORMAL
-  COMPLETION
```

Figure 42. Display on assisting DB2

Using a performance monitor

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.

How DB2 reports what it did

DB2 reports information about parallelism at two times: At bind time (what it *plans* to do) and at run time (what it actually did). This section describes what happens at bind and run time and what can happen to cause the parallel degree to change at run time.

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 CPs on each DB2 when it determines the parallel degree.

The access path that DB2 plans to use is shown in the EXPLAIN output. The PARALLELISM_MODE column contains “X” to indicate that Sysplex query parallelism is chosen for the explainable statement. If only one member of the data sharing group is active at bind time, then it is still possible for DB2 to 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 coordinating DB2 at run time in the QXREPOP1 field of the statistics and accounting traces. (See **F** in Figure 43 on page 197.)
- There is not enough buffer space to handle the optimal degree of parallelism. This event is recorded by the coordinating DB2 at run time in the QXREPOP2 field of the statistics or accounting traces. (See **G** in Figure 43 on page 197.)

To determine whether work was sent to assisting DB2s, use the accounting trace. (See **A** in Figure 43 on page 197 to see how DB2 PM 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 executed, and it also indicates when a particular buffer pool is constrained on a particular member. IFCIDs 0222 and 0231 now include DB2 member names for 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 around the group. For more information, see “Is the buffer pool too small?” on page 197.

Monitoring 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. DB2 PM reports this total time in its accounting report and normalizes the time to the processor size of the originating task. See Figure 43 on page 197 for an example of an accounting report from DB2 PM.

```
⋮
QUERY PARALLEL.     TOTAL
---------------  --------
MAXIMUM MEMBERS  A      2
MAXIMUM DEGREE         13
GROUPS EXECUTED         1
 RAN AS PLANNED         0
 RAN REDUCED     B      1
 ONE DB2 COOR=N   C     0
 ONE DB2 ISOLAT         0
 SEQ - CURSOR           0
 SEQ - NO ESA           0
 SEQ - NO BUF D         0
 SEQ - ENCL.SER         0
MEMB SKIPPED(%)  E      0
DISABLED BY RLF   NO
REFORM PAR-CONF   F     0
REFORM PAR-BUF     G    0
⋮
```

Figure 43. Partial accounting trace that shows Sysplex query parallelism

Improving response time

When DB2 executes a query on more than one DB2 member, the query elapsed time should improve when compared to executing the query within one 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 elapsed time of a query, look at 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.

Here are some things to look for:

- “Is the buffer pool too small?”
- “Is there I/O contention?” on page 198
- “Is there lock contention?” on page 199
- “Is there XCF signalling contention?” on page 199
- “Is there inter-DB2 read/write interest?” on page 200

Is the buffer pool too small?: An indicator of buffer pool shortages are non-zero values in the QXREDGRP (**B** in Figure 43) and QXDEGBUF (**D**) 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. It might be necessary to increase the size of the buffer pools or increase the number of buffers available to assist with processing query work.

For more information about determining buffer pool size for parallel work, see Section 5 (Volume 2) of *DB2 Administration Guide*.

Reformulating the degree: When buffer pool resources are scarce, DB2 reformulates the parallel degree (indicated in **G**) and comes up with a distribution scheme that works best with the reformulated degree. **G** is incremented once for each parallel group that had to be reformulated.

The percentage of members that were unable to perform some or all of the work that was originally planned for is indicated in **E**.

The interaction of these two fields is best illustrated by an example. Assume that you have a 4-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 can only process 5 parallel tasks each, DB2 can reduce the parallel degree to 30, incrementing **G** once and setting **E** to 50%.

Work file buffer pools: Don't forget to set the parallel sequential threshold for work file buffer pools. DB2 assumes that you have all your work files of the same size (4 KB or 32 KB) in the same buffer pool number on all DB2s and 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), then move on 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 DB2 PM report), look to see if the number of I/Os per commit has increased. If so, consider some form of buffer pool tuning. By doing such things as increasing the size of the pool, isolating the data into a separate pool, or by tuning thresholds, you might be able to reduce the number of I/Os and speed up the remaining I/Os in the system.

   If that doesn't solve the problem, move on to pinpointing 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 breaks down where in the I/O subsystem the time is spent. The breakdown 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've determined where the problem is occurring, you can take steps to bring relief. This could be a matter of adding to the I/O subsystem by doing such things as adding extra channel paths to the DASD controller, adding storage directors in the DASD controller, adding extra cache/NVS in the controller, or adding extra DASD volumes. More typically, it means 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 DASD so that I/O operations are distributed as evenly as possible. Make sure 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 doing more than desirable I/O operations on a particular logical partition, or there is significant contention on the DASD volume or channel path.

If there is an uneven distribution of work that is causing the I/O problems, consider moving data that has low activity with 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 more parts, 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's possible. If it is not possible to run with UR, the next best thing is to take advantage of lock avoidance by running with ISOLATION (CS) and specifying CURRENTDATA (NO). This can minimize the locking effect.

Is there XCF signalling contention?

The resource measurement facility (RMF) of MVS provides an XCF Activity Report that contains useful measurement data for analyzing the performance of the XCF signalling service and doing capacity planning for an MVS system in a sysplex. The report shows the data collected on a system during sysplex processing. Each system collects its own data, and RMF on each system produces reports only about that system's data. It might be necessary to run the RMF reports on two or more systems to get data for corresponding outbound and inbound signalling 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 sysplex.

IRLM will report the number of retries in case it cannot deliver a message for Sysplex query parallelism via XCF. This information is reported in the IRLM Exception (EXP) trace and can be used to tune XCF performance.

For more information about monitoring XCF activity, see *OS/390 MVS Setting Up a Sysplex*.

Is there inter-DB2 read/write interest?: Because DB2 can split query processing onto different DB2 subsystems, 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 virtual pool before processing the query on other DB2 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 on the parent (table space or partition) lock. If all the TS locks are IS, then no child locks are propagated. However, if there is an IX lock on the table space, which indicates read/write interest, all the 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 define the table space with LOCKPART YES and bind the plan or package with ACQUIRE(USE). For more information, see “Using the LOCKPART option for partitioned table spaces” on page 209.

Controlling resources used by parallel operations

There are several different ways to control how much system resource is used for processing queries using Sysplex query parallelism:

- “Priority”
- “Amount of buffer space allocated for parallel processing”
- “Setting limits using the resource limit facility”

Priority: MVS workload management can be used to control the priority of parallel query work within a subsystem. You must pay special attention to the priority of query work coming into an assisting DB2. “Setting Workload Management Goals” on page 187 has more information about how to do this.

Amount of buffer space allocated for parallel processing: You can control how much of your total buffer pool space can be used for parallel processing in general, and for Sysplex query parallelism specifically.

- The parallel threshold (VPPSEQT), determines the amount of space for all parallel processing.
- The assisting parallel sequential threshold (VPXPSEQT) determines what subset of the parallel space can be used to assist queries originating from another member of the data sharing group.

Response time can degrade if these thresholds are set too low.

Setting limits using the resource limit facility

DB2's resource limit facility can help control dynamic queries that use Sysplex query parallelism. Resource limits are local in scope, which lets DB2 ensure that a specific dynamic query does not overrun any DB2 subsystem. Members of a data sharing group can share the same RLST, or each member can have its own RLST. In either case, DB2 honors the limits that are specified on that DB2, no matter how many tasks are included in the statement.

Figure 44 on page 201 shows how this works.

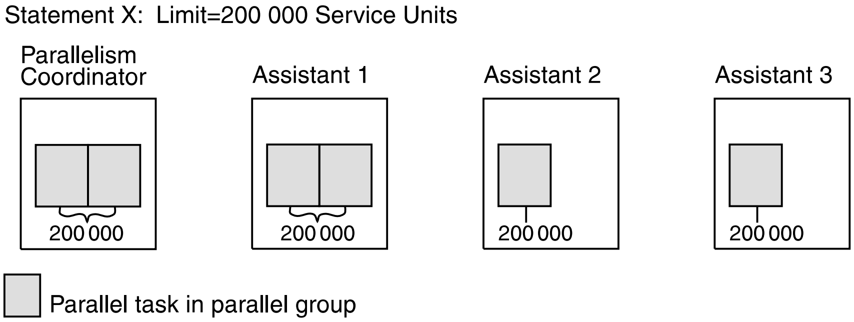


Figure 44. Governing a dynamic query running on four members. This examples assumes that all members share the same RLST. Statement X is not allowed to consume over 200000 service units on any DB2.

Governing statements with more than one parallel group: If more than one parallel group processes the work for a query on a member, each parallel group can run up the service unit limit set for that member. On the coordinator, things work differently because *all* parallel groups, including the originating task, are governed by the limit on the coordinating member. Figure 45 shows how this works.

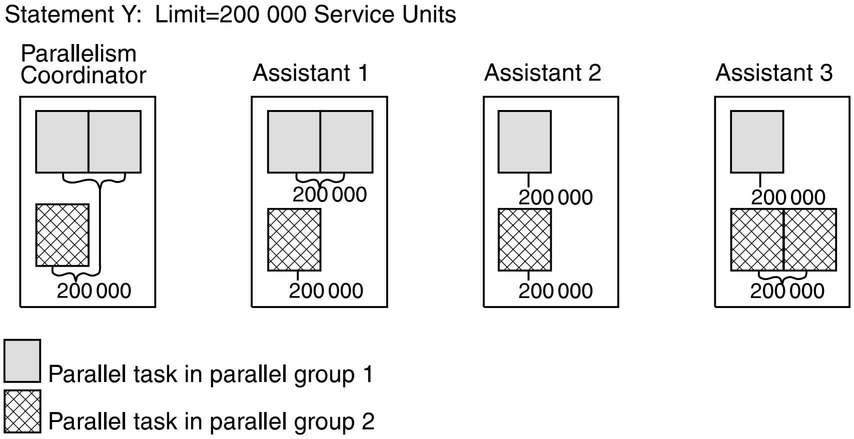

Statement Y: Limit=200 000 Service Units
Parallelism Coordinator
Assistant 1
Assistant 2
Assistant 3
200 000
200 000
200 000
200 000
200 000
200 000
200 000
Parallel task in parallel group 1
Parallel task in parallel group 2

Figure 45. A Query with two parallel groups running at the same time. This example assumes that members are sharing the same RLST. No parallel group that is part of statement Y can run beyond the limit of 200000 service units. On the parallelism coordinator, all tasks in all parallel groups run up to the limit specified for the statement.

Queries using 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 executed more than once: If the same statement executes more than once in an application, the limit is accumulated only on the coordinating DB2. On the assisting DB2s, the limit is reset each time the statement executes.

Disabling Sysplex query parallelism

In addition to existing 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 that are described in “Buffer pool threshold for parallelism assistants” on page 192.

You can also disable Sysplex query parallelism for a single dynamic query by using a value of '5' for the resource limit facility (governor).

Table 31 summarizes these controls.

Table 31 (Page 1 of 2). Controls for Sysplex query parallelism

Control	Checked at...	Effect
COORDINATOR subsystem parameter	Bind and run time	If COORDINATOR is NO, parallelism is restricted to a single DB2. (See QXCOORNO field of IFCID 0002 for cases in which the parameter was changed between bind and run time.) Changes from NO to YES require that plans or packages be rebound before they are considered for Sysplex query parallelism.
ASSISTANT subsystem parameter	Bind and run time	If ASSISTANT is NO, this member is not 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	If 0, no type of query processing is allowed on this DB2.
Assisting parallel sequential threshold (VPXPSEQT)	Run time	If 0, this DB2 is not considered to be an assistant for any query 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	Affects dynamic queries only. The query is not considered for Sysplex query parallelism.
BIND option DEGREE (1)	Bind time	Disables query parallelism for static queries.

Table 31 (Page 2 of 2). Controls for Sysplex query parallelism

Control	Checked at...	Effect
Special register SET CURRENT DEGREE='1' or a 1 in the CURRENT DEGREE field of installation panel DSNTIPF	Run time	Affects dynamic queries only. Disables all query parallelism.

Improving concurrency

This section describes briefly how transaction locking works in DB2 data sharing and some actions you can take to reduce locking overhead:

- “Global transaction locking”
- “Tuning deadlock and timeout processing” on page 210
- “Tuning your use of locks” on page 206
- “Monitoring DB2 locking” on page 214
- “Changing the size of the lock structure” on page 219

Data sharing also introduces a new type of lock, called a *physical lock* (P-lock). But, P-locks are related to caching, not concurrency, and they use different mechanisms than the transaction locks you are familiar with in DB2. See “P-locking” on page 227 for information about P-locks. Transaction locks are often called *logical locks* (L-locks) in data sharing.

Global transaction locking

With data sharing, there is concurrency control both within a specific DB2 and among all DB2s in the DB2 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 MVS's cross-system extended services (XES) and the lock structure in the coupling facility.

Locking optimizations

DB2 has optimizations in place to reduce the necessity to go beyond the local IRLM whenever possible:

- **Explicit hierarchical locking**, described in “Explicit hierarchical locking” on page 204 makes certain optimizations possible. When there is no inter-DB2 R/W interest in an object, it is possible to avoid processing certain locks beyond the local IRLM.

- If there is a single DB2 with update interest and multiple DB2s with read-only interest, DB2 propagates fewer locks than when all DB2s have update interest in the page set.

- All locks that go beyond the local IRLM are owned by the subsystem, not by an individual work unit. This allows for another optimization. Only the most restrictive lock mode for an object on a given subsystem must be propagated to XES and the coupling facility. A new lock that is equal to or less restrictive than one currently being held is not propagated. “A locking scenario” on page 205 gives an example of this type of optimization.

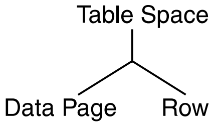
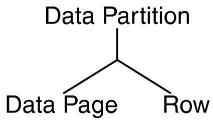
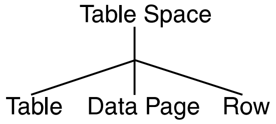
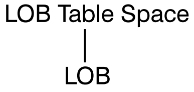
- When the lock structure is allocated in a coupling facility of CFLEVEL=2 or higher, IRLM can release many locks with just one request to XES. This can occur after a transaction commits and has two or more locks that need to be unlocked in XES. It also can occur at DB2 shutdown or abnormal termination when the member has two or more locks that need to be unlocked.

To see if DB2 is using this optimization, the statistics trace should show that there are significantly fewer unlock requests propagated to XES than lock requests, as shown in Figure 53 on page 216.

Explicit hierarchical locking

When sharing data, DB2 uses explicit hierarchical locking to determine whether it is necessary to propagate L-locks beyond the local IRLM to XES and the coupling facility. There is a performance advantage to granting lock requests locally. Explicit hierarchical locking allows IRLM to grant child locks locally when there is no inter-DB2 R/W interest on the parent. Explicit hierarchical locking is based on the lock hierarchy shown in Table 32.

Table 32. Lock hierarchy. Indexes are not included in the hierarchy because their index pages are protected by locks on the corresponding data.

Simple Table Space	Partitioned Table Space	Segmented Table Space	LOB Table Space
			

The top object in the hierarchy is a *parent*, and everything below a parent is a *child*. (A child can be the parent of another child.) While the 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, if you use LOCKPART YES, each locked partition is a parent of the child locks held for that partition. If you use LOCKPART NO (the default), the last data partition is the parent lock for all child locks. For more information about using the LOCKPART clause, see “Using the LOCKPART option for partitioned table spaces” on page 209.

Locks on the children are propagated depending on the compatibility of the maximum lock held on a parent with locks requested by other DB2s in the group for that parent.

Table 33 shows which conditions cause the child locks to be propagated. “A locking scenario” on page 205 describes explicit hierarchical locking in action.

Table 33. Determining when child locks are propagated to XES

Maximum lock mode of my member is ...	And the maximum lock mode of other members is...	Are X children propagated?	Are S children propagated?	Are U children propagated?
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	No
IX	IX	Yes	Yes	Yes
IX, SIX	None	No	No	No

Relationship between L-locks and P-locks: L-locks and P-locks are managed independently of one another. It is possible to have inter-DB2 interest on an L-lock but not on a P-lock that is held on the same object, and it is also possible to have inter-DB2 interest on the P-lock but not the L-lock. The inter-DB2 interest on the parent *L-lock* controls the propagation of the child L-locks. The inter-DB2 interest on the page set *P-lock* controls GBP-dependency.

Therefore, it is possible to propagate child L-locks for an object that is not GBP-dependent. For example, it is possible that an application bound with ACQUIRE(ALLOCATE) could have an IS L-lock on a table space without yet having opened the table space. The table space therefore has no P-lock yet on that one member. If another member opens the table space with an IX lock, its child L-locks have to be propagated, even though the table space is not GBP-dependent (no page set P-lock on the first member).

Conversely, it is also possible to have a GBP-dependent object for which child L-locks are not propagated. If an application is bound with RELEASE(COMMIT), for example, it might have released all its L-locks and write claims to a table space. However, until the table space switches from R/W to R/O, that table space is considered GBP-dependent. But child L-locks from another member can be resolved by its local IRLM.

A locking scenario

Figure 46 shows an example of locking activity between two members of a data sharing group.

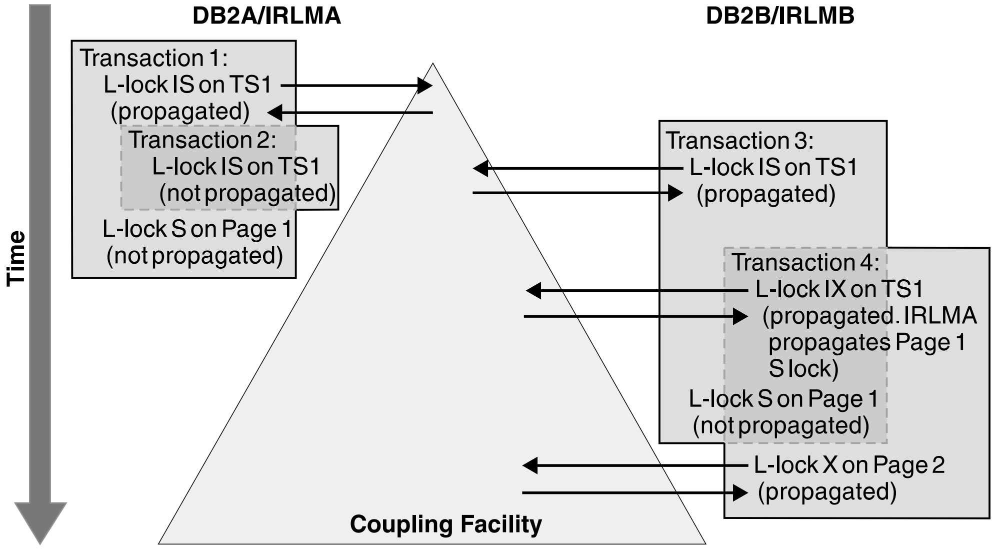


Figure 46. A lock propagation scenario

In the figure:

1. Transaction 2's L-lock on TS1 is not propagated because that DB2 already holds a lock on that object of an equal restriction. (Transaction 3's L-lock on TS1 is propagated, because that lock is from another DB2 subsystem.)
2. Transaction 1's child L-lock on Page 1 is not propagated at the time it was requested because its parent lock is IS on both subsystems. That is, there is no inter-DB2 R/W interest on the parent.

3. When Transaction 4 upgrades its lock to IX on TS1, its X lock on Page 2 must be propagated because there is now inter-DB2 R/W interest on the parent. Also, Transaction 1's child lock on Page 1 must be propagated.

   Transaction 1's child lock is propagated under an IRLM SRB, not the transaction's TCB. This propagation is counted in the statistics report as an asynchronous propagation, as shown in **B** in Figure 53 on page 216.

Determining 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 reflect the effects of explicit hierarchical locking and other locking optimizations. See “Checking for transaction locking optimizations” on page 218 for more information.

Tuning your use of locks

Most recommendations for reducing lock contention and locking costs in a single system hold true when sharing data, as well. This section reiterates some general recommendations and emphasizes the following:

- “Avoid false contention” on page 207
- “Reduce the time needed to resolve contentions” on page 209
- “Using the LOCKPART option for partitioned table spaces” on page 209

There is also information about:

- “Monitoring DB2 locking” on page 214
- “Changing the size of the lock structure” on page 219

General recommendations

To reduce locking contention, use the same tuning actions that are in place for single-DB2 processing today, as described in Section 5 (Volume 2) of *DB2 Administration Guide*. Here is a summary:

- Use partitioned table spaces.
- Use page locking.

  Although row locking can be used to increase concurrency in some cases, you must weigh that benefit 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 but to avoid the additional data sharing lock overhead is to define the table space with MAXROWS 1. See *DB2 SQL Reference* for more information.

- If your applications can tolerate reading uncommitted data, use an ISOLATION level of uncommitted read (UR).
- If you cannot use ISOLATION(UR), take advantage of lock avoidance whenever possible 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.

If you use the BIND option RELEASE(DEALLOCATE) for objects that don't 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. It can also reduce the amount of false contention, described in “Avoid false contention” on page 207 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 as frequently used.

- Design for selective partition locking, and bind with ACQUIRE(USE). For more information about selective partition locking, see Section 2 of *DB2 Administration Guide*.

Avoid false contention

The coupling facility lock structure has two parts to it: a lock table used to determine whether there is inter-DB2 R/W interest on a particular resource, and a list of the update locks that are currently held. When considering false contention, it is the size of the lock table that you must be concerned with. It is the total size of the lock structure that 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 one-half the total size of the lock structure. The number of members in the group determines the size of each entry in that lock table. IRLM uses the value you specify in the LOCK ENTRY SIZE field of installation panel DSNTIPJ to determine the initial size of the lock table entries. See Section 2 of *DB2 Installation Guide* for details.

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, it is possible to have “false” lock contention. This 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 there is no real lock contention on the resource.

False contention can be a problem for work loads that have heavy inter-DB2 R/W interest.

Monitoring for false contention: You can determine the amount of false contention by using the RMF Coupling Activity reports as described in “Using the coupling facility structure activity report of RMF” on page 215. DB2 also provides necessary information in its accounting and statistics trace classes. See “Using the DB2 statistics trace” on page 216 and “Using the DB2 accounting trace” on page 217 for more information. More detailed information can be found in the performance trace, as shown described in “Using the DB2 performance trace” on page 218.

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. Aim for total global lock contention of less than 5 percent, preferably less than 1 percent. In the descriptions of the various reports, we show you how to figure out the contention percentages.

How to reduce false contention: Some things to do when false contention becomes a problem:

- 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 as described in "Changing the size of the lock structure by rebuilding" on page 219.
- Make sure 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 2 allows twice as many lock entries as compared to a lock entry size of 4, as shown in Figure 47.

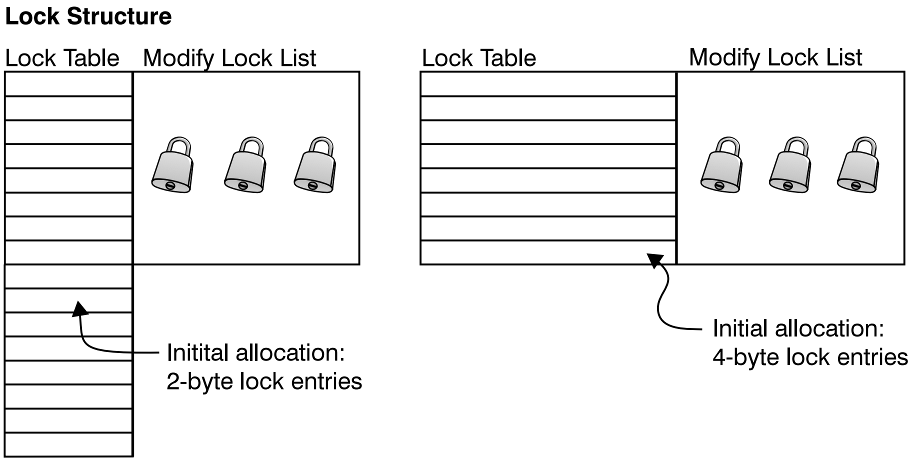


Figure 47. Initial lock entry size

IRLM automatically rebuilds the structure when it needs to. For example, when the 7th member joins the group, the lock structure automatically rebuilds to create the 4-byte lock entries. This prepares the lock structure to handle an 8th member joining the group.

For this reason, even if you anticipate your group growing beyond 7 members, you can start out with a lock entry size of 2 to make the most efficient use of lock structure storage and to reduce false contention.

How to decrease lock entry size: IRLM does not automatically rebuild if the number of members goes down such that a smaller lock entry size is optimal. To decrease the lock entry size, you must:

1. Quiesce all members of the group, using the following command:

   ```
   -DB1G STOP DB2 MODE(QUIESCE)
   ```

2. If IRLM is not automatically stopped along with DB2, enter the MVS command:

   ```
   STOP irlmproc
   ```

3. Force off all connections from the lock structure by issuing the following MVS command:

   ```
   SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL
   ```

4. Force the deallocation of the lock structure by issuing the following MVS command:

   ```
   SETXCF FORCE,STRUCTURE,STRNAME=strname
   ```

5. Change the lock entry size for at least one IRLM. (We recommend changing the value for all of them.)

   If you change the IRLM startup procedure directly, the parameter you change is called MAXUSRS. See the description of MAXUSRS in the description of the command START IRLMPROC in *DB2 Command Reference* ; 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 field of DSNTIPJ.

6. Start the DB2 and IRLM that has the updated value. (You must be sure to 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, this is a disruptive procedure. Consider decreasing the lock entry size only in situations when it 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.

Reduce the time needed to resolve contentions

When there is contention on a hash class, MVS 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, make sure there is no queuing of messages for XCF message buffers. You can use the XCF Activity Report of RMF to detect this queuing. See *OS/390 MVS Setting Up a Sysplex* for more information about tuning the XCF message buffers.

Using the LOCKPART option for partitioned table spaces

For those special cases in which you are purposefully creating affinity between data partitions and DB2 members, you can use the LOCKPART option of CREATE and ALTER TABLESPACE to indicate that you want individual partitions locked only as they are accessed.

With LOCKPART NO, the default, DB2 uses the table space L-lock as the lock parent in the locking hierarchy (the table space lock is represented by a single lock on the last partition of the table space). With LOCKPART YES, each locked partition is a separate lock parent; therefore, DB2 and IRLM can detect when no inter-DB2 R/W interest exists on that partition and thus does not propagate child L-locks unnecessarily.

With LOCKPART YES, you have the option of using the LOCK TABLE with the PART option to exclusively lock individual partitions. See Section 5 (Volume 2) of *DB2 Administration Guide* for more information about the LOCK TABLE statement.

Restrictions: If any of the following conditions are true, DB2 cannot use selective partition locking; instead, it locks *all* partitions:

- The plan is bound with ACQUIRE(ALLOCATE)

- The table space is defined with LOCKSIZE TABLESPACE
- When LOCK TABLE IN EXCLUSIVE MODE is used (without the PART option)

Figure 48 shows partition locks when LOCKPART YES. A partition lock is taken only when the partition is accessed.

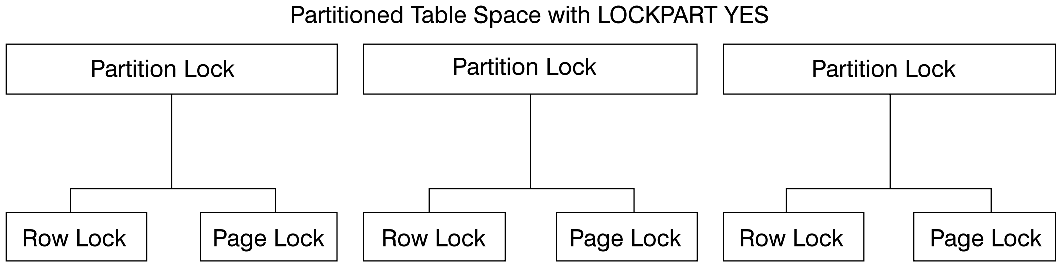


Figure 48. Partition Locks

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 points if using CURSOR WITH HOLD.

The mode of a partition lock: Partition locks have the same possible states as table space locks (IS, IX, S, U, 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, U, 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 DISPLAY DATABASE give information about selective partition locking.

Use performance trace class 6 (IFCID 0020) to see whether you have taken advantage of selective partition locking for your partitioned table spaces. If you are using DB2 PM, you can see this in the locking summary portion of the SQL Activity Report.

Tuning deadlock and timeout processing

This section describes how deadlock detection and resource timeouts work in a data sharing environment. This information can help you choose the appropriate times for deadlock detection intervals and for determining a resource timeout value.

In this section:

- “Global deadlock processing” on page 211
- “Global timeout processing” on page 212

- “Recommendations” on page 213

Global deadlock processing

In a data sharing environment, deadlocks can occur between transactions on different DB2 members. The term *global deadlock* refers to the situation where two or more DB2 members are involved in the deadlock. *Local deadlock* refers to the situation where all of the deadlocked transactions reside on a single DB2 member.

Controlling deadlock detection: Use the DEADLOK parameter in the IRLM startup procedure to control how often IRLM does its deadlock detection processing. Specify the parameter as follows:

