IBM DATABASE 2
System Planning and Administration Guide

Licensed Program

Order Number
SC26-4085-3

Program Number
5740-XYR

Release 3

Third Edition (May 1987)

This edition replaces and makes obsolete the previous edition, SC26-4085-2.

This edition applies to Release 3 of IBM DATABASE 2, Licensed Program 5740-XYR, and to any subsequent releases until otherwise indicated in new editions or technical newsletters.

The changes for this edition are summarized under "Summary of Changes" following the preface. Specific changes are indicated by a vertical bar to the left of the change. These bars will be deleted at any subsequent publication of the page affected. Editorial changes that have no technical significance are not noted.

Changes are made periodically to this publication; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370, 30xx, and 4300 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's program may be used. Any functionally equivalent program may be used instead.

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. If you request publications from the address given below, your order will be delayed because publications are not stocked there.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, P.O. Box 50020, Programming Publishing, San Jose, California, U.S.A. 95150. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Preface

About this Book

This book is written for users of IBM DATABASE 2 (DB2) who perform system planning and system administration tasks. System planning consists of planning to install DB2 or migrate from DB2 Release 2 to DB2 Release 3. System administration consists of managing DB2 resources after you install or migrate DB2. This book contains information about:

- Overall DB2 design
- Choosing DB2 system definition parameters
- Providing security for DB2
- Monitoring and tuning DB2

Throughout this book, the term *MVS* is used to represent both MVS/370 and MVS/Extended Architecture (MVS/XA). When it is necessary to make a technical distinction between the two environments, the specific terms are used. *CICS* is used to represent both CICS/OS/VS and CICS/MVS.

Prerequisite Knowledge

This book assumes that you are familiar with:

- Data base systems
- DB2 structure and function
- Structured Query Language (SQL) concepts

The following publications are prerequisite for using this book:

- *IBM DATABASE 2 General Information*
- *IBM DATABASE 2 SQL Learner's Guide*
- *IBM DATABASE 2 Data Base Planning and Administration Guide*, Chapters 1, 2, and 3

Use this book in conjunction with *IBM DATABASE 2 Install Guide*. Both books are organized to follow the sequence in which DB2 presents the panels to you. First read the introductory material in Section 1 of this book. As you go through the specific parameter information contained in Chapter 4, Install and Migration Parameters, turn to *IBM DATABASE 2 Install Guide* and record your choice for each parameter. Chapter 3 of that book, "Tailoring the Install and Migration Jobs," provides a summary table for each of the panels. This record of your parameter choices will greatly simplify your install or migration task. It will also be useful to you later for tuning purposes.

Organization of this Book

This book discusses two closely related tasks: system planning and system administration. This book has two main sections followed by a set of appendixes:

- Section 1: System Planning

  This section presents system planning concepts, lists hardware and software required to run DB2, provides information on estimating DB2 storage requirements and explains how to choose specific parameter values before you install or migrate to the current release of DB2.

- Section 2: System Administration

  This section discusses system administration—the tasks performed on an ongoing basis after installing or migrating DB2.

- Appendixes

  Appendix A, "DB2 Initialization Parameter Module (DSNZPARM)," explains the macros that contain the DB2 execution-time options you selected on the ISPF panels. This appendix appeared in the previous release of this manual as Appendix D.

  Appendix B, "RACF Examples for DB2," shows examples of accommodating DB2 started tasks with RACF, a RACF router table, and authorizing DB2 and DB2 users with RACF. This appendix appeared in an earlier release of *IBM DATABASE 2 Install Guide.*

  Appendix C, "Interpreting DB2 Trace Output." The information in this appendix appeared in the previous release of this manual in Chapter 7, "Monitoring DB2."

  Appendix D, "DB2 Trace Record Descriptions," details the contents of the records created by the DB2 trace facility. This appendix appeared in the previous release of this manual as Appendix E.

  Appendix E, "Accounting and Statistics Records," details the events identified by accounting and statistics trace records. This appendix appeared in the previous release of this manual as Appendix F.

Also included in this book are a bibliography, which lists the full title and order number of every IBM publication referred to in the text, a glossary, and an index.

The following appendixes, which appeared in the previous release of this manual, have been deleted:

"Install and Migration Parameter Planning Workbook": The information previously contained in this appendix has been incorporated into Chapter 4, Install and Migration Parameters.

"DB2 Catalog Structure": The information previously contained in this appendix supports the task of tuning, which is discussed in Chapter 8. A list of catalog and directory table spaces has been incorporated into Chapter 8.

"Running DB2 Utilities": This appendix has been deleted from this manual. It remains in

IBM DATABASE 2 Data Base Planning and Administration Guide
IBM DATABASE 2 Operation and Recovery Guide
IBM DATABASE 2 Command and Utility Reference

Summary of Changes

Release 3, June 1987

DB2 Release 3 offers you the following new function:

- New data types DATE, TIME, and TIMESTAMP, for quantities representing dates and times.
- The ability to do appropriate arithmetic and comparison operations on date/time data.
- New functions for extracting portions of date/time values, converting values from one data type to another, and extracting substrings of string values; and a new operator for concatenating string values.
- A new data type for single precision floating-point numbers.
- Enhancements to SQL to accommodate ANSI standards.
- Improved error handling for division by zero and certain other invalid arithmetic and conversion operations.
- Removal of certain restrictions on UNION and LIKE.
- Addition of UNION ALL to provide the capability of combining two tables without eliminating duplicates.
- Utility enhancements including the collection of data about the DB2 catalog, and the recovery of several indexes in the same table space with a single scan.
- The option to replace the contents of one partition of a table space rather than replacing the entire table space.
- Optimizer enhancements.
- Serviceability improvements to lessen service costs and problem resolution time.
- A new sample application program demonstrating use of the DB2 Call Attach Facility.
- Linear Data Set (LDS) support for user managed data sets.

For more information about the new function of DB2 Release 3, see *IBM DATABASE 2 General Information.*

The install process for Release 3 is similar to past releases of DB2, except that the size estimates produced by the install CLIST have been revised. Refinements have been made to the formulas the CLIST uses to estimate storage requirements for Release 3.

Date/time expressions can be written in one of four formats supplied with DB2 or in any user-defined format. The format for date/time expressions is specified when you install DB2 or migrate to DB2 Release 3. Examples of date/time expressions are provided with the Release 3 sample tables.

VS COBOL II (COB2) is now treated as a separate application programming default language (specified on install/migration panel DSNTIPF).

A new install parameter has been added to support automatic recall of migrated DB2 data sets using DFHSM (specified on install/migration panel DSNTIPO).

For users who already have DB2 Release 2 installed, migration and fall back between Release 2 and Release 3 are faster than they were between Release 1 and Release 2. Because the structure of the Release 3 DB2 catalog is the same as that for Release 2, you don't have to migrate user or system data for Release 3. Instead, you merely change existing JCL so that it references the new Release 3 code libraries instead of Release 2. In addition, rebinding of application plans is unnecessary for Release 3 migration.

Direct migration and fall back are not supported between DB2 Release 1 and Release 3. Release 1 users must migrate to Release 2 and then to Release 3.

Planning information for install and migration is contained in this manual; refer to *IBM DATABASE 2 Install Guide* for step-by-step install, migration, and fall back procedures.

Release 2, March 1986

New or enhanced capabilities available to DB2 Release 2 users include:

- The DB2 catalog has been restructured to enhance concurrency of catalog-based operations.
- Performance improvements due to internal and external changes made for Release 2.
- An enhanced instrumentation and tracing function that allows users to gather accounting, statistics, performance, and serviceability trace data.
- Support for double-byte character data, including field procedures that allow exits on a field basis.
- Enhancements and extensions to DB2 utilities, such as:
  LOAD REPLACE option
  Restartable LOAD without logging
  Extensions to RECOVER to give more flexibility and function in performing the recovery task
- A new statement, EXPLAIN, which provides data base design implications to aid in performance evaluation.
- Numerous operational enhancements such as the use of nonnull default values.
- Improvements to DB2 locking, such as lock escalation and options for choosing lock durations.
- Improvements to DB2 Interactive (DB2I), including a new defaults panel and new, easier to use program preparation panels.

Contents

Section 1: System Planning

You should read *Section 1* before attempting to install DB2 or migrate to DB2 Release 3. This section consists of the following chapters:

Read these chapters before you begin the actual install or migration process. When you complete these chapters, turn to *IBM DATABASE 2 Install Guide* and record your choices for specific parameter values. Then you will be ready to follow the step-by-step procedures in *IBM DATABASE 2 Install Guide*. This record of your parameter choices will greatly simplify the task of installing DB2 and will be useful to you for tuning and migration.

Chapter 1. System Planning Concepts

This chapter explains the DB2 concepts that relate to system planning. It consists of three sections:

- "Product Planning" presents list of hardware and software that DB2 requires
- "System Planning and DB2 Structure" presents brief discussion of DB2 structure as it relates to system planning
- "DB2 and the MVS Environment" presents a discussion of how DB2 operates with related IBM products

You should be familiar with DB2 before reading this chapter.

Product Planning

Hardware Requirements

DB2 will operate on any processor supported by MVS/System Product Version 1 Release 3.6 or MVS/System Product Version 2 Release 1.3. For the IBM 3033 Processor, it is recommended that hardware Cross-Memory Extension Feature (6850) also be used. The processors must have sufficient real storage to satisfy the combined requirements of DB2, of MVS/XA or MVS/370, of the appropriate Data Facility Product, of batch requirements, and of other customer-required applications.