```
DEADLOK='x,y'
```

where

x
The number of seconds between two successive scans for a local deadlock (DEADLOCK TIME value on installation panel DSNTIPJ). The default is 5 and can be no larger than 5.

y
The number of local scans that occur before a scan for global deadlock starts (DEADLOCK CYCLE value on install panel DSNTIPJ). The value that is always used by IRLM is 1.

Global deadlock detection requires the participation of all IRLM members in the data sharing group. Each IRLM member has detailed knowledge of the locks that are held and being waited for by the transactions on its associated DB2 member. However, to provide global detection and timeout services, each IRLM is told about all requests that are waiting globally so that the IRLM can provide information about its own blockers. That IRLM also provides information on 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. The global deadlock manager is the IRLM member with the lowest member ID value (as specified in its IRLM procedure). 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: There are four XCF messages 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 gathers that information from all local deadlock detectors and then 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 do.)

3. Each local deadlock detector looks at 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, it chooses a victim 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 victims belonging to that IRLM. When DB2 receives this information, it makes the decision whether to request IRLM to reject any given timeout candidate waiter.

These four messages represent one global detection cycle, and it usually takes 2-4 *x*-second intervals to complete (where *x* is local cycles). Figure 49 illustrates an example where the deadlock time value is set to 5 seconds.

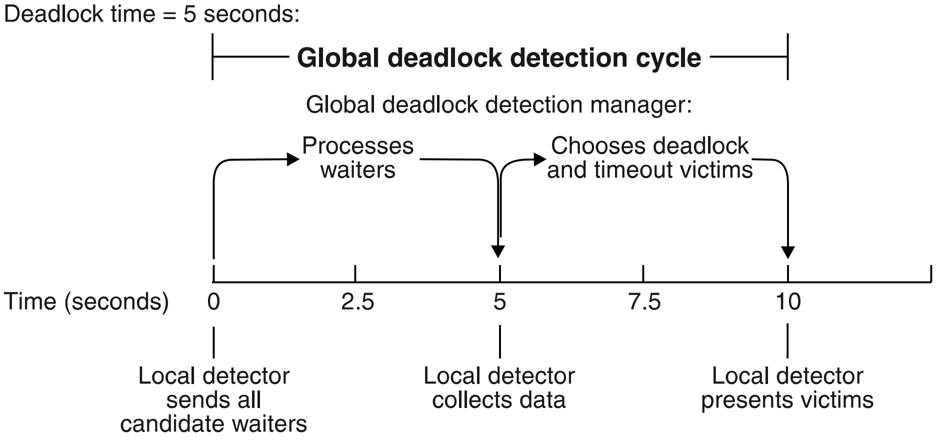

Deadlock time = 5 seconds:
Global deadlock detection cycle
Global deadlock detection manager:
Processes waiters
Chooses deadlock and timeout victims
Time (seconds)
0
2.5
5
7.5
10
Local detector sends all candidate waiters
Local detector collects data
Local detector presents victims

Figure 49. Global deadlock detection cycle

Deadlock detection might be delayed if any of the IRLMs in the group encounter any of the following conditions:

- XCF signalling delays
- IRLM latch contention (could 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 up to the next largest integer
3. Multiply that integer by DEADLOCK TIME

As described in Section 5 (Volume 2) of *DB2 Administration Guide*, 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, then the transaction waits between 60 and 65 seconds before timing out, with the average wait time being about 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 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 there is a heavy IRLM workload.

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 when compared to what you might normally experience 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.

Recommendations

Quick detection of deadlocks and timeouts is necessary in a data sharing environment to prevent a large number of waiters on each system. These large numbers of waiters can cause much longer wait times for timeouts and deadlocks. Based on this assumption, here are two recommendations:

- If your DB2 non-data-sharing subsystem has a problem with deadlocks, consider reducing the deadlock time to prevent these long lists of waiters from developing. (If you don't have a problem with deadlocks, it is likely that you won't 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 your timeout period for your non-data-sharing DB2 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 5.) In a data sharing environment, reduce the value of DEADLOCK TIME so that the timeout period is 40 seconds. This makes it more likely that your actual wait time for timeouts is at or before 60 seconds.

Monitoring DB2 locking

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 your applications are doing, but you must also make sure that any locking problems are not caused by data sharing resources, such as an undersized lock structure, or the over-utilization of the coupling facility or coupling facility channels (links).

This section describes the following ways to watch locking activity and lock structure activity:

“Using the command DISPLAY DATABASE”
“Using the coupling facility structure activity report of RMF” on page 215
“Using the DB2 statistics trace” on page 216
“Using the DB2 accounting trace” on page 217
“Using the DB2 performance trace” on page 218

There's also the MVS command D XCF,STRNAME, described in “Displaying information about specific structures” on page 140.

Using the command DISPLAY DATABASE

General-use Programming Interface

Use the LOCKS ONLY option on DISPLAY DATABASE to display information about page set, partition, or table locks held on resources. The “lock” column of the display describes the type and duration of locks used by corresponding agents.

If the table space is defined with LOCKPART NO, the display looks like Figure 50. The LOCKINFO field shows a value of S, indicating that this is a table space lock.