The configuration must include sufficient I/O devices to support the requirements for system output, system residence, and system data sets. Sufficient direct access storage (DASD) must be available to satisfy the user's information storage requirements and may consist of any direct access facility supported by the system configuration and the programming system.

External Storage: Any DASD or tape device supported by Data Facility Product (DFP) may be used for the DB2 data sets. The following DB2 data sets are supported by the following device types:

- Active recovery log data sets: DASD
- Archive recovery log data sets: DASD, tape, MSS
- Image copy data sets: DASD, tape, MSS
- Bootstrap data set: DASD
- Data base data sets: DASD, MSS
- DB2 catalog data sets: DASD
- Work data sets (for utilities): DASD, tape, MSS

If these data sets are on DASD that is shared by multiple MVS systems, global resource serialization should be used in order to prevent concurrent access by more than one MVS system.

The minimum DASD space requirement (based on installing DB2 using the panel default values) is approximately 261 megabytes, in addition to space for user data. For information on estimating the storage that DB2 needs, see Chapter 3, "Estimating DB2 Storage Needs" on page 36.

If you use tape as your log archiving device, at least two tape drives are required.

Data Communications Devices: DB2 uses the system console. You can control DB2 operations from:

- The system console
- Authorized IMS/VS terminals
- Authorized CICS terminals
- TSO terminals by authorized users

For the data communication devices supported by IMS/VS, CICS, and TSO, see the appropriate documentation.

Software Requirements

This section lists program products that are required in the DB2 environment and those that can be used optionally. Minimum versions and releases of required products are stated for required products. In general, you can use follow-on versions or releases of these products, unless the description for a given product specifically states otherwise. For the minimum version or release of optional products, see the DB2 Release 3 announcement letter or program directory.

Code: The language for the few modules distributed as source code is Assembler. Sample programs are provided in COBOL, FORTRAN, PL/I, and Assembler. Some functions are provided by TSO CLISTs and Interactive System Productivity Facility (ISPF) panels.

Operating System and Support Programs:: DB2 requires the following licensed programs (or their equivalents):

- For an MVS/XA environment:

  MVS/SP-JES2 or JES3 Version 2 Release 1.3

  MVS/XA Data Facility Product Version 2 Release 1

  TSO Extensions (TSO/E) Release 2.1

- For an MVS/370 environment:

  MVS/System Product-JES2 or JES3 Version 1 Release 3.6

  MVS/370 Data Facility Product Release 1.1

  MVS/370 TSO Extensions (TSO/E) Release 2

- For both MVS/XA and MVS/370 environments:

  DFSORT Release 8

  System Modification Program (SMP) Release 4 or System Modification Program/Extended (SMP/E) Release 3

  To use DB2 Interactive (DB2I) services and interactive install: Interactive System Productivity Facility (ISPF) Version 2 Release 2 and ISPF/Program Development Facility (ISPF/PDF) Version 2 Release 2

Optional programs (or their equivalents):

- APL2
- Application Prototype Environment
- Assembler H
- Application System (AS)
- Cross System Product/Application Development (CSP/AD)
- Cross System Product/Application Execution (CSP/AE)
- Customer Information Control System (CICS)
  Note: CICS/OS/VS 1.7 is the lowest release supported by DB2 Release 3.
- Data Base Edit Facility (DBEDIT)
- Data Base Migration Aid Utility
- Data Base Relational Application Directory (DBRAD/MVS)
- Data Extract (DXT)
- Data Facility Hierarchical Storage Manager (DFHSM)
  Note: To use DFHSM with DB2 Release 3 data bases, you must be running in an MVS/XA environment.
- Host Data Base View (HDBV)
- IBM BASIC/MVS
- IBM DATABASE 2 Performance Monitor (DB2PM)
- IBM TSO/E Servers/Requesters
- IMS Batch Terminal Simulator (BTS)
- Info Center/1 (IC/1)
- Information Management System/Virtual Storage (IMS/VS)
- OS PL/I Optimizing Compiler and Libraries
- OS/VS COBOL Compiler and Library
- Resource Access Control Facility (RACF)
- System Level Reporter (SLR)
- Query Management Facility (QMF)
- The Information Facility (TIF)
- VS COBOL II
- VS FORTRAN

System Planning and DB2 Structure

You must make certain decisions before installing or migrating DB2.

- *Installing* involves preparing a DB2 Release 3 subsystem to operate as an MVS subsystem. This information is intended primarily for those who are not DB2 Release 2 users. Release 2 users, however, may choose to install a Release 3 test subsystem before migrating their Release 2 production subsystem.
- *Migrating* involves moving to Release 3 without losing the data and applications you created on Release 2. You can migrate only if you already have Release 2 installed.

During either process, you specify more than 100 parameter values that determine DB2 characteristics, including DB2 library names, storage sizes, security mechanisms, and recovery options. This book explains how to plan the values you want for these parameters.

If you have Interactive System Productivity Facility (ISPF) and Interactive System Productivity Facility/Program Development Facility (ISPF/PDF), you can use a series of ISPF panels to specify parameter values. The panels pass these values to a CLIST, which tailors a series of jobs to help you install and migrate DB2. If you do not have both ISPF and ISPF/PDF, you must run the CLIST directly.

The planning activities for installing and migrating DB2 are similar. Migration planning has some added considerations to account for differences between DB2 Release 2 and DB2 Release 3.

Whether you are installing or migrating, you must plan for virtual and external storage required for the DB2 subsystem data sets.

The Structure of DB2

DB2 consists of two major functional areas: *data base services* and *system services*. Each area is a collection of resource manager subcomponents.

Data base services consists of functions directly involved in the management of user data. Most SQL statements and many commands are processed here.

System services consists of functions only indirectly involved in the management of user data. These functions are primarily internal to DB2. System services provides the environment for data base services and contains service-oriented functions, such as virtual storage management, statistics and accounting data collection, and DB2 command processing support. It also contains coordination-oriented functions that require communication among several DB2 subcomponents, such as checkpoint, startup and shutdown, allocation and deallocation, commit and abort, and recovery. The attachment-oriented functions are also a part of system services.

Address Spaces

Several address spaces are involved in DB2 operation. The two major functional areas of DB2, data base services and system services, operate in separate address spaces. Additionally, the IMS/VS Resource Lock Manager (IRLM) which controls DB2 locking, operates in its own address space, and DB2 users operate in their own address spaces. If IMS/VS is operating, there are also separate IMS/VS address spaces.

Figure 1 on page 6 shows how the address spaces relate to TSO, IMS/VS, and CICS user address spaces. Each user address space communicates with data base services and system services address spaces. Additionally, the IMS/VS user address space communicates with the IRLM and IMS/VS address spaces. The system services, data base services, CICS, IRLM, and IMS/VS address spaces communicate with each other as the arrows indicate.

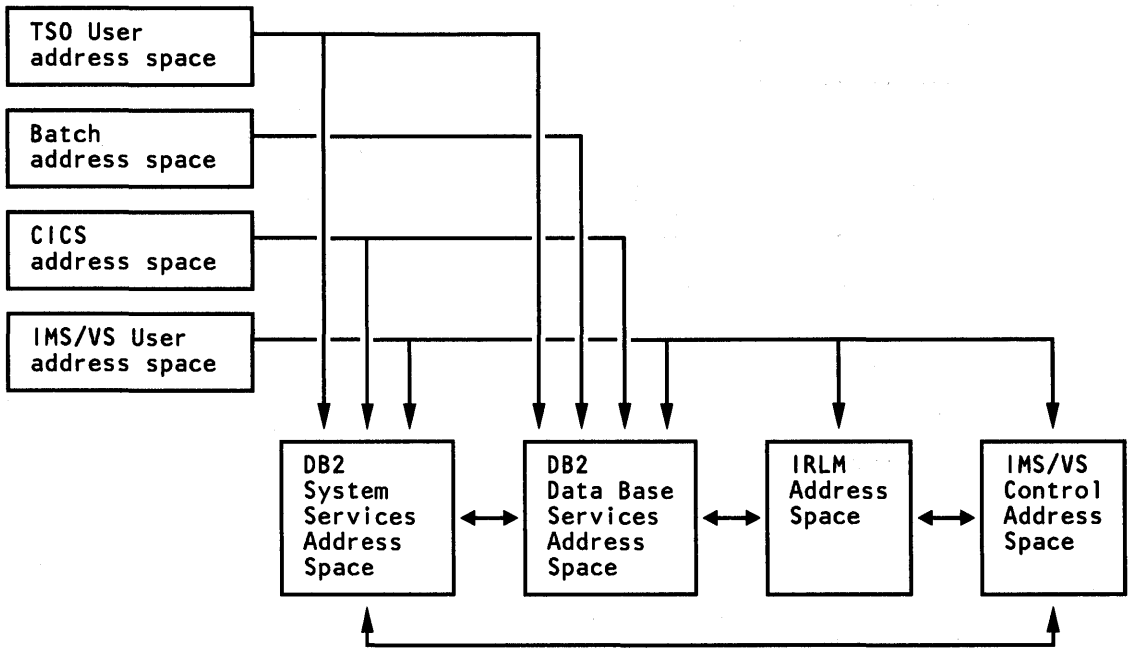


Figure 1. Relationship Between DB2 Users and DB2-Related Address Spaces

DB2 Objects

As system administrator, you will monitor various DB2 objects to achieve optimal system performance. Figure 2 on page 7 shows how the objects relate to each other.