```
NAME     TYPE PART STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ---- ----------------- -------- ------------ ---------
TSPART   TS                          LSS001   DSN2SQL      H-IS,S,C
    -                MEMBER NAME DB1G
TSPART   TS                          LSS002   DSN2SQL      H-IS,S,C
    -                MEMBER NAME DB2G
TSPART   TS    01  RO                                      H-S,PP,I
    -                MEMBER NAME DB1G
TSPART   TS    02  RW                                      H-S,PP,I
    -                MEMBER NAME DB2G
TSPART   TS    03  RW                                      H-S,PP,I
    -                MEMBER NAME DB2G
```

Figure 50. Example DISPLAY DATABASE LOCKS for a table space defined with LOCKPART NO

Figure 51 on page 215 is an example of output of DISPLAY DATABASE when the table space is defined with LOCKPART YES. The application identified as LSS001 on member DB1G has locked partitions 1 and 2. LSS002 on member DB2G 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 DB1G
TSPART   TS    01  RO                                      H-S,PP,I
     -                 MEMBER NAME DB1G
TSPART   TS    01  RO                LSS002   DSN2SQL      H-IS,P,C
     -                 MEMBER NAME DB2G
TSPART   TS    01  RO                                      H-S,PP,I
     -                 MEMBER NAME DB2G
TSPART   TS    02  RW                LSS001   DSN2SQL      H-IS,P,C
     -                 MEMBER NAME DB1G
TSPART   TS    02  RW                                      H-S,PP,I
     -                 MEMBER NAME DB1G
TSPART   TS    03  RW                LSS002   DSN2SQL      H-IS,P,C
     -                 MEMBER NAME DB2G
TSPART   TS    03  RW                                      H-S,PP,I
     -                 MEMBER NAME DB2G
```

Figure 51. Example DISPLAY DATABASE LOCKS for a table space defined with LOCKPART YES

End of General-use Programming Interface

Using the coupling facility structure activity report of RMF

The Coupling Facility Activity Report of RMF describes activity to all structures in the coupling facility for a given time period. Figure 52 shows a partial report, giving information about:

- Total number of lock-related requests (A)
- Number of requests that were deferred because of contention (B).
- The number of deferred requests that were caused by false contention (C).

```
                                     COUPLING  FACILITY  STRUCTURE  ACTIVITY
STRUCTURE NAME = DSNDB0G_LOCK1     TYPE = LOCK
          # REQ    -------------- REQUESTS -------------    -------------- DELAYED REQUESTS -------------
SYSTEM    TOTAL            #    % OF  -SERV TIME(MIC)-    REASON    #    % OF  ---- AVG TIME(MIC) -----  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 A 162K
                                                                                                                     B
          701.7    ASYNC     0    0.0%     0.0      0.0    NO SCH    0   0.0%    0.0       0.0    0.0    REQ DEFERRED 621
                   CHNGD     0    0.0%  INCLUDED IN ASYNC                                                -CONT        621
                                                                                                                     C
                                                                                                         -FALSE CONT  212
```

Figure 52. Partial RMF Coupling Facility Activity Report for lock structure

Determining the contention percentages: 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 our example:

  ```
  (621  / 162000) × 100 = .387%
  ```

  This indicates that the global contention rate is approximately 0.39 percent (a good figure).

- False contention is the number of false contentions (C) divided by the total number of requests (A) multiplies by 100. For our example, then:

  ```
  (212 / 162000) × 100 = 0.13%
  ```

  Thus, the rate of false contention is 0.13 percent, 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 DB2 in the data sharing group is encountering. This trace runs with low overhead. It is therefore a good idea to keep this trace turned on to allow continuous monitoring of each subsystem.

See Figure 53 to see what kind of information you can get from a statistics trace.

```
DATA SHARING LOCKING        QUANTITY  /MINUTE  /THREAD  /COMMIT
---------------------------  --------  -------  -------  -------
GLOBAL CONTENTION RATE (%)       0.41 F
FALSE CONTENTION RATE (%)        0.17  G

LOCK REQUESTS (P-LOCKS)      10515.00  1050.53  2103.00     0.72
UNLOCK REQUESTS (P-LOCKS)     9183.00   917.45  1836.60     0.63
CHANGE REQUESTS (P-LOCKS)      101.00    10.09    20.20     0.01
A
SYNCH.XES - LOCK REQUESTS      484.5K    48.4K    96.9K    33.40
SYNCH.XES - CHANGE REQUESTS  50557.00  5051.02    10.1K     3.49
SYNCH.XES - UNLOCK REQUESTS    242.5K    24.2K    48.5K    16.70
ASYNCH.XES - RESOURCES   B       2.00     0.20     0.40     0.00

SUSPENDS - IRLM GLOBAL CONT C 1829.00   182.73   365.80     0.13
SUSPENDS - XES GLOBAL CONT. D    4.00     0.40     0.80     0.00
SUSPENDS - FALSE CONTENTION E 1342.00   134.08   268.40     0.09
INCOMPATIBLE RETAINED LOCK       0.00     0.00     0.00     0.00

NOTIFY MESSAGES SENT            30.00     3.00     6.00     0.00
NOTIFY MESSAGES RECEIVED       867.00    86.62   173.40     0.06
P-LOCK/NOTIFY EXITS ENGINES     10.00      N/A      N/A      N/A
P-LCK/NFY EX.ENGINE UNAVAIL      0.00     0.00     0.00     0.00

PSET/PART P-LCK NEGOTIATION      2.00     0.20     0.40     0.00
PAGE P-LOCK NEGOTIATION        553.00    55.25   110.60     0.04
OTHER P-LOCK NEGOTIATION       104.00    10.39    20.80     0.01
P-LOCK CHANGE DURING NEG.      655.00    65.44   131.00     0.05
```

Figure 53. Data sharing locking block of DB2 PM Statistics Report

Explanation of fields

A These counters indicate the total numbers of lock-related requests (both L-locks and P-locks) that are propagated to XES synchronously.

B This number indicates the number of lock-related requests that were propagated to XES asynchronously. DB2 uses the term *asynchronous* 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 DB2 has an IS lock on a particular table space and another DB2 requests an IX lock. The S child locks held by the first DB2 must be propagated under a system execution unit to XES and the coupling facility. See Figure 46 on page 205 for an example of this.

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.

C The number of real contentions, as detected by IRLM.

D 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.

E The number of false contentions.

Calculating global contention percentages: To figure the global contention percentages, using the statistic report shown in Figure 53 on page 216. Our calculations only account for synchronous lock requests.

- Total contention is:

  The total number of suspends because of contention (C + D + E)

  divided by
  The total number of requests that went to XES (excluding asynchronous requests): ((three fields under A) + C + D + E)

  Multiplied by 100.

  So, for our example:

  ```
  (3175 / 780732) × 100 = .41%
  ```

  This indicates that the contention rate is approximately .41 percent (F). Because this is such a low rate of contention, it is probably not necessary to determine the amount of false contention. However, here is the calculation you would use:

- False contention is the number of false contentions (E) divided by the total number of requests that went to XES (excluding asynchronous requests) ((three fields under A) + (C + D + E)) multiplied by 100. For our example, then:

  ```
  1342 / 780732 × 100 = .17%  ( G )
  ```

  Thus, the approximate rate of false contention is less than .2 percent, a very low figure.

Using the DB2 accounting trace

Use the accounting trace to determine which users or plans are experiencing global lock contention. The DB2 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 Figure 54 on page 218. 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(%)       0.43       N/A
DEADLOCKS           0.00         0    FALSE CONT RATE(%)    M   0.20       N/A
ESCAL.(SHARED)      0.00         0    LOCK REQ   - PLOCKS   C   2.11      2434
ESCAL.(EXCLUS)      0.00         0    UNLOCK REQ - PLOCKS   K   1.53      1769
MAX LOCKS HELD     20.43        28    CHANGE REQ - PLOCKS   D   0.03        36
LOCK REQUEST   A   42.12     48570    LOCK REQ   - XES     E   41.82     48220
UNLOCK REQUEST J   24.59     28348    UNLOCK REQ - XES     L   24.70     28481
QUERY REQUEST       0.00         0    CHANGE REQ - XES      F  11.10     12796
CHANGE REQUEST  B  11.10     12803    SUSPENDS   - IRLM     G   0.18       205
OTHER REQUEST       0.00         0    SUSPENDS   - XES      H   0.00         0
LOCK SUSPENS.       0.02        19    SUSPENDS   - FALSE    I   0.16       179
LATCH SUSPENS.      1.17      1344    INCOMPATIBLE LOCKS        0.00         0
OTHER SUSPENS.      0.00         0    NOTIFY MSGS SENT          0.00         3
TOTAL SUSPENS.      1.18      1363
```

Figure 54. Portion of DB2 PM accounting report showing locking activity

Checking for false contention: The false contention rate (M) is a percentage of the total number of requests that went to XES (E + L + F).

In our example:

```
(179 / 89497) × 100 = .2%    M
```

False contention is approximately .2 percent of the total number of requests that went to XES.

Checking for transaction locking optimizations: To see how well global transaction locking optimizations are working in your application, you must:

1. Determine the total number of L-lock LOCK requests propagated to XES (call that X). This requires subtracting P-lock LOCK requests:

   ```
   E - C = X
   ```

   In our example:

   ```
   48220 - 2434 = 45786
   ```

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 our example:

   ```
   45786 / 48570 = 94%
   ```

In this particular work load of 100 percent R/W sharing, 6 percent of LOCK requests were not propagated to XES and the coupling facility because of locking optimizations.

Using the DB2 performance trace

The DB2 performance trace gives more detail 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.

See *DB2 PM for OS/390 Report Reference Volume 2* for more information about the data shown in IFCID 0045.

Changing the size of the lock structure

This section describes two possible ways of changing the size of the lock structure. One method is dynamic and changes the storage of only the modify lock list portion of the lock structure. The other method requires a CFRM policy change and a rebuild, but it can change the storage of both the modify lock list and lock table portions of the lock structure.

Changing the lock structure size dynamically

If *all* of the following conditions are true:

- 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.
- You do not want to change the size of the lock table portion of the lock structure.

Then you can enter the following command (this example assumes the group name is DSNDB0G):

```
SETXCF START,ALTER,STRNAME=DSNDB0G_LOCK1,SIZE=newsize
```

This example assumes that *newsize* is less than or equal to the maximum size defined the CFRM policy for the lock structure.

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.

Because the dynamic method does affect only the modify lock list portion of the lock structure, the impact of changing the lock structure size can have a disproportionately large effect on that modify lock list portion of the structure. For example, if you halve the size of the lock structure, it can result in all of the available lock list entries being taken away—probably not the result you want.

Changing the size of the lock structure by rebuilding

If *any* of the following conditions are true:

- 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.
- You want to change the size of the lock table portion of the lock structure.

Then you must do the following procedure:

1. Increase the storage for the lock structure in the CFRM policy to the next power of 2 to allow the lock table portion of the structure to be increased. For example, if the lock structure was 16MB, increase the size to 32MB.
2. Use the MVS command SETXCF START,POLICY to start the updated policy.
3. Issue the following MVS command to rebuild the structure (this example assumes the group name is DSNDB0G):

   ```
   SETXCF START,REBUILD,STRNAME=DSNDB0G_LOCK1
   ```

You can use a performance class 20 trace (IFCIDs 0267 and 0268) to monitor how long the rebuild of the lock structure takes.

Tuning group buffer pools

This chapter describes the following about how DB2:

- Ensures that DB2 does not read down-level data that is cached in its member buffer pools (*cache coherency*).
- As much as possible, enables a quick refresh of a down-level page without having to go to DASD.

With data sharing, group buffer pools are a key component of cache coherency as are the subsystem locking mechanisms, the *P-locks*, used in that process. Your understanding of these processes is helpful when tuning the data sharing group for best performance.

With data sharing, a database page can reside:

In a virtual buffer pool

In a hiperpool

In a group buffer pool

On DASD

Database pages continue to be cached in each sharing member's buffer pools before they can be referenced or updated. Each sharing member can control its own buffer pool configurations (that is, the size and number of buffer pools). However, if there is inter-DB2 R/W interest in the data, then 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).

It is also possible to cache clean pages in the group buffer pool as a mutually exclusive alternative to hiperpools.

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 DB2 subsystems 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 DASD.

This section describes the following topics:

Assigning page sets to group buffer pools

Any data sharing group can have up to 50 4KB-page size group buffer pools and up to ten each of 8KB, 16KB, and 32KB-page size group buffer pools. Different group buffer pools can be on different coupling facilities. Because of a strict naming convention you must use, DB2 maps the group buffer pools to the buffer pools. For example, buffer pool BP0 maps to group buffer pool GBP0. Thus, it is your choice of buffer pool that 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.

General-use Programming Interface

For example, to assign table space DSN8S61D to GBP2, you must:

1. Stop all access to the table space by issuing the following command:

```
-DB1G STOP DATABASE(DSN8D61A) SPACENAM(DSN8S61D)
```

2. Change the buffer pool assignment by executing the following SQL statement:

```
ALTER TABLESPACE DSN8D61A.DSN8S61D
 BUFFERPOOL BP2;
```

3. Allow access to the table space by issuing the following command:

```
-DB1G START DATABASE(DSN8D61A) SPACENAM(DSN8S61D)
```

End of General-use Programming Interface

The above procedure works only when you are altering a table space to a buffer pool with the same page size. For information about changing an existing page size, see Section 2 (Volume 1) of *DB2 Administration Guide*.

Recommendations for performance

- For best performance, it is best to keep GBP-dependent page sets in separate buffer pools from non-GBP-dependent page sets. For example, it is a good idea to keep work file table spaces, which are always non-GBP-dependent, in different buffer pools than that 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 the registration and unregistration of pages to the group buffer pool.

- In prior releases of DB2, data rows that are too large for a single 4–KB must use a 32–KB page. Limited choices of page sizes can result in higher I/O and coupling facility costs, especially for random processing. For example, you could have a table with a row size of greater than 4 KB that is usually randomly accessed. However, a row size of 32 KB would be too large, because the larger row size increases I/O activity and coupling facility overhead. New 8–KB and 16–KB pages give you choices that are just right to improve the balance for different processing requirements. Now you can store larger rows more efficiently and improve performance. This enhancement is beneficial for both non-data sharing and data sharing environments.

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, there are operational issues to consider:

- You cannot do workload balancing for that data, because the other DB2s in the group are not aware of that data. Thus, it is possible for the member that has access to the data to 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.

Defining private data: If you want access to 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 DB2 in the group. This prevents the other members of the group from attempting to use this buffer pool and therefore from accessing table space NOSHARE.

Inter-DB2 interest and GBP-dependency

The concepts of *inter-DB2 R/W interest* and *group buffer pool dependency* (GBP-dependency) are closely related. Whenever there is inter-DB2 R/W interest on a page set or partition, that object is GBP-dependent. Conversely, if there is no inter-DB2 R/W interest, 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 R/W interest has gone away.

Table 34. Determining Group Buffer Pool Dependency

Your DB2's Interest	Other DB2's Interest	Is Page Set GBP-Dependent?
R/O	None, R/O	No
R/O	R/W	Yes
R/W	None	No
Exception: 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.		
R/W	R/O	Yes
R/W	R/W	Yes

How DB2 Tracks Interest

The mechanism DB2 uses to express interest in an object is a global lock called a *physical lock* (abbreviated to *P-lock*). These locks are “physical” to contrast them with transaction locks, sometimes called “logical locks” (or L-locks, for short). 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, they play an important part in how DB2 tracks inter-DB2 interest. Page set and page P-locks are described in more detail in “P-locking” on page 227.

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. Table 35 shows when those operations occur on each member.

Table 35. 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.
There was no update within an installation-specified time period or number of checkpoints (R/O 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 Table 35, there is also a special case that occurs in the following conditions:

- There is a single DB2 with R/W interest and any number of DB2s with R/O interest.
- All members have R/O interest and the page set or partition has been GBP-dependent since the time it was physically opened.

In those conditions, if the R/O DB2s do not reference the page set again in a certain amount of time, DB2 physically closes the page set for the R/O DB2s to remove the inter-DB2 R/W interest. See Figure 56 on page 224 for an example of this.

Scenarios of P-Lock operations

Figure 55 on page 224 shows a typical sequence of events for P-locking and P-lock negotiations between two members of a data sharing group.

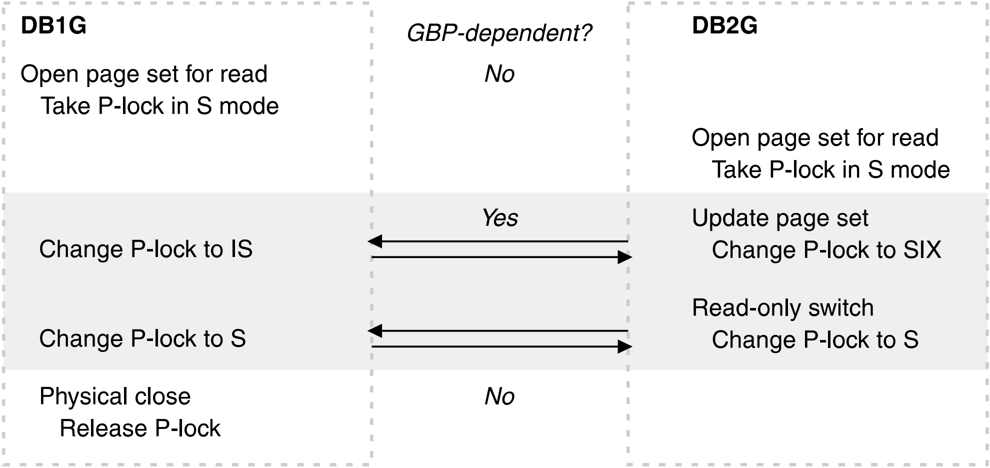


Figure 55. P-lock operations between two members. The arrows indicate that the members are negotiating P-lock modes.

Figure 56 shows what happens when a single updater remains.

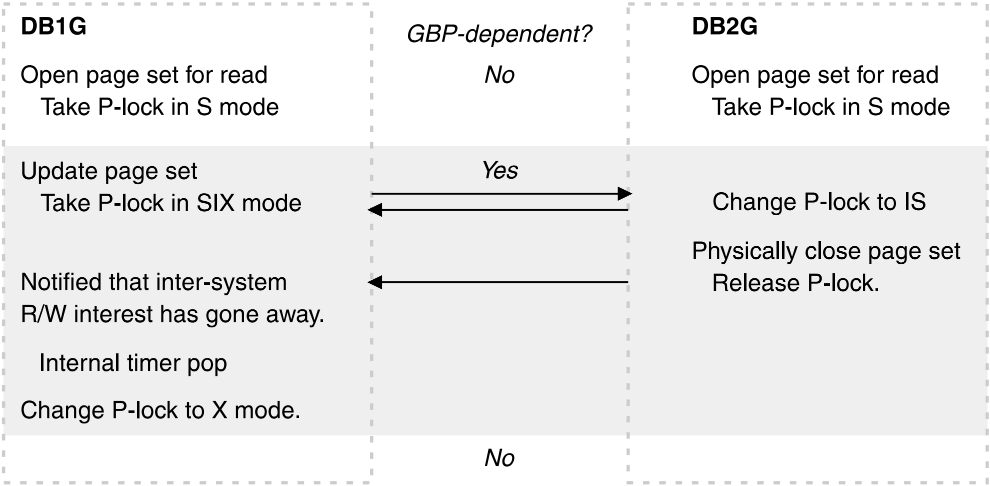


Figure 56. P-lock operations when single updater remains. When the reader physically closes the page set, the updater does not remove the GBP-dependency immediately.

Tuning recommendation

To avoid having DB2 go in and out of GBP-dependency too often, tune the subsystem parameters that affect when data sets are switched to a different state. These parameters are found on installation panel DSNTIPN:

- CHECKPOINT FREQ
- RO SWITCH CHKPTS
- RO SWITCH TIME

See Section 5 (Volume 2) of *DB2 Administration Guide* for more information about these parameters.

Determining the amount of inter-system sharing

To determine the amount of inter-system sharing, you can use the DIS BPOOL command with LIST option to get a snapshot of your objects and how they are being shared across members. Using the DB2 command, you will see, by partition, which members have interest in the object, which level of interest they have (the P-LOCK state) and who the castout owner is. You can also use the DBNAME, SPACENAME, GBPDEP, and CASTOWNR filters to limit the scope of the report.

Issue the command:

```
-DIS BPOOL(BP0) LIST(*)
```

Your output will be similar to the following example:

```
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
DSNB403I : HIPERPOOL SIZE = 1000 BUFFERS, CASTOUT = YES
             ALLOCATED       =        0   TO BE DELETED   =        0
             BACKED BY ES    =        0
DSNB404I : THRESHOLDS -
            VP SEQUENTIAL    = 80   HP SEQUENTIAL          = 75
            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                 0   IS
                                 V61B        Y        0   SIX
DSN8D61A DSN8S61E     001 TS  Y  V61A                 0   IS
                                 V61B        Y        0   SIX
                      002 TS  N  V61A                 0   S
                                 V61B                 0   S
DSNDB01  DSNSPT01         IX  N  V61A                 0   S
                                 V61B                 0   S
         SPT01            TS  N  V61A                 0   S
                                 V61B                 0   S
DSNDB07  DSN4K01          TS  N  V61A                 0   S
...
DSN9022I : DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION
```

Figure 57. Sample DISPLAY DATABASE output showing the amount of inter-system sharing

Displaying GBP-dependent page sets

General-use Programming Interface

To find out if a particular page set is GBP-dependent, use the DISPLAY DATABASE command with the LOCKS option:

```
-DB1G  DISPLAY DB(DSN8D61A) SPACE(DSN8S61D) LOCKS
```

Your output will be similar to the following example:

```
NAME     TYPE PART STATUS             CONNID   CORRID       LOCKINFO
-------- ---- ---- ------------------ -------- ------------ ----------
DSN8S61D    TS        RW                                       H-IX,PP,I
                                      MEMBER NAME DB1G (CO)
DSN8S61D    TS        RW                                       H-IX,PP,I
                                      MEMBER NAME DB2G
```

Figure 58. 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, then the identified page set is GBP-dependent. Thus, the above output shows that DSN8S61D is GBP-dependent.

End of General-use Programming Interface

Determining GBP-dependency for a particular member: There might be a time when you need to know what the impact is to bring down a particular member or disconnect a particular member from the group buffer pool. You can use the DISPLAY BUFFERPOOL command with the GBPDEP(Y) option to discover whether this DB2 member has any page sets opened that are GBP-dependent:

```
-DB1G DISPLAY BUFFERPOOL(BP0) GBPDEP(Y)
```

Your output will be similar to the following example:

```
@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
DSNB403I @ HIPERPOOL SIZE = 1000 BUFFERS, CASTOUT = YES
             ALLOCATED       =        0   TO BE DELETED   =        0
             BACKED BY ES    =        0
DSNB404I @ THRESHOLDS -
            VP SEQUENTIAL    = 80   HP SEQUENTIAL          = 75
            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                 0   IX
DSN8D71A DSN8S71E      001 TS  Y  V61A        Y        0   IX
                                 V61B                 0   IX
                       003 TS  Y  V61A        Y        0   IX
                                 V61B                 0   IX
DSNDB01  DSNLLX01         IX  Y  V61A        Y        0   IX
                                 V61B                 0   IX
         SYSLGRNX         TS  Y  V61A        Y        0   IX
                                 V61B                 0   IX
DSN9022I @ DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION
```

Figure 59. Sample DISPLAY BUFFERPOOL output indicating page sets and partitions are group-buffer-pool-dependent

P-locking

This section describes P-locks on page sets and on pages, and it describes how to monitor and tune those locks.

Page Set P-Locks

P-locks are used on physical objects stored in buffers (table spaces, index spaces, and partitions). They have complete partition independence; that is, 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 there is a P-lock 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 important ways:

- P-locks are owned by a subsystem. 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 DB2 subsystem requests a P-lock that another DB2 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 GBP, and then downgrading the P-lock to a mode that is compatible with the new request.

Displaying retained P-locks

Just as with transaction locks, certain P-locks can be retained because of a system failure. A retained P-lock means other DB2s cannot access the data that the P-lock is protecting if the accessing DB2 requests a P-lock in an incompatible state. Thus, if a DB2 fails holding an IX page set P-lock, it is still possible for another DB2 to obtain an IX page set P-lock on the data. See “Active and retained locks” on page 169 for more information about retained locks and when they are released.

Use the DISPLAY DATABASE command with the LOCKS option to determine if there are retained locks on a table space, index, or partition. An “R” in the LOCKINFO column indicates that a lock is retained.

Table 36 shows the possible modes of access for a page set and the P-lock state that is retained should DB2 fail.

Table 36. Determining retained P-lock state

Your DB2's Interest	Other DB2s' Interest	Retained P-Lock States of Your DB2
R/O	None, R/O	None
R/O	R/W	None
R/W	None	X or NSU
Note: 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.		
R/W	R/O	IX
Note: 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.		
R/W	R/W	IX

Page P-locks

There are times when 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. They 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 DB2 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 DB2 member is updating the index page set or partition
- There are no repeatable read claimers on the read-only DB2 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 carefully before using row locking 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 1 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.

Monitoring P-locks

There is more overhead when there is inter-DB2 R/W interest in a page set than when there is not. Although DB2 does dynamically track inter-DB2 R/W interest, which helps avoid data sharing overhead when it is not needed, you will pay some cost for the advantages of data sharing.

Monitoring P-lock activity, especially page P-locks, can help you determine if it is necessary to take steps to control inter-DB2 R/W interest. If there is excessive global contention that cannot be resolved by any tuning measures, it might be necessary 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 out information about page set P-locks, including what DB2 subsystem is holding or waiting for P-locks, and whether there are P-locks being held because of a DB2 failure. Figure 58 on page 226 has a sample of output obtained from the command. A “PP” in the LOCKINFO field of the output indicates that a particular lock is a page set or partition P-lock.

Information about P-locks can be obtained by the statistics and accounting traces, along with information about transaction locking. 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 turn on a full DB2 lock trace. IFCID 0259 is mapped by DSNDQW04.

Reducing space map page contention

This section describes a couple of options you can use when defining table spaces to help reduce the space map hot spots that can occur when a lot 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.

Neither option is one to choose lightly, so understand the implications of each option before choosing either one.

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 can't 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 DASD.
- 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, so if you have a query application that performs best when data is in clustering order, run REORG on the table space before starting the query application.

Avoid tracking updates: 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, there is less coupling facility overhead, but the cost of incremental image copies is much higher because DB2 must use a table space scan to read all pages to find out 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 DASD 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

This section describes how the process of reading data is changed for data sharing.

Where DB2 looks for a page

DB2 searches in this order:

1. In the virtual buffer pool. If the page is invalid, it refreshes the page from the group buffer pool (or DASD).
2. In the hiperpool. If the page is invalid, it refreshes the page from the group buffer pool (or DASD). (This step is skipped for GBPCACHE ALL page sets.)
3. 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 group buffer pool 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
4. If the page is not in the group buffer pool, DB2 refreshes the page in the virtual buffer pool from DASD.

Note that 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 DASD because it might no longer be the most current version of the data. This 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. Table 37 indicates when this test is performed.

For duplexed group buffer pools, only the primary structure is used for cross-invalidations.

Table 37. Determining when page validity must be tested

Your DB2's Interest	Other DB2s' Interest	Test Page Validity?
R/O	None, R/O	No
R/O	R/W	Yes
R/W	None	No
R/W	R/O	No
R/W	R/W	Yes

Prefetch processing

DB2's prefetch processing for GBP-dependent page sets and partitions varies depending on what release of MVS you are running and what level (CFLEVEL) of coupling facility the group buffer pool is allocated in.

If the group buffer pool is allocated in a coupling facility with CFLEVEL=0 or 1, then DB2 reads and registers one page at a time in the group buffer pool.

When 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.

DB2 does not include on the list any valid pages that are found in the local virtual buffer pool or hiperpool.

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, there most likely won't be any changed pages 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 DASD read I/O request, depending on which is most efficient.

Determining 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 **J** in Figure 70 on page 253). You can also use the DISPLAY GROUPBUFFERPOOL command with the MDETAIL option (see message DSNB789I).

Caching pages that are read in from DASD

You can cache pages in the group buffer pool as they are read in 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 DASD, even if there is no inter-DB2 R/W interest in those pages.

However, when there is only a single DB2 that has exclusive R/W interest in the page set (that is, only one DB2 has the page set open for update) , pages are not cached in the group buffer pool when they are read in. They can, however, be cached in the hiperpool, if one exists.

Choosing GBPCACHE ALL does not prevent DB2 from continuing to cache changed pages in the group buffer pool before writing them to DASD (the function provided by the default GBPCACHE CHANGED).

Example

General-use Programming Interface

Here is an example of using the GBPCACHE clause to cache read-only page sets in the group buffer pool:

```
ALTER TABLESPACE DSN8D61A.DSN8S61D
  GBPCACHE ALL;
```

See Chapter 6 of *DB2 SQL Reference* for more information about the GBPCACHE clause.

End of General-use Programming Interface

Why choose GBPCACHE ALL?: GBPCACHE ALL avoids having several different members read the same page in from DASD. It is most useful for workloads where there is primarily inter-DB2 read interest. To prevent double buffering for clean pages, hiperpools are not used for page sets or partitions that are defined with the GBPCACHE ALL option unless the DB2 is an exclusive updater (no other DB2s have the page set open). In a non-data-sharing environment, hiperpools are still used regardless of the GBPCACHE option.

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.

Recommendation: For LOB table spaces, use GBPCACHE SYSTEM, which is the default for LOB table spaces. If you use GBPCACHE ALL or 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.

Write operations

With data sharing, DB2 usually writes changed data to the group buffer pool before writing that changed data to DASD.

This section includes the following topics:

“The GBPCACHE option's effect on writes”
“Writing to the group buffer pool” on page 235
“Writing to DASD from the group buffer pool” on page 236

The GBPCACHE option's effect on writes

There are several options on CREATE TABLESPACE for telling DB2 how you want data to be handled through the group buffer pool. This section describes the various options and in what cases each might be appropriate.

GBPCACHE CHANGED or ALL: Most of the time you will be using GBPCACHE CHANGED. (The discussion of when to use GBPCACHE ALL is in “Caching pages that are read in from DASD” on page 232.) For both ALL and CHANGED, the write operations are the same for changed pages; that is, changed pages are written to the group buffer pool before being written to DASD at some later point. Details about write operations are described in “Writing to the group buffer pool” on page 235.

GBPCACHE SYSTEM: GBPCACHE SYSTEM 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 DASD, similar to GBPCACHE NONE page sets. SYSTEM is the default for LOB table spaces.

Recommendation: 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 DASD 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 defined as GBPCACHE(NO), no pages are written to the group buffer pool. The group buffer pool is used solely for the purpose of buffer cross-invalidation. At every COMMIT, any pages that were updated by the transaction and still have not yet been written are synchronously written to DASD during commit processing. This can have a severe impact on performance for most types of transactions.

One potential advantage of not caching 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 it is recommended to specify an alternate coupling facility in the CFRM policy. 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 DASD 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. It is usually less disruptive to change the group buffer pool attribute than the page set attribute. Because the GBPCACHE(NO) attribute takes precedence over the GBPCACHE option on the page set, you could plan for using 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 could 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 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 command to enable the new attribute.

Is there ever any reason not to cache? It is possible to obtain a performance benefit 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 transfering data to the group buffer pool and casting out from the group buffer pool. Again, this benefit is at the cost of synchronous DASD I/O at COMMIT. To reduce the amount of synchronous DASD I/O at COMMIT, you can lower the defered 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 GRPCACHE(NO), or the page sets using this group buffer pool might be a good candidate for GRPCACHE 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.

Writing 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 R/W interest, DB2 forces the updated pages to the group buffer pool when the transaction commits, or before. 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 is short of 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 doesn't issue frequent commits). In this case, a system checkpoint can clean out the buffer pool before the commit.
- The same page is required for update by another system because there is no conflict on transaction locking (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 pool are invalidated. This means that the next time one of those members needs that page, the page must be refreshed from the group buffer pool (or DASD).

Before an updated page is written to the group buffer pool or DASD, DB2 also ensures that the last update log record for that page is externalized to the active log. This is necessary to ensure that updates can be backed out when necessary.

When committing an updating transaction, pages that were updated but not yet written to the group buffer pool are synchronously 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, use a START DATABASE command with the SPACENAM option to recover the pages on the LPL.

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 nonsystem pages of a page set defined with GBPCACHE SYSTEM

Instead, DB2 writes changed pages for the transaction directly to DASD at or before COMMIT. DB2 batches up to 32 pages in a single I/O.

When DB2 writes a changed page to DASD, DB2 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, the following events occur:

1. For some fixed number of pages that must be written, do the following for each page:
   a. Write the page to the secondary structure asynchronously
   b. Write the page to the primary structure synchronously
2. After all pages have been written to the primary structure, DB2 checks to see if all pages have been written to the secondary structure. If some pages are still to be written, DB2 forces the completion of those writes.

Writing to DASD from the group buffer pool

The process of writing pages from the group buffer pool to DASD is called *castout*. Because there is no physical connection between the group buffer pool and DASD, the castout process involves reading the page from the group buffer pool into a particular DB2's private buffer (not part of the buffer pool storage) and writing the page from the private buffer to DASD. This DB2 is the owner of the castout process for the page set or partition. The DB2 that is assigned ownership of castout is the DB2 subsystem that had the first update intent on the page set or partition. After the castout ownership is assigned, subsequent updating DB2 subsystems 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 DB2s 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 DASD, 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. Castout class thresholds are described in “Group buffer pool class castout threshold” on page 241.
- The total number of changed pages for a group buffer pool exceeds a group buffer pool threshold value, described in “Group buffer pool castout threshold” on page 242.
- The group buffer pool checkpoint is triggered. See “Group buffer pool checkpoint” on page 238 for more information.
- There is no more inter-DB2 R/W interest 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 there is no more shared interest in the page set are purged from the group buffer pool.

Casting out from a duplexed group buffer pool: DB2 casts out data to DASD 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.

Determining the castout owner:

General-use Programming Interface

Use the DISPLAY DATABASE command with the LOCKS option to display the current castout owner for a given page set:

```
-DB1G    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 Figure 60.

```
⋮
TBS43    TS    01  RW                                         H-SIX,PP,I
    -                MEMBER NAME DB2G     (CO)
TBS43    TS    02  RW                BATCH    SELEC        H-IS,P,C
    -                MEMBER NAME DB1G
⋮
```

Figure 60. Partial DISPLAY DATABASE output showing castout owner for a partition

End of General-use Programming Interface

Use the DISPLAY BUFFERPOOL command with the CASTOWNR keyword to see which page sets or partitions the members hold castout ownership for:

```
-DIS BUFFERPOOL(BP0) CASTOWNR(Y)
```

The output will list the members that hold the page set or partition p-lock in "U" state.

```
@DIS BPOOL(BP0) CASTOWNR(Y)
DSNB401I @ BUFFERPOOL NAME BP0, 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
DSNB403I @ HIPERPOOL SIZE = 1000 BUFFERS, CASTOUT = YES
             ALLOCATED       =        0   TO BE DELETED   =        0
             BACKED BY ES    =        0
DSNB404I @ THRESHOLDS -
            VP SEQUENTIAL    = 80   HP SEQUENTIAL          = 75
            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                 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                 0   IX
         SYSLGRNX         TS  Y  V61A        Y        0   IX
                                 V61B                 0   IX
DSN9022I @ DSNB1CMD '-DIS BPOOL' NORMAL COMPLETION
```

Figure 61. 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 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 (the primary one only, 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 DB2 that connects to this group buffer pool, although the ownership can change over time. Message DSNB798I in the DISPLAY GROUPBUFFERPOOL output shows which DB2 is the current structure owner.

Default checkpoint frequency: The default checkpoint frequency is 8 minutes. You can change the default checkpoint frequency by using the ALTER GROUPBUFFERPOOL command, described in "Changing the checkpoint frequency" on page 256. For group buffer pools defined as GBPCACHE(NO), the checkpoint interval is ignored; no checkpointing occurs for those group buffer pools.

Tuning the group buffer pool checkpoint interval: 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 output.

DB2 has two possible ways of gathering checkpoint information:

- By issuing many "read directory info" requests

  This method is used when your 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 *ssnm*DBM1 address space, especially for the structure owner. This is because of the increased number of times it 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, and when APAR OW28460 is applied to MVS on all members of the data sharing group. (Or when the members are running at OS/390 Release 6, or a subsequent release.)

  When you look at the MDETAIL report from DISPLAY GROUPBUFFERPOOL, as shown in Figure 63 on page 240, 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 8 minutes is a good balance between the performance and recovery considerations in most cases.

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 –DIS GBPOOL command.

General-use Programming Interface

The following instrumentation helps you more effectively monitor and tune the group buffer pool checkpoint:

- DISPLAY GROUPBUFFERPOOL shows which member is the structure owner and also shows the group buffer pool checkpoint recovery LRSN:

```
DSNB798I -DB1G LAST GROUP BUFFER POOL CHECKPOINT 17:23:21 MAY  9, 1996
               GBP CHECKPOINT RECOVERY LRSN             = ACD74C01EE30
               STRUCTURE OWNER                          = DB1G
```

Figure 62. Partil DISPLAY GROUPBUFFERPOOL output showing which member owns the structure and the the group buffer pool checkpoint recovery LRSN

- The DISPLAY GROUPBUFFERPOOL MDETAIL contains the number of checkpoints that occurred for this group buffer pool. The statistics trace also includes this information.

  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.

```
DSNB778I -DB1G CASTOUT THRESHOLDS DETECTED
               FOR CLASSES                              = 3
               FOR GROUP BUFFER POOL                    = 1
             GBP CHECKPOINTS TRIGGERED                  = 1
             PARTICIPATION IN REBUILD                   = 0
DSNB796I -DB1G CASTOUTS
               PAGES CASTOUT                            = 18
               UNLOCK CASTOUT                           = 3
               READ CASTOUT CLASS                       = 5
               READ CASTOUT STATISTICS                  = 6
               READ DIRECTORY INFO                      = 0
DSNB779I -DB1G   ENGINES NOT AVAILABLE
                 FOR CASTOUT                            = 0
                 FOR WRITING                            = 0
```

Figure 63. DISPLAY GROUPBUFFERPOOL MDETAIL report

- 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 castout that's triggered for any other reason).

End of General-use Programming Interface

If the recovery LRSN for group buffer pool checkpoint is not advancing as fast as you want, determine if there have been any DASD or coupling facility connectivity problems that are impairing DB2's ability to cast out.

Group buffer pool thresholds

You can control the castout process by changing the two group buffer pool thresholds:

- Group buffer pool castout threshold
- Class castout threshold

These thresholds have no effect for GBPCACHE(NO) group buffer pools.

As Figure 64 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*.

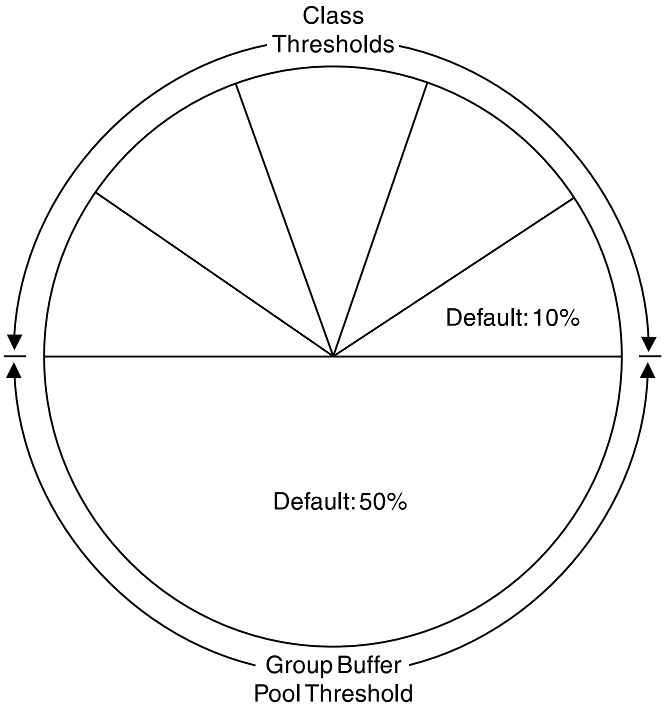


Figure 64. Group buffer pool castout thresholds. Both thresholds are expressed as percentages of the total number of pages in the group buffer pool.

Group buffer pool class castout threshold

In each group buffer pool there is a fixed number of castout class queues. This number is an internal value 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 10%, then DB2 sets CLASST to 100 pages (10% of 1000 pages).

Default group buffer pool class castout threshold: The default for the class castout is 10, which means that castout is initiated for a particular page set or partition when 10 percent of the group buffer pool contains changed pages for the class.

Group buffer pool castout threshold

This 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 contains 5000 data entries and the secondary contains 1000 data entries, and GBPOOLT is 50%, then DB2 sets GBPOOLT to 500 pages (50% of 1000 pages).

Default group buffer pool castout threshold: The default value for the group buffer pool castout threshold is 50, which means that when the group buffer pool is 50 percent full of changed pages, castout is initiated.

Guidelines

In most cases, we suggest you use the default values (10 percent for the class threshold and 50 percent for the group buffer pool threshold). Depending on your work load, these values help reduce DASD 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 DISPLAY GROUPBUFFERPOOL. An example report is shown in Figure 68 on page 249. The field indicated by **E** is the one to watch for this type of problem.

Tuning the castout thresholds: The following can help you more effectively monitor the group buffer pool castout thresholds:

- The DISPLAY GROUPBUFFERPOOL command with the MDETAIL option.
- The DB2 statistics trace.
- 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.

Example from MDETAIL report

General-use Programming Interface

Here is partial output from the command DISPLAY GROUPBUFFERPOOL (GBP0) MDETAIL:

```
⋮
DSNB796I -DB1G CASTOUTS
               PAGES CASTOUT                          = 217
               UNLOCK CASTOUT                         = 35
               READ CASTOUT CLASS                     = 47
               READ CASTOUT STATISTICS                = 47
               READ DIRECTORY INFO                    = 290
⋮
```

The UNLOCK CASTOUT counter should always be significantly less than the PAGES CASTOUT counter. If it is not (for example, if "unlock castout" is more than half of "pages cast out"), then the castout write I/O is not being done efficiently (the number of pages written per I/O is normally close to the number you get by dividing PAGES CASTOUT by UNLOCK CASTOUT). This is probably because you have random update patterns on the DB2 data.

End of General-use Programming Interface

Effect of GBPCACHE ALL on guidelines: If you are using a group buffer pool to cache pages as they are read in from DASD (GBPCACHE ALL page sets), then consider lowering the threshold values to allow more space for caching those clean pages.

Monitoring group buffer pools

This section describes how you can monitor group buffer pool activity:

- “Using the MVS D XCF,STR command”
- “Using the coupling facility activity report of RMF” on page 245
- “Using the DISPLAY GROUPBUFFERPOOL command” on page 245
- “Using DB2 statistics trace” on page 247

Using the MVS D XCF,STR command

You can use MVS 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 DSNDB0G:

```
D XCF,STR,STRNAME=DSNDB0G_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). Output similar to the following is produced:

```
DISPLAY XCF
STRNAME: DSNDB0G_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: 10/14/1997 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: 10/14/1997 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_DB1G         01 00010001 UTEC469  DB1GDBM1 002E ACTIVE NEW,OLD
 DB2_DB2G         02 00020001 UTEC469  DB2GDBM1 0031 ACTIVE NEW,OLD
```

Figure 65. MVS Command D XCF showing group buffer pool information

For more information about the D XCF command, see *OS/390 MVS System Commands*.

Using the coupling facility activity report of RMF

See the portion of an RMF coupling facility structure report show in Figure 66. A value for CHANGD (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 percent or so, and there is a non-zero value in A, it could mean that your configuration does not have enough subchannels to handle the work.

```
STRUCTURE NAME = DSNDB0G_GBP8      TYPE = CACHE
          # REQ    -------------- REQUESTS -------------    -------------- DELAYED REQUESTS -------------
SYSTEM    TOTAL            #     % OF  -SERV TIME(MIC)-    REASON    #   % OF  ---- AVG TIME(MIC) -----
NAME      AVG/SEC          REQ    ALL     AVG   STD_DEV           REQ    REQ   /DEL    STD_DEV   /ALL

STLABC2   66662    SYNC     61K  43.2%   107.7     32.5        A
          370.3    ASYNC  5677    4.0%   602.8    419.8     NO SCH    0   0.0%    0.0        0.0    0.0
              B      CHNGD     0    0.0%  INCLUDED IN ASYNC
                                                           DUMP      0   0.0%    0.0        0.0
```

⋮

Figure 66. Portion of RMF Coupling Facility Structure Activity Report

Using the DISPLAY GROUPBUFFERPOOL command

General-use Programming Interface

Use the DISPLAY GROUPBUFFERPOOL command to display information about group buffer pools. Assume that you want a summary report about group buffer pool zero, including all connections to that group buffer pool. Enter the following command:

```
-DB1G DISPLAY GROUPBUFFERPOOL(GBP0) CONNLIST(YES)
```

Here is what the display might look like, assuming that the group buffer pool is duplexed:

```
DSNB750I -DB1G DISPLAY FOR GROUP BUFFER POOL GBP0 FOLLOWS
DSNB755I -DB1G 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 -DB1G   CLASS CASTOUT THRESHOLD                    = 10%
                 GROUP BUFFER POOL CASTOUT THRESHOLD        = 50%
                 GROUP BUFFER POOL CHECKPOINT INTERVAL      = 8 MINUTES
                 RECOVERY STATUS                            = NORMAL
                 AUTOMATIC RECOVERY                         = Y
DSNB757I -DB1G MVS CFRM POLICY STATUS FOR DSNDB0G_GBP0      = NORMAL
                 MAX SIZE INDICATED IN MVS POLICY           = 61440 KB
                 DUPLEX INDICATOR IN POLICY                 = ENABLED
                 CURRENT DUPLEXING MODE                     = DUPLEX
                 ALLOCATED                                  = YES
DSNB758I -DB1G   ALLOCATED SIZE                             = 61440 KB
                 VOLATILITY STATUS                          = NON-VOLATILE
                 REBUILD STATUS                            = DUPLEXED
                 CFNAME                                     = LF01
                 CFLEVEL                                    = 5
DSNB759I -DB1G   NUMBER OF DIRECTORY ENTRIES                = 61394
                 NUMBER OF DATA PAGES                       = 11370
                 NUMBER OF CONNECTIONS                      = 3
DSNB798I -DB1G LAST GROUP BUFFER POOL CHECKPOINT 17:31:23 MAY  9, 1996
                 GBP CHECKPOINT RECOVERY LRSN              = ACD74C77388C
                 STRUCTURE OWNER                           = DB1G
DSNB799I  -DB1G SECONDARY GBP IS ALLOCATED
                 ALLOCATED SIZE                            = 61440 KB
                 VOLATILITY STATUS                         = NON-VOLATILE
                 CFNAME, CFLEVEL                           = LF01, 5
                 NUMBER OF DIRECTORY ENTRIES               = 61394
                 NUMBER OF DATA PAGES                      = 11370
DSNB766I -DB1G THE CONNLIST REPORT FOLLOWS
DSNB767I -DB1G CONNECTION NAME = DB2_DB1G     , CONNECTION STATUS = A
                   CONNECTOR'S RELEASE                    =    6100
DSNB767I -DB1G CONNECTION NAME = DB2_DB2G     , CONNECTION STATUS = A
                   CONNECTOR'S RELEASE                    =    6100
DSNB767I -DB1G CONNECTION NAME = DB2_DB3G     , CONNECTION STATUS = F
                   CONNECTOR'S RELEASE                    =    6100
DSNB769I -DB1G THE CONNLIST REPORT IS COMPLETE
DSNB790I -DB1G DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1G DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION
```

Figure 67. Summary report with connection list included

See Chapter 2 of *DB2 Command Reference* for more information about the syntax of the command.

Detailed statistics can be displayed using the GDETAIL or MDETAIL keywords. See Figure 68 on page 249 to see what statistical information looks like.

End of General-use Programming Interface

Hint: For an easy way to collect interval statistics for performance analysis, create a batch job that issues the following command periodically:

```
-DB1G DISPLAY GROUPBUFFERPOOL(*) GDETAIL(INTERVAL)
```

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.

Using 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 DB2 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, DB2 PM group-scope statistics reports combine the data of the individual members and present it to for the entire group. The member data is apportioned to the same user-specified interval. DB2 PM presents the synchronized statistics intervals for each member, adds up the counters across all members and presents them as statistics on a per-group basis.

Consider also using DB2 PM to do group-scope exception reporting when a particular counter exceeds a user-specified value.

For more information about using the statistics report, see “What to look for in a DB2 PM statistics report” on page 252.

Determining the correct size and ratio

One of the critical tuning factors in a DB2 data sharing configuration is the size of the group buffer pools. There are three aspects of group buffer pool (cache structure) size that need to be considered:

- Total structure size

  As described in “General information about coupling facility storage” on page 54, 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 virtual buffer pools of three members, that page still has only one directory entry.

  See *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide* for information about the size of directory entries for your CFLEVEL

- Number of data entries

  Data entries are the actual places where the data page resides. These are 4KB, 8KB, 16KB, or 32KB in size (the same size as the data page).

  For GBPCACHE NO group buffer pools, there are no data entries.

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 the first member of the DB2 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 that for the primary.

For formulas to help you choose a ratio, see “Group buffer pool sizes” on page 55.

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 of values that are not ideal for best performance and how you can fix the problems.

Group buffer pool size is too small

When the group buffer pool is too small, the following problems can occur:

- The thresholds for changed pages is reached more frequently, causing data to be cast out to DASD 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. See “Monitor storage of the group buffer pool” on page 250 for a hint about avoiding this problem. See “Problem: storage shortage in the group buffer pool” on page 165 for recovery actions should the problem occur.
- Possibly many cross-invalidations caused by reusing existing directory entries, which might require refreshing a page from DASD later when the page is referenced again.

General-use Programming Interface

In any event, pages in the group buffer pool have to be refreshed from DASD more often because they are not in the group buffer pool. 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:

```
-DB1G DISPLAY GROUPBUFFERPOOL(GBP0) GDETAIL(*)
```

Here is what the detail portion of the report output looks like:

```
DSNB783I -DB1G CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 Mar 17, 1994
DSNB784I -DB1G GROUP DETAIL STATISTICS
                 READS
                   DATA RETURNED A                    = 3845
DSNB785I -DB1G     DATA NOT RETURNED
                     DIRECTORY ENTRY EXISTED B        = 27
                     DIRECTORY ENTRY CREATED C        = 28336
                     DIRECTORY ENTRY NOT CREATED D  = 332, 0

DSNB786I -DB1G   WRITES
                   CHANGED PAGES                      = 20909
                   CLEAN PAGES                        = 0
                   FAILED DUE TO LACK OF STORAGE E    = 8
                 CHANGED PAGES SNAPSHOT VALUE         = 974
DSNB787I -DB1G   RECLAIMS
                   FOR DIRECTORY ENTRIES F            = 18281
                   FOR DATA ENTRIES                   = 47
                 CASTOUTS                             = 16073
DSNB788I -DB1G   CROSS INVALIDATIONS
                   DUE TO DIRECTORY RECLAIMS G        = 4489
                   DUE TO WRITES                      = 3624
                   EXPLICIT                           = 0
DSNB762I -DB1G   DUPLEXING STATISTICS FOR GBP0-SEC
                   WRITES
                     CHANGED PAGES                    = 20909
                     FAILED DUE TO LACK OF STORAGE    = 8
                   CHANGED PAGES SNAPSHOT VALUE       = 974
DSNB790I -DB1G DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I -DB1G DSNB1CMD 'DISPLAY GROUPBUFFERPOOL' NORMAL COMPLETION
```

Figure 68. Example output of group detail statistics

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:

```
( A / ( A + B + C + D (first number)))× 100
```

In our example, the calculation is:

```
(3845 / 32540) × 100 = 11.81%
```

Data was returned in approximately 12 percent 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, as described in “Changing the size of the group buffer pool” on page 256.

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, it is to be expected that not many pages you need would 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, it is unlikely to find any given page in the group buffer pool.

To determine if the low read hit percentage is a problem, see the field indicated by B in the statistics report, shown in Figure 70 on page 253. (The same counter also exists in the accounting report.) Ideally, that field contains 0. A non-zero value there in conjunction with a low read hit percentage can indicate that your group buffer pool is too small.

“Too few directory entries” on page 251 and “Too few data entries” on page 251 describe how to determine if the problem is caused by a suboptimal ratio of directory entries to data entries.

End of General-use Programming Interface

Monitor storage of the group buffer pool

General-use Programming Interface

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.

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. Make sure this snapshot value does not rise significantly above the group buffer pool castout threshold. Figure 69 highlights the key fields from the report.