Tables

DB2 data is presented to users in tables. A *table* is a collection of rows that all have the same columns. When you create a table, you're creating an ordered set of columns. At the intersection of a column and row is a *value*. The storage representation of a row is called a *record*.

All the data in a particular column must be the same data type. DB2 data types are described in *IBM DATABASE 2 Data Base Planning and Administration Guide*.

Table Spaces

Table spaces are created to hold tables. When you create a table space, you can specify to which data base it belongs and which storage group it uses. Each table space is divided into equal units, called *pages*. A table space consists of 1 to 64 VSAM data sets and can contain up to 64 gigabytes of information. These VSAM data sets may be either entry-sequenced data sets (ESDS) or linear data sets (LDS). The LDS type is only valid for user-defined table spaces.

Table spaces may be divided into smaller physical units called *partitions*. (See Figure 3 on page 7.) Partitions can be assigned independently to different storage groups.

Partitioned table spaces divide available space into a number of units of storage. A partitioned table space can contain only one table. Partitioning enables you to put different parts of your table on different device types; thus, frequently accessed data can be placed on faster devices. Partitioning is also an advantage when you want to

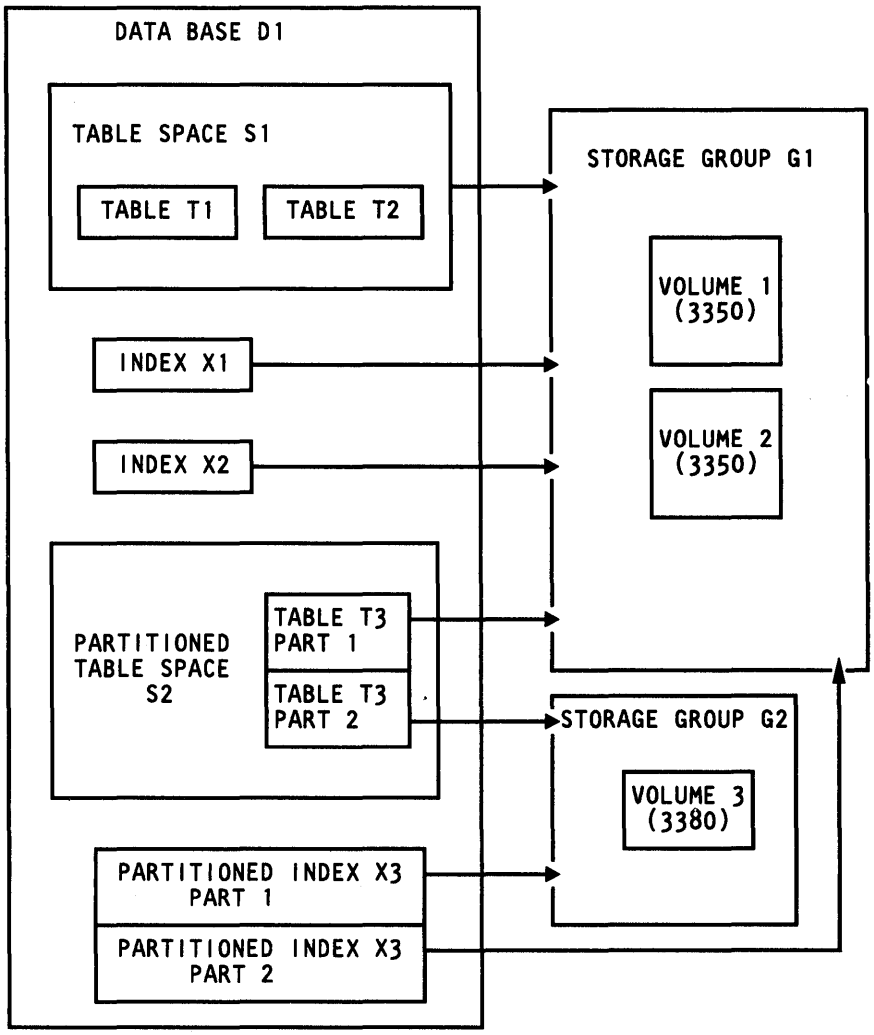


Figure 2. DB2 Objects

reorganize or recover a table space, speeding recovery and reorganization by dealing with smaller amounts of data. You can reorganize or recover a partition instead of a whole table space.

Simple table spaces are table spaces that are not partitioned. A simple table space can contain more than one table.

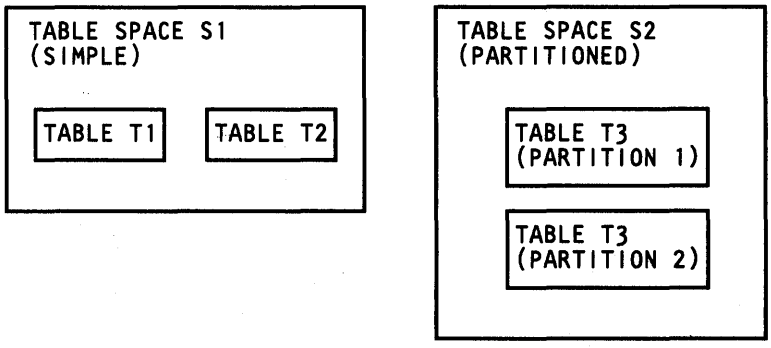


Figure 3. Tables in Table Spaces

Indexes

An *index* is an ordered set of pointers to the data in a DB2 table. Each index is based on the values of data in one or more table columns. An index is an object that's separate from the data in the table. When you request an index, DB2 builds this structure and automatically maintains it.

Indexes help:

- Improve performance. In most cases, access to data is faster with an index.
- Ensure uniqueness. A table with a unique index cannot have rows that are duplicates with respect to values of the key. (A key is a column or an ordered collection of columns on which an index is created.)
- Enable you to use partitioned table spaces. A special index, called a clustering index, controls the distribution of the data across partitions.

An index on a table can be created any time after the table is created. Except for changes in performance, users of the table are unaware of the existence of the index. DB2 will determine whether or not to use the index to access the table.

A *clustering index* is an index that causes the records to be stored in a physical order that approximates the order of the key. Thus, when DB2 needs to get data that follows the order of the clustering key, it can do so faster than it could if the data were not arranged in a clustering order. A table can have only one clustering index.

Each index is stored in its own *index space*. An index space consists of 1 to 64 VSAM data sets. These VSAM data sets may be either entry-sequenced data sets (ESDS) or linear data sets (LDS). The LDS type is only valid for user-defined index spaces. When you create an index, an index space is automatically created in the same data base as the table on which the index is defined.

Views

A *view* is a subset of data from one or more tables. Views can also be subsets of other views. Like tables, views have rows and an ordered set of columns. Unlike tables, a view does not have a storage representation. When a view is created, its definition is stored in the DB2 catalog. No data is stored; therefore no index can be created for a view. However, an index created for a table on which a view is based may improve the performance of operations on the view.

Some basic reasons for the use of views are:

- They enable you to tailor a table for particular users. Certain tables may have many columns, not all of which are of interest to certain users. A view can include only the columns or rows of interest.
- They can be used for data security. Views enable users to access only the data they are authorized to see.

Storage Groups

A *storage group* is a set of DASD volumes used to hold the data sets required for table spaces and indexes. The volumes of a storage group must be of the same device type. The description of a storage group includes its name, its volumes, and the Integrated Catalog Facility (ICF) catalog used to keep track of the data sets.

Data Bases

A DB2 *data base* is a collection of tables, indexes, table spaces, and index spaces. The data base serves as:

- A unit of START/STOP
- A unit of authorization (although smaller units also serve this purpose)
- A unit of clustering for separation

DB2 Catalog

The DB2 *catalog* contains information about every object in a DB2 subsystem. It consists of a set of DB2 tables stored in the DSNDB06 data base.

The DB2 catalog is not the same as the ICF catalog. The ICF catalog contains information about VSAM data sets; the DB2 catalog contains information about DB2 objects. The DB2 catalog is itself a DB2 data base consisting of tables and indexes, and is contained in VSAM data sets.

To give you some idea of the information recorded in the DB2 catalog:

- When a CREATE TABLE statement is issued, a row is inserted into the catalog table SYSTABLES, which keeps track of all the tables and views in the DB2 subsystem. The entry in this catalog table contains such information as the name of the table, the name of the person who created it, the object type (table or view), the name of the data base containing the object, and the name of the table space containing the object. A row is also inserted into catalog table SYSTABAUTH to note that the creator of the table has all privileges on the table, with grant option.
- For each column in a newly created table, a row is inserted into the catalog table SYSCOLUMNS. Each row has such information as the name of the table it belongs to, the width of the column, its data type, and the sequence number of the column in the table.
- One column in catalog table SYSTABLESPACE shows the number of tables that are in the table space occupied by the new table, and the value of this column is increased by 1 each time a new table is created in the table space.

Because SQL data definition statements update the DB2 catalog, they require exclusive locks on changed pages. The catalog structure is designed to minimize contention among users by clustering related records together on a given data page. Likewise, related index key values are clustered together on a given index page. For example, the catalog table space SYSDBASE, which contains the descriptions of all constituent objects of each data base, will be organized such that records pertaining to different data bases are not stored on the same data page. Further, indexes on this table space, clustered according to CREATOR or GRANTOR/GRANTEE, will be organized such that entries for different users (that is, for different authorization IDs) are not included on the same index page. Contention among users for access to the catalog is therefore reduced, provided each user has a unique authorization ID and a private data base for data that is not intended to be shared.

DB2 Directory

The DB2 directory contains information required to start DB2 and is used by DB2 during its normal operation. SQL cannot be used to access the directory. The directory is not described in the DB2 catalog.