```
DSNB756I -DB1G   CLASS CASTOUT THRESHOLD                  = 10%
               A GROUP BUFFER POOL CASTOUT THRESHOLD      = 50%
                 GROUP BUFFER POOL CHECKPOINT INTERVAL    = 8 MINUTES
                 RECOVERY STATUS                          = NORMAL
                 AUTOMATIC RECOVERY                       = Y
DSNB759I -DB1G   NUMBER OF DIRECTORY ENTRIES              = 61394
               B NUMBER OF DATA PAGES                     = 11370
                 NUMBER OF CONNECTIONS                    = 3
DSNB783I -DB1G CUMULATIVE GROUP DETAIL STATISTICS SINCE 15:35:23 Mar 17, 1994
DSNB784I -DB1G GROUP DETAIL STATISTICS
⋮
DSNB786I -DB1G   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. Make sure the SNAPSHOT value (C) does not rise significantly above (B × A).

End of General-use Programming Interface

Too few directory entries

General-use Programming Interface

When existing directory entries are being reclaimed to handle new work, cross-invalidation must occur for all the DB2 subsystems that have the particular data pages in their buffer pools, even when the data hasn't actually changed.

For example, in Figure 68 on page 249, **F** indicates that there have been 18281 directory reclaims. **G** shows 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 DASD, which can degrade performance of the system.

If there is a high value in **G**, check the group buffer pool hit percentage (described in “Group buffer pool size is too small” on page 248) to see if the lack of directory entries might be causing an excessive number of reads from DASD.

You can also check the “SYNCHRONOUS READS DUE TO BUFFER INVALIDATION” counters shown in the member detail report.

To increase the number of directory entries in the group buffer pool, you can do one of the following:

- Increase the total size of the group buffer pool, as described in “Changing the size of the group buffer pool” on page 256.
- Use the ALTER GROUPBUFFERPOOL command to adjust the ratio in favor of directory entries, as described in “Changing the ratio of directory to data entries” on page 258.

End of General-use Programming Interface

Too few data entries

General-use Programming Interface

If a group buffer pool does not have enough data entries, then castout to DASD occurs more frequently. 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 the field denoted by **E** in the DISPLAY GROUPBUFFERPOOL GDETAIL report shown in Figure 68 on page 249. 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.

To increase the number of data entries in the group buffer pool, you can do one of the following:

- Increase the total size of the group buffer pool, as described in “Changing the size of the group buffer pool” on page 256.
- Use the ALTER GROUPBUFFERPOOL command to adjust the ratio in favor of data entries, as described in “Changing the ratio of directory to data entries” on page 258.

End of General-use Programming Interface

What to look for in a DB2 PM statistics report

See the DB2 PM statistics detail report shown in Figure 70 on page 253. We'll use fields from that report to explain some of the activity that takes place for cross-invalidation and refresh of buffers. You'll be looking at much of the same type of information described in the DISPLAY GROUPBUFFERPOOL output described earlier in this chapter.

```
GROUP BP4                    QUANTITY  /MINUTE  /THREAD  /COMMIT
---------------------------  --------  -------  -------  -------
SYN.READS(XI)-DATA RETURNED A 3672.00   367.13    99.24     0.13
SYN.READS(XI)-NO DATA RETURN   B 0.00     0.00     0.00     0.00
SYN.READS(NF)-DATA RETURNED C 5106.00   510.50  1405.00     1.91
D
SYN.READS(NF)-NO DATA RETURN 51987.00  5197.66  1405.05     1.91
UNREGISTER PAGE  T              83.00     8.30     2.24     0.00

F
CLEAN PAGES SYNC.WRITTEN         0.00     0.00     0.00     0.00
CHANGED PAGES SYNC.WRITTEN   66058.00  6604.48  1785.35     2.42
G
CLEAN PAGES ASYNC.WRITTEN        0.00     0.00     0.00     0.00
CHANGED PAGES ASYNC.WRITTEN    722.00    72.19    19.51     0.03
 E

ASYNC.READS-DATA RETURNED        0.00     0.00     0.00     0.00
ASYNC.READS-NO DATA RETURNED   182.3K    18.2K0 4927.14     2.47
H
REG.PAGE LIST (RPL) REQUEST   7065.00   706.36    190.95     0.26
CLEAN PAGES READ AFTER RPL    8940.00   893.82    241.62     0.33
CHANGED PGS READ AFTER RPL    6783.00   678.16    183.32     0.25

PAGES CASTOUT   I            67423.00  6740.95  1822.24     2.47
UNLOCK CASTOUT      J        24908.50  2490.30   673.19     0.91
READ CASTOUT CLASS K            00.00     0.00     0.00     0.00
READ CASTOUT STATISTICS  L       0.00     0.00     0.00     0.00

READ DIRECTORY INFO  M           0.00     0.00     0.00     0.00
 N
READ STORAGE STATISTICS  V       4.00     0.40     0.11     0.00
REGISTER PAGE   S               00.00     0.00    00.00     0.00
DELETE NAME   U                  0.00     0.00     0.00     0.00
EXPLICIT X-INVALIDATIONS    W    0.00     0.00     0.00     0.00

CASTOUT CLASS THRESHOLD         00.00     0.00     0.00     0.00
GROUP BP CASTOUT THRESHOLD       0.00     0.00     0.00     0.00
GBP CHECKPOINTS TRIGGERED O      0.00     0.00     0.00     0.00
PARTICIPATION IN GBP REBUILD P   0.00     0.00     0.00     0.00
Q

CASTOUT ENGINE NOT AVAIL.        0.00     0.00     0.00     0.00
WRITE ENGINE NOT AVAILABLE       0.00     0.00     0.00     0.00
R
READ FAILED-NO STORAGE           0.00     0.00     0.00     0.00
WRITE FAILED-NO STORAGE          0.00     0.00     0.00     0.00

WRITE TO SGBP                66780.00  6676.66  1804.86     2.45
WRITE TO SGBP FAILED  X          0.00     0.00     0.00     0.00
DELETE NAME LIST SGBP        24909.00  2490.40   673.22     0.91
DELETE NAME FROM SGBP  Y         0.00     0.00     0.00     0.00
READ CASTOUT STATS SGBP          0.00     0.00     0.00     0.00
```

Figure 70. Portion of DB2 PM statistics detail report showing GBP activity

Explanation of fields

A The number of reads to the group buffer pool that were required because the page was invalidated in the member's buffer pool. The member did find the needed data in the group buffer pool.

B The number of reads to the group buffer pool that were required because the page was invalidated in the member's buffer pool. The member did not find the needed data in the group buffer pool and had to go to DASD to retrieve the page.

C The number of reads to the group buffer pool that were required because the page was not in the member's buffer pool. The member did find the needed data in the group buffer pool.

D The number of reads to the group buffer pool that were required because the page was not in the member's buffer pool. The member did not find the needed data in the group buffer pool and had to go to DASD to find the page.

E This section of counters show asynchronous reads from the group buffer pool. DB2 uses the term *asynchronous* to mean that the request was done under a system execution unit, asynchronous to the allied work unit.

Asynchronous reads can occur for prefetch processing, and for those cases when P-lock negotiation necessitates registering pages cached in the member's buffer pool.

F These counters indicate the number of changed and clean pages that were synchronously written to the group buffer pool from the virtual pool.

G These counters indicate the number of changed and clean pages that were asynchronously written to the group buffer pool from the virtual pool. Pages can be forced out before the application commits if a 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.

H This section of counters reflect the DB2's use of the ability to register a list of pages for prefetch activity. See “Prefetch processing” on page 232 for more information about this.

I The number of unlock requests issued for casting out of the group buffer pool. The number of data pages that were cast out of the group buffer pool.

J The number of unlock-for-castout requests was issued for castout I/Os that have completed. When pages are in the process of being cast out to DASD, they are “locked for castout” in the coupling facility. This castout lock is not an IRLM lock; its purpose is to enforce that only one system at a time is casting out a particular page.

DB2 usually includes more than one page in the request to write pages to DASD. Therefore, this counter should be less-than or equal-to the value in **I**.

K The number of requests made to the group buffer pool to determine which pages that belong to a particular page set or partition must be cast out because they are cached as changed pages. This request is issued by the page set or partition castout owner and, when the group buffer pool castout threshold is reached, by the group buffer pool structure owner.

L The number of requests issued by the group buffer pool structure owner to determine which castout classes have pages that are changed. This request is issued by the group buffer pool structure owner when the group buffer pool threshold is reached. The request is generally issued only once or twice for each occurrence of the group buffer pool threshold.

This request is also issued for group buffer pool checkpoint when your Sysplex is enabled to take advantage of the group buffer pool checkpoint enhancement.

M 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 when your Sysplex is not enabled for the group buffer pool checkpointing performance enhancement. The purpose of the request is to record the oldest recovery LRSN, which is used as a basis for recovery if the group buffer pool fails. See “Tuning the group buffer pool checkpoint interval” on page 239 for more information.

N These counters indicate how many times the castout class and group buffer pool castout thresholds have been reached.

O The number of group buffer pool checkpoints that occurred.

P The number of times this member participated in rebuilding this group buffer pool.

Q These counters indicate a problem obtaining the capability to write to the group buffer pool or to cast out to DASD. These numbers should be very low, ideally 0.

R These counters indicate storage problems in the coupling facility. You might need to increase the size of the group buffer pool if these numbers are large and are not reflecting just a momentary surge in activity.

S 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.

T The number of times DB2 unregistered interest for a single page. This happens when DB2 steals pages from the local buffer pool that belong to GBP-dependent page sets or partitions.

U The number of times DB2 issued a request to delete directory and data entries associated with a particular page set or partition. DB2 issues this request when it converts 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.

V The number of times DB2 requested statistics information from the group buffer pool. It is issued once on a timer interval basis by the group buffer pool structure owner to detect when the GBPOOLT threshold is reached. It is also issued whenever the DISPLAY GROUPBUFFERPOOL GDETAIL command is issued or when IFCID 0254 is recorded.

W The number of explicit cross-invalidations that are issued. A non-zero value indicates that this is a GBPCACHE(NO) group buffer pool or that objects in this group buffer pool are defined with GBPCACHE NONE or SYSTEM storage.

X The number of times a write to the secondary group buffer pool failed because of lack of storage.

Y The number of times a “delete name list” request was issued for the secondary group buffer pool. The delete name list request is used to clean out the secondary group buffer pool after the primary group buffer pool has been cast out.

Changing group buffer pools

The information under this heading, up to “Access path selection in a data sharing group” on page 258, is General-use Programming Interface and Associated Guidance Information as defined in Appendix B, “Notices” on page 265.

This section describes how you can change attributes of the group buffer pool. The following tasks are described:

- “Changing the castout threshold values”
- “Changing the checkpoint frequency”
- “Changing the size of the group buffer pool”
- “Changing the ratio of directory to data entries” on page 258

If you want to start or stop duplexing for a group buffer pool, see “Starting and stopping duplexing for a group buffer pool” on page 178. If you want to make hardware changes to the coupling facility or move a group buffer pool from one coupling facility to another, see “Shutting down the coupling facility” on page 179.

Changing the castout threshold values

Use ALTER GROUPBUFFERPOOL to change the group buffer pool castout thresholds. For example, the following command:

```
-DB1G ALTER GROUPBUFFERPOOL(GBP1) CLASST(15) GBPOOLT(55)
```

changes the class castout threshold to 15 percent and the group buffer pool threshold to 55 percent. These changes take effect immediately.

Changing the checkpoint frequency

Use ALTER GROUPBUFFERPOOL to change the group buffer pool castout thresholds. For example, to indicate that you want group buffer pool checkpoints to occur every 3 minutes, enter the following command:

```
-DB1G ALTER GROUPBUFFERPOOL(GBP1) GBPCHKPT(3)
```

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 DSNDB0G):

```
SETXCF START,ALTER,STRNAME=DSNDB0G_GBPn,SIZE=newsize
```

This example assumes that *newsize* is less than or equal to the maximum size defined 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 in “Static method” on page 258.

Assume a DISPLAY GROUPBUFFERPOOL command shows the following:

```
⋮
DSNB757I -DB1G MVS CFRM POLICY STATUS FOR DSNDB0G_GBP1 = NORMAL
             MAX SIZE INDICATED IN POLICY             = 4096 KB
             ALLOCATED                                = YES
DSNB758I -DB1G ALLOCATED SIZE                         = 1024 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = NONE
               DUPLEXING STATUS                       = SIMPLEXED
               CFNAME, CFLEVEL                        = LF01, 5
DSNB759I -DB1G NUMBER OF DIRECTORY ENTRIES            = 924
               NUMBER OF DATA PAGES                   = 180
               NUMBER OF CONNECTIONS                  = 2
⋮
```

And then you enter the following MVS command to increase the size:

```
SETXCF START,ALTER,STRNM=DSNDB0G_GBP1,SIZE=1536
```

Here's what the DISPLAY GROUPBUFFERPOOL command output might look like after you alter the size:

```
⋮
DSNB757I -DB1G MVS CFRM POLICY STATUS FOR DSNDB0G_GBP1 = NORMAL
             MAX SIZE INDICATED IN POLICY             = 4096 KB
             ALLOCATED                                = YES
DSNB758I -DB1G ALLOCATED SIZE                         = 1536 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = NONE
               DUPLEXING STATUS                       = SIMPLEXED
               CFNAME, CFLEVEL                        = LF01, 5
DSNB759I -DB1G NUMBER OF DIRECTORY ENTRIES            = 1426
               NUMBER OF DATA PAGES                   = 284
               NUMBER OF CONNECTIONS                  = 2
⋮
```

Notice that the allocated size has increased and the numbers of directory entries and data pages have increased as well. 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.

Then you must use the following procedure. Because the group buffer pool must be rebuilt, use this procedure when there is less activity in the group.

1. Increase the storage for the group buffer pool in the CFRM policy.
2. Use the following MVS command to start the updated policy:

   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=policyname
   ```

3. Use the following command to rebuild the group buffer pool:

   ```
   SETXCF START,REBUILD,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 5 directory entries to every one data page), you can use the ALTER GROUPBUFFERPOOL command shown here to increase the ratio to 7:

```
-DB1G ALTER GROUPBUFFERPOOL (GBP0) RATIO (7)
```

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. The procedure, then, is this:

1. Stop duplexing, as described in “Stopping duplexing” on page 179 to revert the group buffer pool to simplex mode.
2. Alter the group buffer pool ratio and issue the SETXCF START,REBUILD command.
3. Start duplexing, as described in “Starting duplexing” on page 178.

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 command.

DB2 does let you duplex a GBPCACHE(NO) group buffer pool. If the group buffer pool is currently GBPCACHE(YES) but the NO attribute is pending, then if you start a duplexing rebuild, DB2 ignores the pending GBPCACHE(NO) attribute.

Access path selection in a data sharing group

This section describes the following:

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, 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 (CP) might change your access path. This effect is more likely 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 the original bind occurred on.

Using EXPLAIN in a data sharing group

Product-sensitive Programming Interface

EXPLAIN informs you about access paths DB2 chooses. Because EXPLAIN can be run on one DB2 subsystem and a plan can be bound and executed on other DB2 subsystems in a data sharing group, it is important to know which member performed the EXPLAIN. The PLAN_TABLE column GROUP_MEMBER contains the member name of the DB2 that performed the EXPLAIN. The column is blank if the DB2 subsystem was not in a data sharing environment when the EXPLAIN was performed.

End of Product-sensitive Programming Interface

Appendix A. DB2 and IRLM names

This appendix includes information about name formats for the licensed program DB2 for OS/390 and its IRLM.

DB2 group names

Table 38. Group Names. There is one set of names per group.

Name	Length	Example Format	Comments
Catalog alias	8	*catalias*	You must place DB2's catalog alias in the MVS master catalog.
Catalog and directory database names		*catalias*.DSNDB01 *catalias*.DSNDB06	All DB2s in the group share the same catalog and directory.
DSNHDECP	8	DSNHDECP	This resides in SDSNEXIT. There is a single load module for the group.
Generic LU name	8	*xxxxxxxx*	One name per group. Requesters use this name to configure their communications control information.
Group attachment name	4	*gssn*	This name can be used by TSO/batch, CAF, RRSAF, and utilities as a generic attachment name.
Group buffer pools (coupling facility cache structures)	16	*groupname*_GBP*xxxx*	You must enter this name on the CFRM policy.
Group name	8	*xxxxxxxx*	This name must be unique within the Sysplex.
location name	16	*xxxxxxxxxxxxxxxx*	One name per group. Requesters use this name in their SQL applications.
lock structure	16	*groupname*_LOCK1	You must enter this name on the CFRM policy.
shared communications area (coupling facility list structure)	16	*groupname*_SCA	You must enter this name on the CFRM policy.
Target libraries		*catalias*.SDSNCLST *catalias*.SDSNLINK *catalias*.SDSNLOAD *catalias*.SDSNEXIT	Target libraries can be shared among DB2 group members or can be replicated.

DB2 member names

Member names include the individual DB2 member (subsystem) name and its associated MVS subsystem name, procedure names, and BSDS and log names.

Table 39 (Page 1 of 2). Member Names. There is one set of names per member.

Name	Length	Example Format	Comments
Active log prefixes	30	*membname*.LOGCOPY1 *membname*.LOGCOPY2	The catalog alias can be added as first qualifier.
Archive log prefixes	35 or 19	*membname*.ARCLG1 *membname*.ARCLG2	The catalog alias can be added as the first qualifier. There are only 19 characters for the prefix if the name is timestamped.

Table 39 (Page 2 of 2). Member Names. There is one set of names per member.