The directory consists of a set of DB2 tables stored in four table spaces in the DSNDB01 data base. Each table space is supported by a VSAM entry-sequenced data set. They are listed below:

- DBD01 is the data base descriptor (DBD) table space
- SCT02 is the skeleton cursor table (SCT) table space
- SYSLGRNG is the log range table space
- SYSUTIL is the system utilities table space

The directory also contains DSNSCT02, an index space for SCT02.

The name for each directory data set is in the form:

dddddddd.DSNDB*n*.DSNDB01.*xxxxxxxx*.I0001.A001

where:

dddddddd is the high-level qualifier specified during the install or migration process. This is the value you specify for the CATALOG ALIAS parameter (VCATALOG) on the Data Parameters Panel (DSNTIPA2). The default is DSNC130.

n is C for a cluster name and D for a data set name.

xxxxxxxx is the table space or index space name: DBD01, SCT02, SYSLGRNG, SYSUTIL, or DSNSCT02. They are described below.

Data Base Descriptor

The data base descriptor table space (DBD01) contains the data base descriptors (DBDs). DBDs are the internal control blocks that describe the data bases existing within DB2. Each data base has exactly one corresponding DBD that describes its data base, table spaces, tables, and indexes. DBDs also contain other information that DB2 uses to access tables in that data base. DB2 creates and updates DBDs whenever their corresponding data bases are created or updated. Figure 4 on page 11 illustrates the contents of DBD01.

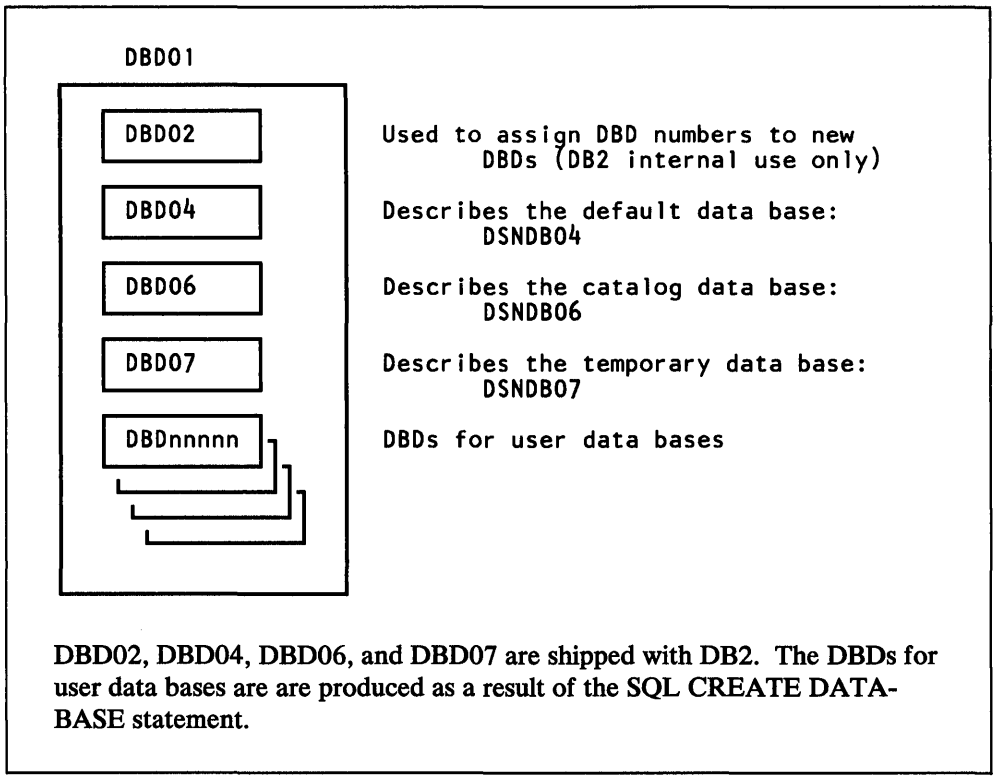


Figure 4. Contents of the Data Base Descriptor Table Space (DBD01)

Skeleton Cursor Table

A skeleton cursor table contains information that associates applications using SQL statements with the DB2 data bases they require for execution. Each time an application is bound, DB2 creates a skeleton cursor table and writes it in SCT02. The index on SCT02 is DSNSCT02.

Log Range

DB2 writes a record in the log range table space (SYSLGRNG) every time a table space is opened and updated, and updates SYSLGRNG whenever that table space is closed. The record contains the opening and/or closing log RBA (relative byte address) for the table space. The log RBA is the relative byte address in the log data set where open and close information about the table space is contained.

The use of SYSLGRNG speeds up table space recovery by limiting the log information that must be scanned to apply data base changes from the log to a table space that is being restored.

System Utilities

This data set contains information needed to control utility startup and termination processing, along with information needed to restart a utility after a subsystem failure.

DB2 writes a record in the system utilities table space (SYSUTIL) for every utility job that is run. The record remains in SYSUTIL while the utility is executing, and also remains there for any utility that has stopped executing before it has completed its full processing. If the utility terminates without completing its full processing, DB2 uses the record in SYSUTIL to restart the utility.

Active and Archive Logs

DB2 records all data changes and significant events as they occur. For each data change or significant event, DB2 creates a log record. In the case of failure, you can use this data to help you recover.

DB2 writes each log record to a DASD data set called the *active log*. When the active log is full, DB2 copies the contents of the active log to a DASD, MSS, or magnetic tape data set called the *archive log*.

The active log consists of from 2 to 53 active log data sets, each of which is a single-volume, single-extent VSAM entry-sequenced data set (ESDS). When you define an ESDS for an active log data set, the data set is associated with an ICF catalog.

Within the active log data set, a VSAM control interval (CI) is one VSAM record. DB2 manages the individual log records within the VSAM record. The CI size is 4K bytes. An active log data set can be processed by VSAM record management and access method services. However, if you read the active log using VSAM record management, you must interpret the log control information at the beginning of each log record and log record segment, because each CI is one VSAM record, but not necessarily one log record.

Active log data sets are not reused until all other active log data sets have been filled and all the log information has been archived. Until they are reused, their data (on DASD) can be used to satisfy log read requests. If log data is needed from an active log data set that has already been reused, log data is retrieved from the archive log data set to which it was copied.

The archive log consists of up to 1000 data sets, each of which is a SAM data set (physical sequential) that resides on a magnetic tape, MSS, or DASD volume. An archive log data set is created during the log off-load process (when an active log data set is copied to an archive log data set). It can be cataloged in an ICF catalog, and can be protected with an MVS data set password or with RACF.

Each record in the archive log data set is a control interval (CI) of data from an active log data set. If the archive log data set resides on magnetic tape, the tape must have IBM standard labels.

The log off-load function copies data from the current active log data set to an archive log data set. This is done automatically when an active log data set is filled. New archive log data sets are automatically created and allocated by DB2, using sizes and devices that you specify on the Archive Log Data Sets Panel (DSNTIPA). When these data sets are allocated, they are allocated with the release option (RLSE) to avoid unnecessary use of DASD space.

During log off-load, DB2 updates the BSDS to record data about the new archive log data set. The log off-load function also updates the BSDS to indicate that the copied active log data set can be reused for new log data.

During log off-load, DB2 also creates a BSDS backup copy. Each time a new archive log data set is created, a backup copy of the BSDS is created. The data set name for the BSDS backup copy is the same as the archive log data set name, except that *A* is changed to *B*. For example, if the data set name of the archive log is DSNARLC1.*A*000287, the data set name of the BSDS backup copy will be DSNARLC1.*B*000287. If the off-load device type is tape, the backup copy of the BSDS is the first data set on the tape and the archive log data set is the second data

set on the tape. If the off-load device type is DASD, space for the backup BSDS copy is separately allocated on the same volume as the archive log data set.

Bootstrap Data Set (BSDS)

The *bootstrap data set (BSDS)* lists all log data sets known to DB2. DB2 uses the BSDS to keep track of the active and archive log data sets and DB2 checkpoint activity. DB2 uses records in the BSDS during recovery and DB2 restart to allocate active and archive log data sets.

The log inventory contains a list of all log data sets, both active and archive, and the RBA range contained in each. For the active log, it also indicates which are full and which are available for reuse. DB2 records data about the log data set in the BSDS each time a new archive log data set is defined or an active log data set is reused.

When DB2 terminates processing (either normally or abnormally), it saves information about its status in the BSDS for use during DB2 restart.

When the number of archive log data sets reaches the specified limit, DB2 deletes the record of the oldest archive data set from the BSDS. For example, if 20 were specified as the maximum number of archive log data sets for your DB2 subsystem, and the log off-load function is about to archive an active log for the twenty-first time, DB2 stores information about the newly archived data into the BSDS and deletes the BSDS entry for the first archive log data set.

When an archive log data set is needed, DB2 uses information in the BSDS to allocate it dynamically. The information needed to allocate active log data sets when DB2 is started is also contained in the BSDS.

Archive log data sets may be cataloged. If an archive log data set is not cataloged, the BSDS contains the data set name, volume serial, and device type. If the data set is cataloged, DB2 sets a flag in the BSDS indicating that the data set is cataloged and that subsequent allocations are to be made using the catalog.

Because the BSDS is essential to recovery in the event of subsystem failure, DB2 automatically creates two copies of the BSDS and, if space permits, places them on separate volumes.

Default Data Base

If you create a table space or a table and don't specify a data base, the table or table space is created in the default data base. The default data base is a predefined user data base. Its name is DSNDB04; its default buffer pool is BP0; and its default storage group is SYSDEFLT.

Storage group SYSDEFLT is created during the install process. After you complete an install, all users (PUBLIC) have the authority to create table spaces or tables in data base DSNDB04. The system administrator (SYSADM) can revoke these privileges and grant them only to particular users as necessary.

For a migration, Release 3 adopts the default data base and default storage group you used for Release 2. Users have the same authority for Release 3 as they did for Release 2.

Temporary Data Base

The *temporary data base* consists of table spaces that store temporary data generated during the execution of certain SQL statements. Many different types of SQL statements may need working storage that is provided by temporary table spaces.

Additional temporary table spaces can be created during or after install or migration. This improves DB2 performance by reducing device contention among applications that require working storage. In addition, you can concatenate temporary table spaces. This allows you to support large temporary files.

Utility Work Data Sets

Several DB2 utilities need work data sets to function. These data sets:

- Unload a table space during table space reorganization
- Hold index entries during index creation, reorganization, and initial load
- Contain input statements for a utility
- Hold an image copy of a table space
- Provide working storage for sorting

These data sets can be on any tape or DASD device supported by BSAM. This device type is also used for the precompiler, compiler, and linkage editor.

The utility work data sets are allocated by DD statements. Within each utility, each required data set has a default DDNAME (for example, SYSCOPY for the image copy data sets). The utility's invoker can override the default DDNAME by specifying the desired DDNAME on the utility command.

Sorting is usually required for index creation, reorganization, and initial load.

Utility work data sets can be either temporary or permanent. Temporary work data sets are used only during the life of the utility job; permanent work data sets exist after the execution of the utility job which created them. Generally, permanent work data sets deal with backup and recovery, and are identified in the SYSCOPY catalog table.

DD statements are required for all temporary work data sets and for the creation of permanent work data sets. However, when a permanent work data set must be read (for example, when reading an image copy during media recovery) the data set is allocated dynamically, and no DD statement is required.

Buffer Pools

Buffer pools are areas of virtual storage used to temporarily store pages of table spaces or indexes. When an application needs to access a row of a table, DB2 retrieves the page containing that row and places the page into a buffer. If the row is changed, the buffer must be written back to the table space.

If the needed data is already in a buffer, the program will not have to wait for it to be retrieved. The result is quicker performance.

When you install DB2, you can specify the numbers of buffers for four different buffer pools. The 32K-byte pool will have 32K-byte buffers, and the three 4K-byte pools will each have 4K-byte buffers.

The number of buffers within each pool always falls between a minimum and a maximum value you specify. Typically, the minimum value will be used. In unusual situations requiring the expansion of the buffer pool, the number of buffers may reach the maximum value. However, the buffer manager will try to return to the minimum value.

Locks

DB2 allows more than one program to access the same data at essentially the same time; this is known as *concurrency*. To control concurrency and prevent lost updates, DB2 uses *locks*.

Locks prevent one program from accessing data that another program has changed but not yet committed. DB2 acquires all locks implicitly, under DB2 control. It is never necessary for an application to request a lock explicitly, and it is never possible for a program to fail to request a lock it needs.

DB2 and the MVS Environment

DB2 operates as a formal subsystem of MVS.

Applications that access DB2 resources can run in the TSO (foreground or batch), IMS/VS, or CICS environments. IBM provides attachment facilities to connect DB2 to each of these environments. DB2 utilities run in the batch environment.

DB2 and MVS

As a formal subsystem of MVS, DB2 uses:

- The MVS subsystem interface (SSI) protocols
- Multiple address spaces
- Key 7 operation and storage
- Synchronous cross memory services for address space switching
- System Management Facilities (SMF) for statistics, accounting information and performance data
- These reliability and serviceability features:
  - Functional recovery routines (FRR)
  - ESTAE recovery routines
  - SYS1.LOGREC
  - SYS1.DUMP

You can enter all DB2 commands from an authorized MVS console by using the DB2 subsystem recognition character (SRC) as the first character of the command. The default is the hyphen (-); you can change this when you install or migrate DB2.

DB2 and RACF

Resource Access Control Facility (RACF) can be used to control access to your MVS system. When users begin sessions with TSO, IMS/VS, or CICS, their identities are checked to prevent unauthorized access to the system.

We recommend RACF be used to security-check DB2 users and to protect DB2 resources. RACF provides effective protection for DB2 data by permitting only DB2-mediated access to DB2 data sets.

DB2 and ISPF

DB2 provides many ISPF panels that allow you to perform most DB2 tasks interactively. These panels comprise a DB2 facility called DB2 Interactive, or DB2I.

You can use DB2I panels to:

- Run Structured Query Language (SQL) statements
- Prepare and execute applications
- Issue DB2 commands
- Run DB2 utility jobs
- Display online HELP information for DB2 functions
- BIND, REBIND and FREE plans
- Generate table declarations for inclusion in programs

The DB2I panels that allow you to submit and test SQL statements are called *SPUFI* (SQL Processor Using File Input). Application programmers, for instance, can use SPUFI to create, edit, and test SQL statements before including them in a program. This reduces testing time, placing programs into production more quickly. System administrators can use SPUFI to create and query DB2 objects and to grant and revoke authorization.

The DB2I panels that display online information about DB2 functions are called *online HELP panels*. These panels provide examples and syntax guidelines for DB2 statements and commands. HELP panels exist for SQL statements and return codes, DB2 utilities, DB2 Interactive (DB2I), the DB2 catalog, and DB2 commands and subcommands.

For more information on using DB2 and ISPF in an application, see *IBM DATABASE 2 Advanced Application Programming Guide*.

DB2 and TSO

The TSO attachment facility is required for several online functions that are provided with DB2 and to bind application plans.

Using the TSO attachment facility, TSO users can access DB2 by running in either foreground or batch. You gain foreground access through a TSO terminal; you gain batch access by invoking the TSO Terminal Monitor Program (TMP) from an MVS batch job.

Regardless of how you access DB2 (foreground or batch), the DSN command processor makes access easier. It allows appropriately authorized DB2 users or jobs to create, modify, and maintain data bases and programs.

You invoke the DSN command processor from the foreground by issuing a command at a TSO terminal. From batch, first invoke TMP from within an MVS batch job, then pass TMP the commands you would have issued at a TSO terminal in the SYSTSPRT data set.

After DSN is running, you can issue DB2 commands or DSN subcommands. You cannot, however, issue a -START DB2 command from within DSN. If DB2 is not running, DSN cannot establish a connection to it; a connection is required so that DSN can transfer commands to DB2 for processing.

Figure 5 below shows the relationship between DB2 and TSO.

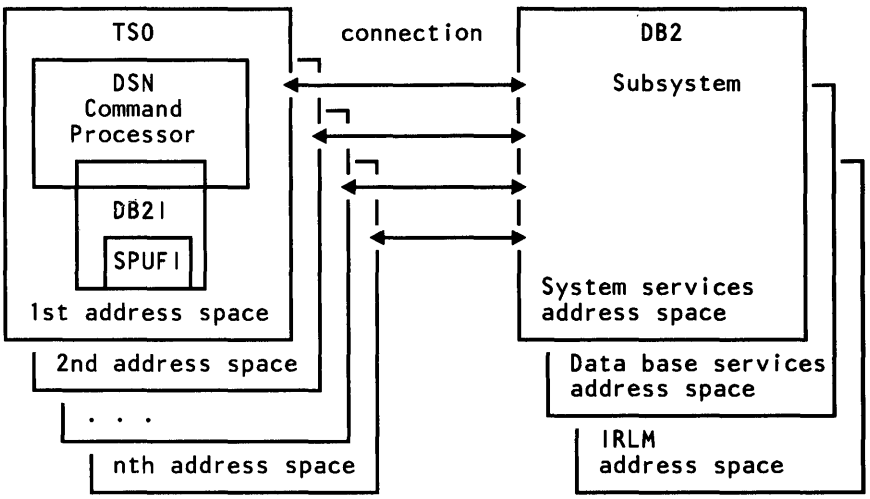


Figure 5. The Relationship of DB2 to TSO

Call Attachment Facility

Most TSO applications should use the TSO attachment facility, which invokes the DSN command processor. Together DSN and TSO provide services such as automatic connection to DB2, attention key support, and translation of return codes into error messages. However, when using DSN services, your application must run under the control of DSN. For some applications, this might be a limitation.

The call attachment facility (CAF) provides an alternate connection for TSO and batch applications needing tight control over the session environment. Applications using CAF can *explicitly* control the state of their connections to DB2 by using connection functions supplied by CAF. For more information on CAF, see "Call Attachment Facility" on page 153, and *IBM DATABASE 2 Advanced Application Programming Guide.*

DB2 and CICS

The CICS attachment facility provided with DB2 allows you to access DB2 from CICS. After DB2 has started, it can be operated from a CICS terminal. You can start and stop CICS and DB2 independently, and establish and terminate the connection between them at any time. Optionally, you can have CICS connect to DB2 automatically.

The CICS attachment facility also provides CICS applications access to DB2 data while operating in the CICS environment. CICS applications, therefore, can access both DB2 data and CICS data. In case of system failure, CICS coordinates recovery of DB2 and CICS data.

CICS batch support for SQL is not available.

Application Programming with CICS

Programmers writing CICS command-level programs can use the same data communication coding techniques to write the data communication portions of programs that access DB2 data. Only the data base portion of the programming changes. For the data base portions, they use SQL statements to retrieve or modify data in DB2 tables.