Name	Length	Example Format	Comments
BSDS	33	*membname*.BSDS01 *membname*.BSDS02	The catalog alias can be added as the first qualifier.
SCA and group buffer pool connection names	16	DB2_*membername*	The connection name is generated by DB2. You see it only in certain commands (MVS D XCF,STRUCTURE and the DB2 connection list display of DISPLAY GROUPBUFFERPOOL).
Command prefix	8	*xmembcpf*	The first character must be a special character.
LU name	8	*luname*	This name must be unique within the data sharing group and the network.
Member name	8	*membname*	This name must be unique within the data sharing group. If the member name is also used as the high level qualifier for the member's data sets (BSDS, logs, and so on), then member names must be unique within the MVS Sysplex. This is because the MVS Sysplex can have a shared master catalog.
Procedure names	8	*mssn*MSTR *mssn*DBM1 *mssn*DIST *mssn*SPAS	These names are generated from the member subsystem name at installation time.
Subsystem name	4	*mssn*	This name is used by all the attachment interfaces and must be unique within the MVS Sysplex.
Subsystem parameters load module	8	DSNZP*xxx*	Resides in SDSNEXIT. The name is specified as a parameter on *mssn*MSTR procedure.
XCF member name	16	*membname*	This is the same as DB2's member name; thus, a maximum of 8 characters is used.
Work file database	8	*mworkdb*	Work file data sets have names of the format *catalias*.DSNDBC.*mworkdb*.DSN*kknn*.I0001.A001

IRLM names

Each DB2 subsystem in the data sharing group has an associated IRLM. Each member's IRLM and its associated procedures must be named. The IRLM group name, subsystem name, and member ID are parameters on the IRLM proc. This requires a separate IRLM proc for every IRLM in the group.

Table 40 (Page 1 of 2). IRLM Names. There is one set of names per DB2 member.

Name	Length	Example Format	Comments
IRLM group name	8	*xxxxxxxx*	This is a parameter on *issn*IRLM proc. It must be unique across the Sysplex, to avoid name conflicts.
IRLM subsystem name	4	*issn*	This is a parameter on the *issn* IRLM procedure. This subsystem name must be unique within the data sharing group to avoid the problem of DB2 connecting to the wrong IRLM.

Table 40 (Page 2 of 2). IRLM Names. There is one set of names per DB2 member.

Name	Length	Example Format	Comments
IRLM procedure name	8	*mssn*IRLM	Each DB2 member knows its IRLM by the procedure and subsystem name saved in that member's subsystem parameter load module.
IRLM member ID	3	This is a number between 1 and 255 (inclusive).	This ID uniquely names an IRLM within a group. It is a parameter on *issn*IRLM procedure and must be unique within the data sharing group.
IRLM member XCF name	16	*xxxxxxxx*$*issnNNN*	*xxxxxxxx* is the IRLM group name, *issn* is the IRLM subsystem ID, and *NNN* is the IRLM member ID. Dollar signs ($) are used as padding. This name is generated at startup time.
lock structure connection name	16	*xxxxxxxx*$*issnNNN*	The connection name is the same as the IRLM XCF member name. It is generated by IRLM. You see it only as the output of certain commands (such as MVS D XCF,STRUCTURE). In some cases, the connection name can be of the format *xxxxxxxx#issnNNN*.

Appendix B. 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:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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 publication to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is as your own risk.

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
J74/G4
555 Bailey Avenue
P.O. Box 49023
San Jose, CA 95161-9023
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 information 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 illustrates 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.

Programming interface information

This book is intended to help you plan for the use of DB2 data sharing.

This book also documents General-use Programming Interface and Associated Guidance Information and Product-sensitive Programming Interface and Associated Guidance Information provided by DB2 for OS/390 (DB2).

General-use programming interfaces allow the customer to write programs that obtain the services of DB2.

General-use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

General-use programming interface

General-use Programming Interface and Associated Guidance Information ...

End of General-use programming interface

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of DB2. 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 marking:

Product-sensitive programming interface

Product-sensitive Programming Interface and Associated Guidance Information ...

End of Product-sensitive programming interface

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, in other countries, or both:

AIX
BookManager
CICS
CICS/ESA
CICSPlex
DATABASE 2
DataPropagator
DB2
DB2 Connect
DB2 Universal Database
DFSMS
DFSMShsm
Distributed Relational Database Architecture
DRDA
eNetwork
Enterprise System/3090
Enterprise System/9000
IBM
IMS
MVS/ESA
Net.Data
OS/2
OS/390
Parallel Sysplex
QMF
PR/SM
RACF
RAMAC
RMF
Sysplex Timer
System/390
S/390
System/390
VTAM

Other company, product, and service names may be trademarks or service marks of others.

The Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or in other countries.

Microsoft, Windows, Windows NT, and the Windows log are trademarks of Microsoft™ Corporation in the United States and/or other countries.

Glossary

The following terms and abbreviations are defined as they are used in the DB2 library. If you do not find the term you are looking for, refer to the index or to *IBM Dictionary of Computing*.

A

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, whereas the archive log holds those records that are older and no longer fit on the active log.

active member state. A state of a member of a data sharing group. An active member is identified with a group by XCF, which associates the member with a particular task, address space, and MVS system. A member that is not active has either a failed member state or a quiesced member state.

archive log. The portion of the DB2 log that contains log records that have been copied from the active log.

B

base table. (1) A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with *result table* and *temporary table*.

(2) A table containing a LOB column definition. The actual LOB column data is not stored with the base table. The base table contains a row identifier for each row and an indicator column for each of its LOB columns. Contrast with *auxiliary table*.

binary large object (BLOB). A sequence of bytes, where the size of the value ranges from 0 bytes to 2 GB - 1. Such a string does not have an associated CCSID.

BLOB. Binary large object.

built-in function. A function that DB2 supplies. Contrast with *user-defined function*.

C

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.

castout. The DB2 process of writing changed pages from a group buffer pool to DASD.

castout owner. The DB2 member that is responsible for casting out a particular page set or partition.

CEC. Central electronic complex. See *central processor complex*.

central electronic complex (CEC). See *central processor complex*.

central processor complex (CPC). A physical collection of hardware (such as an ES/3090) that consists of main storage, one or more central processors, timers, and channels.

CFRM policy. A declaration by an MVS administrator regarding the allocation rules for a coupling facility structure.

character large object (CLOB). A sequence of bytes representing single-byte characters or a mixture of single- and double-byte characters where the size of the value can be up to 2 GB - 1. In general, character large object values are used whenever a character string might exceed the limits of the VARCHAR type.

CLOB. Character large object.

coexistence. During migration, the period of time in which two releases exist in the same data sharing group.

command prefix. A one- to eight-character command identifier. The command prefix distinguishes the command as belonging to an application or subsystem rather than to MVS.

command scope. The scope of command operation in a data sharing group. If a command has *member scope*, the command displays information only from the one member or affects only non-shared resources that are owned locally by that member. If a command has *group scope*, the command displays information from all members, affects non-shared resources that are owned locally by all members, displays information on sharable resources, or affects sharable resources.

coupling facility. A special PR/SM™ LPAR logical partition that runs the coupling facility control program and provides high-speed caching, list processing, and locking functions in a Sysplex.

CPC. Central processor complex.

cross-system coupling facility (XCF). A component of MVS that provides functions to support cooperation between authorized programs running within a Sysplex.

cross-system extended services (XES). A set of MVS 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.

D

database exception status. An indication that something is wrong with a database. All members of a data sharing group must know and share the exception status of databases.

data sharing. The ability of two or more DB2 subsystems to directly access and change a single set of data.

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 space. A range of up to 2 GB of contiguous virtual storage addresses that a program can directly manipulate. Unlike an address space, a data space can hold only data; it does not contain common areas, system data, or programs.

DBCLOB. Double-byte character large object.

degree of parallelism. The number of concurrently executed operations that are initiated to process a query.

distinct type. A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

DNS. Domain name server.

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 sequence of bytes representing double-byte characters where the size of the values can be up to 2 GB. In general, double-byte character large object values are used whenever a double-byte character string might exceed the limits of the VARGRAPHIC type.

E

exclusive lock. A lock that prevents concurrently executing application processes from reading or changing data. Contrast with *shared lock*.

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.

F

failed member state. A state of a member of a data sharing group. When a member fails, the XCF permanently records the failed member state. This state usually means that the member's task, address space, or MVS system terminated before the state changed from active to quiesced.

false global lock contention. A contention indication from the coupling facility when multiple lock names are hashed to the same indicator and when no real contention exists.

G

GBP. 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 DASD.

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.

global lock. A lock that provides concurrency control within and among DB2 subsystems. The scope of the lock is across all the 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.

gross lock. The *shared*, *update*, or *exclusive* mode locks on a table, partition, or table space.

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 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*. OS/390 publications refer to these instances as the 'old' (for primary) and 'new' (for secondary) structures.

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 MVS 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.

I

index partition. A VSAM data set that is contained within a partitioning index space.

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.

IP address. A 4-byte value that uniquely identifies a TCP/IP host.

L

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 - 1 byte in length. See also *BLOB*, *CLOB*, and *DBCLOB*.

list structure. A coupling facility structure that lets data be shared and manipulated as elements of a queue.

L-lock. Logical lock.

LOB. 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 table space. A table space that contains all the data for a particular LOB column in the related base table.

| **locale**. The definition of a subset of a user's
| environment that combines characters that are defined
| for a specific language and country, and a CCSID.

local lock. A lock that provides intra-DB2 concurrency control, but not inter-DB2 concurrency control; that is, its scope is a single DB2.

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.

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 parent. For explicit hierarchical locking, a lock that is held on a resource that has child locks that are lower in the hierarchy; usually the table space or partition intent locks are the parent locks.

lock promotion. The process of changing the size or mode of a DB2 lock to a higher level.

lock size. The amount of data 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.

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 *P-lock*.

logically complete. A state in which the concurrent copy process is finished with the initialization of the target objects 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 DASD) might not contain any errors. Usually a connection to the media has been lost.

log initialization. The first phase of restart processing during which DB2 attempts to locate the current end of the log.

log record sequence number (LRSN). A number that DB2 generates and associates with each log record. DB2 also uses the LRSN for page versioning. The LRSNs that a particular DB2 data sharing group generates form a strictly increasing sequence for each DB2 log and a strictly increasing sequence for each page across the DB2 group.

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. Logical page list.

LRSN. Log record sequence number.

M

member name. The MVS XCF identifier for a particular DB2 subsystem in a data sharing group.

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 fails, that DB2 subsystem's modify locks are converted to retained locks.

N

negotiable lock. A lock whose mode can be downgraded, by agreement among contending users, to be compatible to all. A physical lock is an example of a negotiable lock.

nonpartitioning index. Any index that is not a partitioning index.

P

page version number. A 6-byte field in a page header that is strictly increasing.

Parallel Sysplex. A set of MVS systems that communicate and cooperate with each other through certain multisystem hardware components and software services to process customer workloads.

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.

physical lock (P-lock). A lock type 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)*.

physical lock contention. Conflicting states of the requesters for a physical lock. See *negotiable lock*.

physically complete. The state in which the concurrent copy process is completed and the output data set has been created.

P-lock. Physical lock.

policy. See *CFRM policy*.

primary group buffer pool. For a duplexed group buffer pool, the structure used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The OS/390 equivalent is *old* structure. Compare with *secondary group buffer pool*.

Q

quiesced member state. A state of a member of a data sharing group. An active member becomes quiesced when a STOP DB2 command takes effect without a failure. If the member's task, address space, or MVS system fails before the command takes effect, the member state is failed.

R

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 failure.

S

SCA. Shared communications area.

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 OS/390 equivalent is *new* structure.

shared communications area (SCA). A coupling facility list structure that a DB2 data sharing group uses for inter-DB2 communication.

shared lock. A lock that prevents concurrently executing application processes from changing data, but not from reading data. Contrast with *exclusive lock*.

structure. A construct that uses MVS to map and manage storage on a coupling facility. See *cache structure*, *list structure*, or *lock structure*.

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

Sysplex. See *Parallel Sysplex*.

T

table space. A page set that is used to store the records in one or more tables.

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.

# **temporary table**. A table that holds temporary data;
# for example, temporary tables are useful for holding or
# sorting intermediate results from queries that contain a
# large number of rows. The two kinds 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*.

transaction lock. A lock that is used to control concurrent execution of SQL statements.

trigger package. A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

triggering table. The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

type 1 indexes. Indexes that were created by a release of DB2 before DB2 Version 4 or that are specified as type 1 indexes in Version 4. Contrast with *type 2 indexes*. As of Version 6, type 1 indexes are no longer supported.

type 2 indexes. Indexes that are created on a release of DB2 after Version 5 or that are specified as type 2 indexes in Version 4 or Version 5.

X

XCF. See *cross-system coupling facility*.

XES. See *cross-system extended services*.

Bibliography

DB2 Universal Database Server for OS/390 Version 6 Product Libraries:

DB2 Universal Database for OS/390

- *DB2 Administration Guide, SC26-9003*
- *DB2 Application Programming and SQL Guide, SC26-9004*
- *DB2 Application Programming Guide and Reference for Java™, SC26-9018*
- *DB2 ODBC Guide and Reference, SC26-9005*
- *DB2 Command Reference, SC26-9006*
- *DB2 Data Sharing: Planning and Administration, SC26-9007*
- *DB2 Data Sharing Quick Reference Card, SX26-3843*
- *DB2 Diagnosis Guide and Reference, LY36-3736*
- *DB2 Diagnostic Quick Reference Card, LY36-3737*
- *DB2 Image, Audio, and Video Extenders Administration and Programming, SC26-9650*
- *DB2 Installation Guide, GC26-9008*
- *DB2 Licensed Program Specifications, GC26-9009*
- *DB2 Messages and Codes, GC26-9011*
- *DB2 Master Index, SC26-9010*
- *DB2 Reference for Remote DRDA Requesters and Servers, SC26-9012*
- *DB2 Reference Summary, SX26-3844*
- *DB2 Release Planning Guide, SC26-9013*
- *DB2 SQL Reference, SC26-9014*
- *DB2 Text Extender Administration and Programming, SC26-9651*
- *DB2 Utility Guide and Reference, SC26-9015*
- *DB2 What's New? GC26-9017*
- *DB2 Program Directory, GI10-8182*

DB2 Administration Tool

- *DB2 Administration Tool for OS/390 User's Guide, SC26-9847*

DB2 Buffer Pool Tool

- *DB2 Buffer Pool Tool for OS/390 User's Guide and Reference, SC26-9306*

DB2 DataPropagator

- *DB2 Replication Guide and Reference, SC26-9642*

Net.Data for OS/390

The following books are available at # *http://www.ibm.com/software/net.data/library.html:*

- *Net.Data Library: Administration and Programming Guide for OS/390*
- *Net.Data Library: Language Environment Interface Reference*
- *Net.Data Library: Messages and Codes*
- *Net.Data Library: Reference*

DB2 PM for OS/390

- *DB2 PM for OS/390 Batch User's Guide, SC26-9167*
- *DB2 PM for OS/390 Command Reference, SC26-9166*
- *DB2 PM for OS/390 General Information, GC26-9172*
- *DB2 PM for OS/390 Installation and Customization, SC26-9171*
- *DB2 PM for OS/390 Messages, SC26-9169*
- *DB2 PM for OS/390 Online Monitor User's Guide, SC26-9168*
- *DB2 PM for OS/390 Report Reference Volume 1, SC26-9164*
- *DB2 PM for OS/390 Report Reference Volume 2, SC26-9165*
- *DB2 PM for OS/390 Using the Workstation Online Monitor, SC26-9170*
- *DB2 PM for OS/390 Program Directory, GI10-8183*

Query Management Facility

- *Query Management Facility: Developing QMF Applications, SC26-9579*
- *Query Management Facility: Getting Started with QMF on Windows, SC26-9582*
- *Query Management Facility: High Peformance Option User's Guide for OS/390, SC26-9581*
- *Query Management Facility: Installing and Managing QMF on OS/390, GC26-9575*
- *Query Management Facility: Installing and Managing QMF on Windows, GC26-9583*
- *Query Management Facility: Introducing QMF, GC26-9576*
- *Query Management Facility: Messages and Codes, GC26-9580*
- *Query Management Facility: Reference, SC26-9577*
- *Query Management Facility: Using QMF, SC26-9578*

Ada/370

- *IBM Ada/370 Language Reference, SC09-1297*
- *IBM Ada/370 Programmer's Guide, SC09-1414*
- *IBM Ada/370 SQL Module Processor for DB2 Database Manager User's Guide, SC09-1450*

APL2

- *APL2 Programming Guide, SH21-1072*
- *APL2 Programming: Language Reference, SH21-1061*
- *APL2 Programming: Using Structured Query Language (SQL), SH21-1057*

AS/400

- *DB2 for OS/400 SQL Programming, SC41-4611*
- *DB2 for OS/400 SQL Reference, SC41-4612*

BASIC

- *IBM BASIC/MVS Language Reference, GC26-4026*
- *IBM BASIC/MVS Programming Guide, SC26-4027*

BookManager READ/MVS

- *BookManager READ/MVS V1R3: Installation Planning & Customization, SC38-2035*

C/370

- *IBM SAA AD/Cycle C/370 Programming Guide, SC09-1841*
- *IBM SAA AD/Cycle C/370 Programming Guide for Language Environment/370, SC09-1840*
- *IBM SAA AD/Cycle C/370 User's Guide, SC09-1763*
- *SAA CPI C Reference, SC09-1308*

Character Data Representation Architecture

- *Character Data Representation Architecture Overview, GC09-2207*
- *Character Data Representation Architecture Reference and Registry, SC09-2190*