To the CICS terminal user, programs that access both CICS and DB2 data appear identical to programs that access only CICS data.

DB2 supports this cross-product programming environment by coordinating recovery resources with those of CICS. CICS applications can therefore access CICS controlled resources as well as DB2 data bases.

Function shipping of SQL requests is not supported. In a CICS multi-region operation (MRO) environment, each CICS address space may have its own attachment to the DB2 subsystem.

System Administration and Operation with CICS

An authorized CICS terminal operator can issue DB2 commands to control and monitor both the attachment facility and DB2 itself. Authorized terminal operators can also start and stop DB2 data bases.

Even though you perform DB2 functions through CICS, you need to have the TSO attachment facility and ISPF to take advantage of the online functions supplied with DB2 for install and online help. You also need the TSO attachment to bind application plans.

Figure 6 shows the relationship between DB2 and CICS. Connection threads are explained in Chapter 6, "DB2 Attachment Facilities" on page 150.

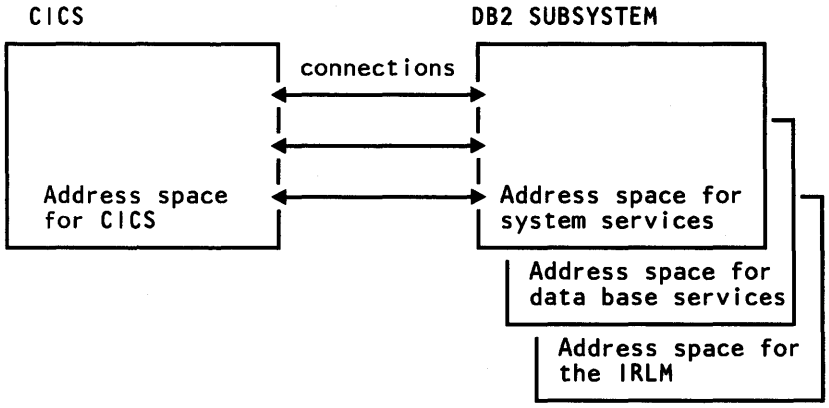


Figure 6. The Relationship of DB2 to CICS

DB2 and IMS/VS

The IMS/VS attachment facility allows you to access DB2 from IMS/VS. The IMS/VS attachment facility receives and interprets requests for access to DB2 data bases using IMS/VS subsystem-provided exits. Usually, IMS/VS connects to DB2 automatically with no operator intervention.

In addition to DL/I (Data Language I) and Fast Path calls, IMS/VS applications can make calls to DB2 using embedded SQL statements. In case of system failure, IMS/VS coordinates recovery of DB2 and IMS/VS data.

DB2 can be used in an Extended Recovery Facility (XRF) environment to facilitate recovery from an IMS/VS failure. To accomplish this, you must place all DB2 data sets on DASD shared between the primary and alternate XRF processors. This will enable DB2 to be manually stopped on the primary processor and started on the alternate. To preserve data integrity in case of an operator error, global resource serialization must be active, and the primary and alternate XRF processors must be included in the global resource serialization ring. For more information about XRF, see *IBM DATABASE 2 Operation and Recovery Guide* and *IMS/VS Version 2 Operations and Recovery Guide*.

Application Programming with IMS/VS

With the IMS/VS attachment facility, DB2 provides data base services for IMS/VS dependent regions. IMS/VS batch applications cannot access DB2 data.

IMS/VS programmers writing the data communication portion of programs do not need to alter their coding technique to write the data communication portion when accessing DB2; only the data base portions of the programs change. For the data base portions, they code SQL statements to retrieve or modify data in DB2 tables.

To an IMS/VS terminal user, IMS/VS programs that access DB2 appear identical to IMS/VS.

DB2 supports this cross-product programming environment by coordinating data base recovery services with those of IMS/VS. IMS/VS data communication programmers use the same synchronization and rollback calls in programs that access DB2 data as they would in IMS DB/DC programs that access DL/I data.

System Administration and Operation with IMS/VS

An authorized IMS/VS terminal operator can issue DB2 commands to control and monitor DB2. The terminal operator can also start and stop DB2 data bases.

Even though you perform DB2 functions through IMS/VS, you need to have the TSO attachment facility and ISPF to take advantage of the online functions supplied with DB2 for install and online help. You also need to have the TSO attachment facility to bind application plans and to run batch jobs.

Figure 7 on page 20 shows the relationship between DB2 and IMS/VS.

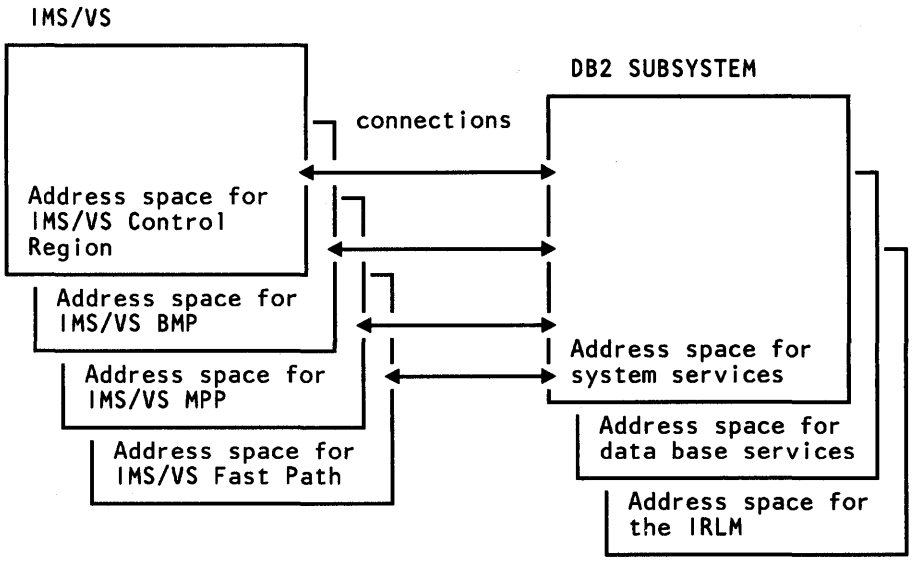


Figure 7. The Relationship of DB2 to IMS/VS

DB2 and the IRLM

The IMS/VS Resource Lock Manager (IRLM) is distributed with and required by DB2 Release 3. The IRLM is responsible for managing all requests for locks and for controlling access to both DB2 and IMS/VS data bases. IMS/VS users must decide whether DB2 and IMS/VS will share one IRLM or use separate IRLMs.

There are two ways in which DB2 and IMS/VS can share a single IRLM:

1. You can run two occurrences of IRLM (loaded from a single DASD copy of IRLM code)—one for DB2, the other for IMS/VS. This simplifies service by preventing confusion about the IRLM level being used. Also, by invoking the IRLM separately for DB2 and IMS/VS, it allows you to set DB2 IRLM parameters independently from IMS/VS IRLM parameters. In this way you can tailor the parameters to the needs of the two different subsystems.
2. You can opt to have DB2 and IMS/VS operationally share a single IRLM. However, this may cause problems, reducing isolation levels and available lock space.

In either case, ensure that you install the latest version of IRLM that has been shipped to you. IRLM Release 4, which is distributed with DB2 Release 3 and with IMS/VS Version 2 Release 2, supports:

DB2 Release 3
DB2 Release 2
IMS/VS Version 2 Release 2
IMS/VS Version 2 Release 1
IMS/VS Version 1 Release 3

Figure 8 on page 21 gives an overview of the IRLM role in a DB2 and IMS/VS environment.

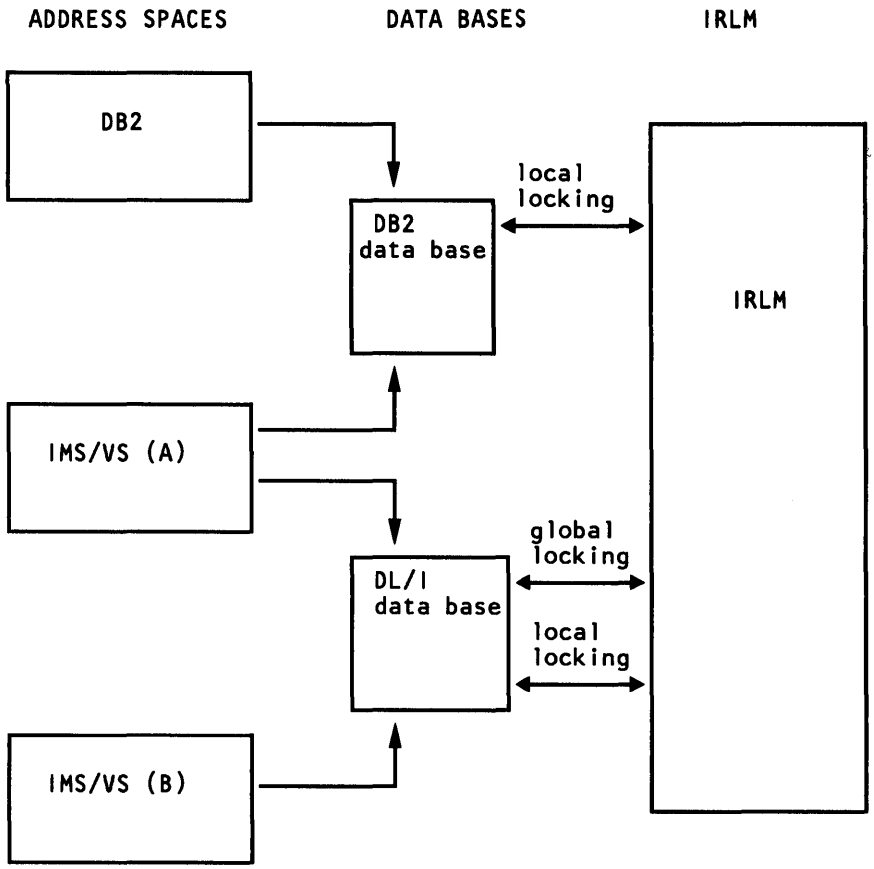


Figure 8. IRLM in an IMS/VS-DB2 Environment

The IRLM performs local locking for DB2 and IMS/VS subsystems each with respect to its DB2 data base or DL/I data base. It performs global locking for IMS/VS subsystems that share a DL/I data base.