CICS/ESA

- *CICS/ESA Application Programming Guide, SC33-1169*
- *CICS for MVS/ESA Application Programming Reference, SC33-1170*
- *CICS for MVS/ESA CICS-RACF Security Guide, SC33-1185*
- *CICS for MVS/ESA CICS-Supplied Transactions, SC33-1168*
- *CICS for MVS/ESA Customization Guide, SC33-1165*
- *CICS for MVS/ESA Data Areas, LY33-6083*
- *CICS for MVS/ESA Installation Guide, SC33-1163*
- *CICS for MVS/ESA Intercommunication Guide, SC33-1181*
- *CICS for MVS/ESA Messages and Codes, GC33-1177*
- *CICS for MVS/ESA Operations and Utilities Guide, SC33-1167*
- *CICS/ESA Performance Guide, SC33-1183*
- *CICS/ESA Problem Determination Guide, SC33-1176*
- *CICS for MVS/ESA Resource Definition Guide, SC33-1166*
- *CICS for MVS/ESA System Definition Guide, SC33-1164*
- *CICS for MVS/ESA System Programming Reference, GC33-1171*

CICS/MVS

- *CICS/MVS Application Programmer's Reference, SC33-0512*
- *CICS/MVS Facilities and Planning Guide, SC33-0504*
- *CICS/MVS Installation Guide, SC33-0506*
- *CICS/MVS Operations Guide, SC33-0510*
- *CICS/MVS Problem Determination Guide, SC33-0516*
- *CICS/MVS Resource Definition (Macro), SC33-0509*
- *CICS/MVS Resource Definition (Online), SC33-0508*

IBM C/C++ for MVS/ESA

- *IBM C/C++ for MVS/ESA Library Reference, SC09-1995*
- *IBM C/C++ for MVS/ESA Programming Guide, SC09-1994*

IBM COBOL

- *IBM COBOL Language Reference, SC26-4769*
- *IBM COBOL for MVS & VM Programming Guide, SC26-4767*

Conversion Guide

- *IMS-DB and DB2 Migration and Coexistence Guide, GH21-1083*

Cooperative Development Environment

- *CoOperative Development Environment/370: Debug Tool, SC09-1623*

Data Extract (DXT)

- *Data Extract Version 2: General Information, GC26-4666*
- *Data Extract Version 2: Planning and Administration Guide, SC26-4631*

DataPropagator NonRelational

- *DataPropagator NonRelational MVS/ESA Administration Guide, SH19-5036*
- *DataPropagator NonRelational MVS/ESA Reference, SH19-5039*

Data Facility Data Set Services

- *Data Facility Data Set Services: User's Guide and Reference, SC26-4388*

Database Design

- *DB2 Design and Development Guide, Gabrielle Wiorkowski and David Kull, Addison Wesley, ISBN 0-20158-049-8*
- *Handbook of Relational Database Design, C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8*

DataHub

- *IBM DataHub General Information, GC26-4874*

DB2 Connect

- *DB2 Connect Enterprise Edition for OS/2 and Windows NT: Quick Beginnings, GC09-2828*
- *DB2 Connect Personal Edition Quick Beginnings, GC09-2830*
- *DB2 Connect User's Guide, SC09-2838*

DB2 Server for VSE & VM

- *DB2 Server for VM: DBS Utility, SC09-2394*
- *DB2 Server for VSE: DBS Utility, SC09-2395*

DB2 Universal Database (UDB)

- *DB2 UDB Administration Guide Volume 1: Design and Implementation, SC09-2839*
- *DB2 UDB Administration Guide Volume 2: Performance, SC09-2840*
- *DB2 UDB Administrative API Reference, SC09-2841*
- *DB2 UDB Application Building Guide, SC09-2842*
- *DB2 UDB Application Development Guide, SC09-2845*
- *DB2 UDB Call Level Interface Guide and Reference, SC09-2843*
- *DB2 UDB SQL Getting Started, SC09-2856*
- *DB2 UDB SQL Reference Volume 1, SC09-2847*
- *DB2 UDB SQL Reference Volume 2, SC09-2848*

Device Support Facilities

- *Device Support Facilities User's Guide and Reference, GC35-0033*

DFSMS/MVS

- *DFSMS/MVS: Access Method Services for the Integrated Catalog, SC26-4906*
- *DFSMS/MVS: Access Method Services for VSAM Catalogs, SC26-4905*
- *DFSMS/MVS: Administration Reference for DFSMSdss, SC26-4929*
- *DFSMS/MVS: DFSMShsm Managing Your Own Data, SH21-1077*
- *DFSMS/MVS: Diagnosis Reference for DFSMSdfp, LY27-9606*
- *DFSMS/MVS Storage Management Library: Implementing System-Managed Storage, SC26–3123*
- *DFSMS/MVS: Macro Instructions for Data Sets, SC26-4913*
- *DFSMS/MVS: Managing Catalogs, SC26-4914*
- *DFSMS/MVS: Program Management, SC26-4916*
- *DFSMS/MVS: Storage Administration Reference for DFSMSdfp, SC26-4920*
- *DFSMS/MVS: Using Advanced Services, SC26-4921*
- *DFSMS/MVS: Utilities, SC26-4926*
- *MVS/DFP: Using Data Sets, SC26-4749*

DFSORT

- *DFSORT Application Programming: Guide, SC33-4035*

Distributed Relational Database

- *Data Stream and OPA Reference, SC31-6806*
- *IBM SQL Reference, SC26-8416*
- *Open Group Technical Standard (the Open Group presently makes the following books available through its Web site at http://www.opengroup.org):*
  - *DRDA Volume 1: Distributed Relational Database Architecture (DRDA), ISBN 1-85912-295-7*
  - *DRDA Version 2 Volume 2: Formatted Data Object Content Architecture, available only on Web* # # #
  - *DRDA Volume 3: Distributed Database Management (DDM) Architecture, ISBN 1-85912-206-X*

Domain Name System

- *DNS and BIND, Third Edition, Paul Albitz and Cricket Liu, O'Reilly, SR23-8771*

Education

- *IBM Dictionary of Computing, McGraw-Hill, ISBN 0-07031-489-6*
- *1999 IBM All-in-One Education and Training Catalog, GR23-8105*

Enterprise System/9000 and Enterprise System/3090

- *Enterprise System/9000 and Enterprise System/3090 Processor Resource/System Manager Planning Guide, GA22-7123*

High Level Assembler

- *High Level Assembler for MVS and VM and VSE Language Reference, SC26-4940*
- *High Level Assembler for MVS and VM and VSE Programmer's Guide, SC26-4941*

Parallel Sysplex Library

- *OS/390 Parallel Sysplex Application Migration, GC28-1863*
- *System/390 MVS Sysplex Hardware and Software Migration, GC28-1862*
- *OS/390 Parallel Sysplex Overview: An Introduction to Data Sharing and Parallelism, GC28-1860*
- *OS/390 Parallel Sysplex Systems Management, GC28-1861*
- *OS/390 Parallel Sysplex Test Report, GC28-1963*
- *System/390 9672/9674 System Overview, GA22-7148*

ICSF/MVS

- *ICSF/MVS General Information, GC23-0093*

IMS/ESA

- *IMS Batch Terminal Simulator General Information, GH20-5522*
- *IMS/ESA Administration Guide: System, SC26-8013*
- *IMS/ESA Administration Guide: Transaction Manager, SC26-8731*
- *IMS/ESA Application Programming: Database Manager, SC26-8727*
- *IMS/ESA Application Programming: Design Guide, SC26-8016*
- *IMS/ESA Application Programming: Transaction Manager, SC26-8729*
- *IMS/ESA Customization Guide, SC26-8020*
- *IMS/ESA Installation Volume 1: Installation and Verification, SC26-8023*
- *IMS/ESA Installation Volume 2: System Definition and Tailoring, SC26-8024*
- *IMS/ESA Messages and Codes, SC26-8028*
- *IMS/ESA Operator's Reference, SC26-8030*
- *IMS/ESA Utilities Reference: System, SC26-8035*

ISPF

- *ISPF V4 Dialog Developer's Guide and Reference, SC34-4486*
- *ISPF V4 Messages and Codes, SC34-4450*
- *ISPF V4 Planning and Customizing, SC34-4443*
- *ISPF V4 User's Guide, SC34-4484*

Language Environment

- *Debug Tool User's Guide and Reference, SC09-2137*

National Language Support

- *National Language Support Reference Volume 2, SE09-8002*

NetView

- *NetView Installation and Administration Guide, SC31-8043*
- *NetView User's Guide, SC31-8056*

ODBC

- *Microsoft ODBC 3.0 Programmer's Reference and SDK Guide, Microsoft Press, ISBN 1-55615-658-8*

OS/390

- *OS/390 C/C++ Programming Guide, SC09-2362*
- *OS/390 C/C++ Run-Time Library Reference, SC28-1663*
- *OS/390 C/C++ User's Guide, SC09-2361*
- *OS/390 eNetwork Communications Server: IP Configuration, SC31-8513*
- *OS/390 Hardware Configuration Definition Planning, GC28-1750*
- *OS/390 Information Roadmap, GC28-1727*
- *OS/390 Introduction and Release Guide, GC28-1725*
- *OS/390 JES2 Initialization and Tuning Guide, SC28-1791*
- *OS/390 JES3 Initialization and Tuning Guide, SC28-1802*
- *OS/390 Language Environment for OS/390 & VM Concepts Guide, GC28-1945*
- *OS/390 Language Environment for OS/390 & VM Customization, SC28-1941*
- *OS/390 Language Environment for OS/390 & VM Debugging Guide, SC28-1942*
- *OS/390 Language Environment for OS/390 & VM Programming Guide, SC28-1939*
- *OS/390 Language Environment for OS/390 & VM Programming Reference, SC28-1940*
- *OS/390 MVS Diagnosis: Procedures, LY28-1082*
- *OS/390 MVS Diagnosis: Reference, SY28-1084*
- *OS/390 MVS Diagnosis: Tools and Service Aids, LY28-1085*
- *OS/390 MVS Initialization and Tuning Guide, SC28-1751*
- *OS/390 MVS Initialization and Tuning Reference, SC28-1752*
- *OS/390 MVS Installation Exits, SC28-1753*
- *OS/390 MVS JCL Reference, GC28-1757*
- *OS/390 MVS JCL User's Guide, GC28-1758*
- *OS/390 MVS Planning: Global Resource Serialization, GC28-1759*
- *OS/390 MVS Planning: Operations, GC28-1760*
- *OS/390 MVS Planning: Workload Management, GC28-1761*
- *OS/390 MVS Programming: Assembler Services Guide, GC28-1762*
- *OS/390 MVS Programming: Assembler Services Reference, GC28-1910*
- *OS/390 MVS Programming: Authorized Assembler Services Guide, GC28-1763*
- *OS/390 MVS Programming: Authorized Assembler Services Reference, Volumes 1-4, GC28-1764, GC28-1765, GC28-1766, GC28-1767*
- *OS/390 MVS Programming: Callable Services for High-Level Languages, GC28-1768*
- *OS/390 MVS Programming: Extended Addressability Guide, GC28-1769*

- *OS/390 MVS Programming: Sysplex Services Guide, GC28-1771*
- *OS/390 MVS Programming: Sysplex Services Reference, GC28-1772*
- *OS/390 MVS Programming: Workload Management Services, GC28-1773*
- *OS/390 MVS Routing and Descriptor Codes, GC28-1778*
- *OS/390 MVS Setting Up a Sysplex, GC28-1779*
- *OS/390 MVS System Codes, GC28-1780*
- *OS/390 MVS System Commands, GC28-1781*
- *OS/390 MVS System Messages Volume 1, GC28-1784*
- *OS/390 MVS System Messages Volume 2, GC28-1785*
- *OS/390 MVS System Messages Volume 3, GC28-1786*
- *OS/390 MVS System Messages Volume 4, GC28-1787*
- *OS/390 MVS System Messages Volume 5, GC28-1788*
- *OS/390 MVS Using the Subsystem Interface, SC28-1789*
- *OS/390 Security Server (RACF) Auditor's Guide, SC28-1916*
- *OS/390 Security Server (RACF) Command Language Reference, SC28-1919*
- *OS/390 Security Server (RACF) General User's Guide, SC28-1917*
- *OS/390 Security Server (RACF) Introduction, GC28-1912*
- *OS/390 Security Server (RACF) Macros and Interfaces, SK2T-6700 (OS/390 Collection Kit), SK27-2180 (OS/390 Security Server Information Package)*
- *OS/390 Security Server (RACF) Security Administrator's Guide, SC28-1915*
- *OS/390 Security Server (RACF) System Programmer's Guide, SC28-1913*
- *OS/390 SMP/E Reference, SC28-1806*
- *OS/390 SMP/E User's Guide, SC28-1740*
- *OS/390 RMF User's Guide, SC28-1949*
- *OS/390 TSO/E CLISTS, SC28-1973*
- *OS/390 TSO/E Command Reference, SC28-1969*
- *OS/390 TSO/E Customization, SC28-1965*
- *OS/390 TSO/E Messages, GC28-1978*
- *OS/390 TSO/E Programming Guide, SC28-1970*
- *OS/390 TSO/E Programming Services, SC28-1971*
- *OS/390 TSO/E User's Guide, SC28-1968*
- *OS/390 DCE Administration Guide, SC28-1584*
- *OS/390 DCE Introduction, GC28-1581*
- *OS/390 DCE Messages and Codes, SC28-1591*
- *OS/390 UNIX System Services Command Reference, SC28-1892*
- *OS/390 UNIX System Services Planning, SC28-1890*
- *OS/390 UNIX System Services User's Guide, SC28-1891*
- *OS/390 UNIX System Services Programming: Assembler Callable Services Reference, SC28-1899*

j

PL/I for MVS & VM

- *IBM PL/I MVS & VM Language Reference, SC26-3114*
- *IBM PL/I MVS & VM Programming Guide, SC26-3113*

OS PL/I

- *OS PL/I Programming Language Reference, SC26-4308*
- *OS PL/I Programming Guide, SC26-4307*

Prolog

- *IBM SAA AD/Cycle Prolog/MVS & VM Programmer's Guide, SH19-6892*

Remote Recovery Data Facility

- *Remote Recovery Data Facility Program Description and Operations, LY37-3710*

Storage Management

- *DFSMS/MVS Storage Management Library: Implementing System-Managed Storage, SC26-3123*
- *MVS/ESA Storage Management Library: Leading a Storage Administration Group, SC26-3126*
- *MVS/ESA Storage Management Library: Managing Data, SC26-3124*
- *MVS/ESA Storage Management Library: Managing Storage Groups, SC26-3125*
- *MVS Storage Management Library: Storage Management Subsystem Migration Planning Guide, SC26-4659*

System/370 and System/390

- *ESA/370 Principles of Operation, SA22-7200*
- *ESA/390 Principles of Operation, SA22-7201*
- *System/390 MVS Sysplex Hardware and Software Migration, GC28-1210*

System Network Architecture (SNA)

- *SNA Formats, GA27-3136*
- *SNA LU 6.2 Peer Protocols Reference, SC31-6808*
- *SNA Transaction Programmer's Reference Manual for LU Type 6.2, GC30-3084*
- *SNA/Management Services Alert Implementation Guide, GC31-6809*

TCP/IP

- *IBM TCP/IP for MVS: Customization & Administration Guide, SC31-7134*
- *IBM TCP/IP for MVS: Diagnosis Guide, LY43-0105*
- *IBM TCP/IP for MVS: Messages and Codes, SC31-7132*

- *IBM TCP/IP for MVS: Planning and Migration Guide, SC31-7189*

VS COBOL II

- *VS COBOL II Application Programming Guide for MVS and CMS, SC26-4045*
- *VS COBOL II Application Programming: Language Reference, GC26-4047*
- *VS COBOL II Installation and Customization for MVS, SC26-4048*

VS FORTRAN

- *VS FORTRAN Version 2: Language and Library Reference, SC26-4221*
- *VS FORTRAN Version 2: Programming Guide for CMS and MVS, SC26-4222*

VTAM

- *Planning for NetView, NCP, and VTAM, SC31-8063*
- *VTAM for MVS/ESA Diagnosis, LY43-0069*
- *VTAM for MVS/ESA Messages and Codes, SC31-6546*
- *VTAM for MVS/ESA Network Implementation Guide, SC31-6548*
- *VTAM for MVS/ESA Operation, SC31-6549*
- *VTAM for MVS/ESA Programming, SC31-6550*
- *VTAM for MVS/ESA Programming for LU 6.2, SC31-6551*
- *VTAM for MVS/ESA Resource Definition Reference, SC31-6552*

Index

How to send your comments

DB2 Universal Database for OS/390
Data Sharing:
Planning and Administration
Version 6

Publication No. SC26-9007-01

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 for OS/390 documentation. You can use any of the following methods to provide comments.

- Send your comments by e-mail to db2pubs@vnet.ibm.com and include the name of the product, the version number of the product 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, page number, or a help topic title).
- Send your comments from the Web. Visit the DB2 for OS/390 Web site at:

  http://www.ibm.com/software/db2os390

  The Web site has a feedback page that you can use to send comments.
- Complete the readers' comment form at the back of the book and return it by mail, by fax (800-426-7773 for the United States and Canada), or by giving it to an IBM representative.

Readers' Comments

DB2 Universal Database for OS/390
Data Sharing:
Planning and Administration
Version 6

Publication No. SC26-9007-01

How satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Technically accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□
Grammatically correct and consistent	□	□	□	□	□
Graphically well designed	□	□	□	□	□
Overall satisfaction	□	□	□	□	□

Please tell us how we can improve this book:

May we contact you to discuss your comments? □ Yes □ No

Name

Address

Company or Organization

Phone No.

Readers' Comments
SC26-9007-01

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department BWE/H3
PO Box 49023
San Jose, CA 95161-9945

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Program Number: 5645-DB2

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC26-9007-01