IBM DATABASE 2 Data Base Planning and Administration Guide describes locking for the DB2 subsystem.

Except for the DB2 libraries, DB2 data sets cannot be accessed concurrently by more than one DB2 subsystem. If executing multiple DB2 subsystems, you must use a global resource serialization feature of MVS to prevent concurrent access.

DB2 and QMF

The Query Management Facility (QMF), another IBM licensed program, is used for interactive and batch queries and report-writing. It is designed to allow people who are not data processing professionals to use the data that DB2 manages. However, it is also a valuable tool for data processing professionals.

With QMF, you can:

- Access data kept in tables
- Perform calculations on that data
- Produce reports in many different formats
- Create and format charts
- Insert new data and change or delete existing data
- Communicate with other products

QMF provides a choice of two query languages to access data: SQL and Query By Example (QBE).

SQL is the same query language used in DB2. QMF users can execute SQL statements from an ISPF terminal and receive the results from DB2. They can write, store, and run queries. QMF users can also export SQL queries to TSO data sets and use SPUFI to run the queries.

QBE, which is unique to QMF, presents the QMF user with graphic representations of tables. Users can then specify an example of the output they want to produce from the table.

QMF users can also print reports based on their queries. When a user executes a QMF query, QMF produces a formatted report from the data it retrieves from DB2. Users can then tailor the format of the report to their needs and print it. The query and report format can be saved for future use. Figure 9 shows the relationship between DB2 and QMF.

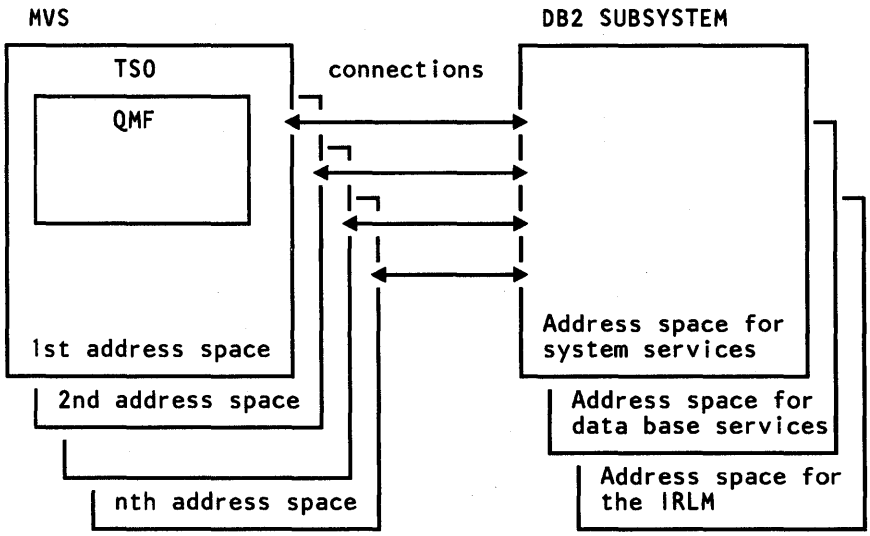


Figure 9. The Relationship of DB2 to QMF

DB2 and DXT

Data Extract (DXT) is a licensed program that extracts and formats data so that it can be loaded into DB2 tables using the DB2 LOAD utility. DXT can extract data from:

- IMS/VS DL/I data bases
- Sequential Access Method (SAM) data sets
- Virtual Sequential Access Method (VSAM) data sets
- DB2 tables
- SQL/DS tables

You do not need to extract all the data in a data base or data set when you use DXT. You can use a statement similar to the SQL SELECT statement to specify the source fields you want to extract and the conditions the source records and segments must satisfy to be eligible for extraction.

DXT can also produce a DB2 LOAD job. This job includes DB2 LOAD control cards that are dynamically generated by DXT to relate fields in the extracted data to target DB2 columns.

Figure 10 shows the relationship between DB2 and DXT.

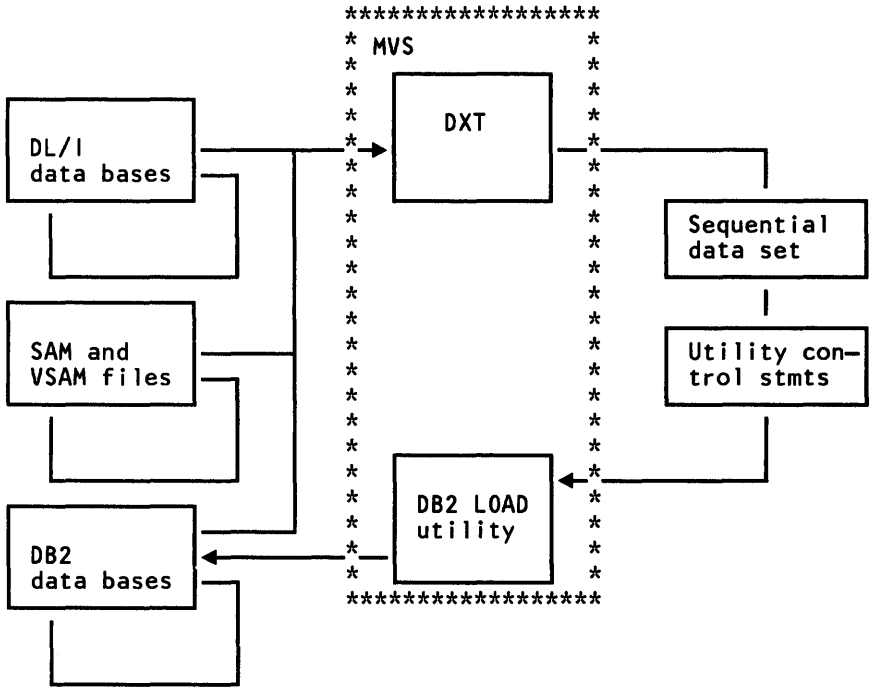


Figure 10. The Relationship of DB2 to DXT

Chapter 2. Introduction to Install and Migration

This chapter introduces you to the steps that you must perform in order to install DB2 Release 3 or migrate to DB2 Release 3.

Loading DB2 Libraries

IBM distributes DB2 Release 3 on two tapes or cartridges, depending on which option you order. Whether you are installing DB2 or migrating to a new release of DB2, you must load the data sets on these tapes or cartridges into DB2 libraries. To do this, you use the System Modification Program (SMP). It processes the distribution tapes, creating the DB2 libraries.

You can use either SMP or SMP/E. If you are not familiar with SMP, refer to *System Modification Program (SMP) System Programmers Guide* or *System Modification Program Extended (SMP/E) User's Guide* before installing or migrating DB2.

You invoke SMP using several jobs that DB2 provides. Detailed information about these jobs appears in *IBM DATABASE 2 Install Guide*.

DB2 Library Names

The SMP jobs create DB2 *distribution* libraries and DB2 *target* libraries. The distribution libraries are used by SMP to build and apply service to the target libraries. The target libraries contain various DB2 components, such as the CLISTs, macros, and execution-time load modules that are used when you run DB2.

Figure 11 lists the distribution and target libraries with the default data set name prefixes for Release 3 and Release 2. If you are migrating from Release 2 you may find the Release 2 names useful. You can change the prefix for Release 3 by using the parameters described in Chapter 4, "Install and Migration Parameters" on page 58.

Release 3 Name	Release 2 Name	Description
DB2 Distribution Libraries		
DSN130.DSNAMACS DSN130.DSNALOAD DSN130.DSNAHELP	DSN120.DSNAMACS DSN120.DSNALOAD DSN120.DSNAHELP	Distribution libraries that build the other libraries and maintain DB2.

Figure 11 (Part 1 of 2). DB2 Distribution and Target Libraries

Release 3 Name	Release 2 Name	Description
DB2 Target Libraries		
DSN130.DSNLINK DSN130.DSNLOAD	DSN120.DSNLINK DSN120.DSNLOAD	Program libraries containing the executable code. The load modules in DSNLINK are in the MVS link list.
DSN130.DSNMACS	DSN120.DSNMACS	Macro library containing macros needed for the attachment facilities, for the initialization parameter macros, and also for some of the instrumentation facility mapping macros.
DSN130.DSNSAMP	DSN120.DSNSAMP	Initialization library containing the sample applications and data, the install and migration jobs, the install and migration parameter defaults, the sample applications for verification, and various initialization data for DB2. Sample programs for each release have unique names. Do not delete the Release 2 sample programs as they are needed to verify migration to Release 3 and fall back to Release 2 in case of failure.
DSN130.DSNCLIST	DSN120.DSNCLIST	A TSO CLIST library containing code for simplifying the install and migration processes, assist in the use of DB2 utilities, and program preparation, and to control DB2I panels.
DSN130.DSNHELP	DSN120.DSNHELP	A TSO HELP data set containing help information for the DSN command processor, and the TSO CLISTs, DSNH and DSNU.
DSN130.DSNSPFP	DSN120.DSNSPFP	An ISPF panel library containing the Installation and the DB2I panels.
DSN130.DSNSPFM	DSN120.DSNSPFM	An ISPF message library containing messages about the ISPF functions.

Figure 11 (Part 2 of 2). DB2 Distribution and Target Libraries

DB2 Load Module Library Considerations

DB2 provides two load module libraries: DSN130.DSNLINK and DSN130.DSNLOAD.

- They are separate so that users who are supporting multiple releases of DB2 can access modules from either release by using STEPLIB and JOBLIB statements. They are also separate so that the number of IPLs required for DB2 corrective service is reduced.
- If you are installing for the first time, you must IPL MVS in order to activate the changes to SYS1.PARMLIB caused by install job DSNTIJMV. You must also IPL MVS to begin using new or changed modules in DSN130.DSNLINK.
- DSN130.DSNLINK contains modules that must be placed in the link list because they are used at subsystem initialization during MVS IPL.

If you are migrating, note that for both Release 2 and Release 3, the load module library DSN130.DSNLINK contains early code modules. The Release 2 early code is upward compatible with Release 3. The Release 3 early code is downward compatible with Release 2.

- Schedule an MVS IPL prior to or during migration to a new release of DB2. This is necessary because migration job DSNTIJMV makes changes to SYS1.PARMLIB which are not recognized by MVS until the next IPL.
- DSN130.DSNLOAD contains modules that may, but do not have to, be placed in the link list.

For other link list considerations refer to *IBM DATABASE 2 Install Guide.*

Tailoring DB2 Parameters

DB2 provides a series of ISPF panels to help install, migrate, and update the product. These panels allow you to specify parameter values that determine many DB2 characteristics, including DB2 library and data set names, storage sizes, security mechanisms, and performance options.

A TSO CLIST that DB2 provides, DSNTINST, uses the values you specify on the ISPF panels to tailor a series of JCL jobs and some other CLISTs.

You must have ISPF to use the panels. If you do not, you must run the DSNTINST CLIST in linear mode (see "Example of Invoking the CLIST Without ISPF" on page 222).

The Panels

DB2 provides a set of panels for each of the three tasks:

- Install DB2
- Migrate DB2
- Update DB2

The value you choose for the first install parameter on the Main Panel determines which set of panels will be presented to you.

Figure 12 on page 27 shows the Main Panel (DSNTIPA1). The panel ID and title appear at the top of the screen. The parameter fields are numbered on the left of the screen and described briefly on the right. If you want the panel identification to appear on panels, enter PANELID ON once on the command line of the panel. This will cause panel IDs to display for your entire ISPF session. Default values for the parameters are listed immediately to the right of the arrows that line the middle of the screen. If a blank appears in this space, the default for the parameter is null.

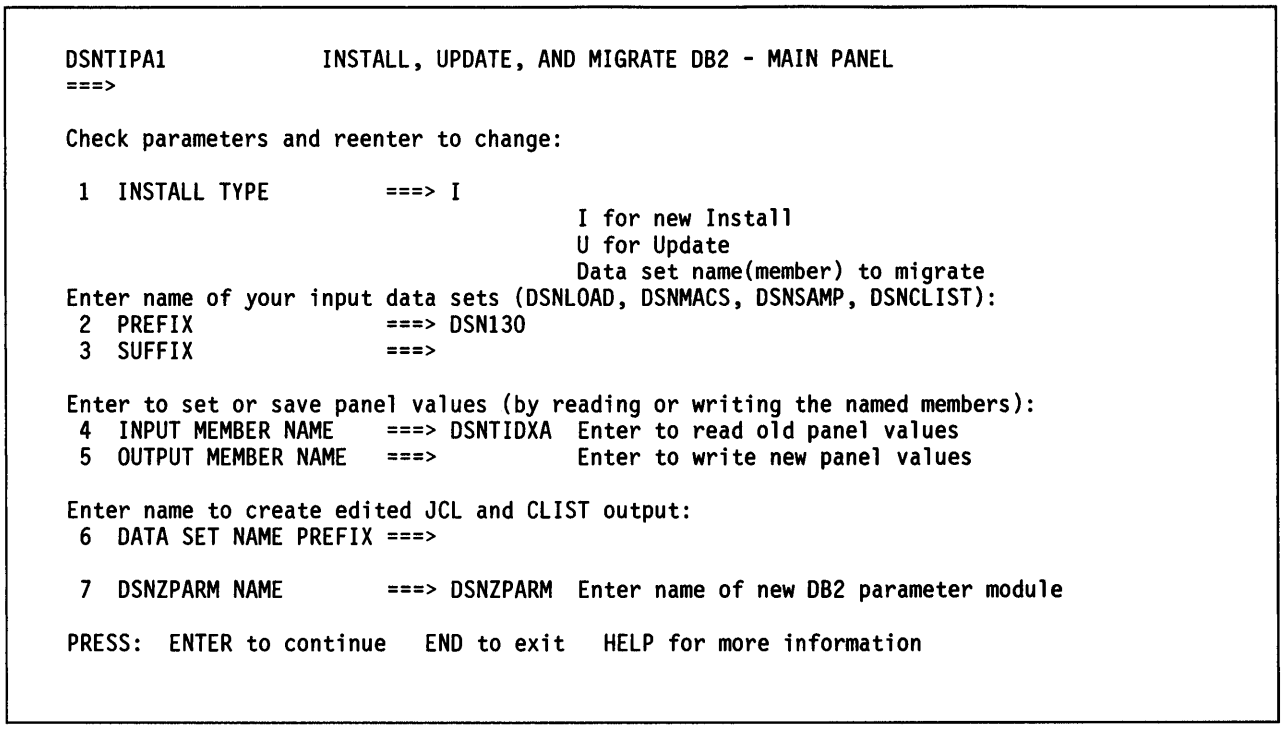


Figure 12. Main Panel (DSNTIPA1)

Panels have both a *panel ID* and a *panel title*. For both install and migration, you use 13 panels. The list of install and migration panels appears below.

Panel ID	Panel Title
DSNTIPA1	Main Panel
DSNTIPA2	Data Parameters
DSNTIPD	Sizes
DSNTIPE	Storage Sizes
DSNTIPO	Operator Functions
DSNTIPF	Application Programming Defaults
DSNTIPI	IRLM Panel 1
DSNTIPJ	IRLM Panel 2
DSNTIPP	Protection
DSNTIPM	MVS Parmlib Updates
DSNTIPL	Log Data Sets
DSNTIPA	Archive Log Data Sets
DSNTIPS	Data Bases and Spaces to Start Automatically

Update panels are discussed in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

The Parameters

Several parameters appear on each panel. In all, you specify values for more than 100 parameters. Most parameters correspond to a parameter that is used by the DSNTINST CLIST. For instance, on the Data Parameters Panel (DSNTIPA2), you specify an alias for the ICF catalog. On the ISPF panel, this parameter is identified as CATALOG ALIAS; in the CLIST, it is identified as VCATALOG. This book uses

both names, usually listing the ISPF panel name first, followed by the CLIST parameter name in parentheses.

All parameters have an IBM-supplied default. The default value appears when you first display a panel, unless you have explicitly overridden the default by passing in a value as a CLIST parameter. If the default is null, no value appears on the panel. You can accept the default by leaving the panel field unchanged, or you can specify a different value by entering it in place of the default.

Defaults are sometimes different for MVS/XA and MVS/370. The default values, as well as the range of acceptable values, are identified in Chapter 4, "Install and Migration Parameters" on page 58. Use this chapter as you plan the parameters values you will enter during the ISPF tailoring session. The default values are stored in parameter members (see pages 62 and 62), which are input to and output from the CLIST.

The DSNTINST CLIST

The DSNTINST CLIST displays panels and uses the values you enter on the panels to control its output. It runs in three modes—install, migrate, and update. The input to this CLIST is:

- The input parameter member from DSN130.DSNSAMP
- If you are migrating from Release 2, the input parameter member used for Release 2
- The input from CLIST invocation and panels
- The skeleton JCL from DSN130.DSNSAMP
- The skeleton CLISTs from DSN130.DSNCLIST

The CLIST produces three things:

1. A new member of DSN130.DSNSAMP that contains the parameter values you enter on the ISPF panels. This member is DSNTID*xx*. You specify the actual name on the OUTPUT PARAMETER NAME (OUTMEM) parameter of the Main Panel (DSNTIPA1).
2. A new *outprefix*.DSNSAMP that contains the skeleton JCL with the parameter values you enter on the ISPF panels.
3. A tailored CLIST data set called *outprefix*.DSNTEMP. This data set contains a set of tailored CLISTs for DB2 and is used as input to job DSNTIJVC, which you run during the install and migration processes.

Figure 13 on page 29 provides an overview of DSNTINST input and output.

There are two prefixes you need to understand to successfully manage DB2. They are parameters 2 and 6 on the Main Panel (DSNTIPA1). Parameter 2, PREFIX (or LIBPREFX as a CLIST parameter), is usually used to construct the data set names that are input to CLIST processes. In this book, we use the default value, DSN130, for this prefix. Parameter 6, DATA SET NAME PREFIX (or OUTPUT as a CLIST parameter), is used to construct data set names that are output to CLIST processes. The default value supplied with DB2 for this parameter is null. In this book, we use the variable *outprefix* for this value.

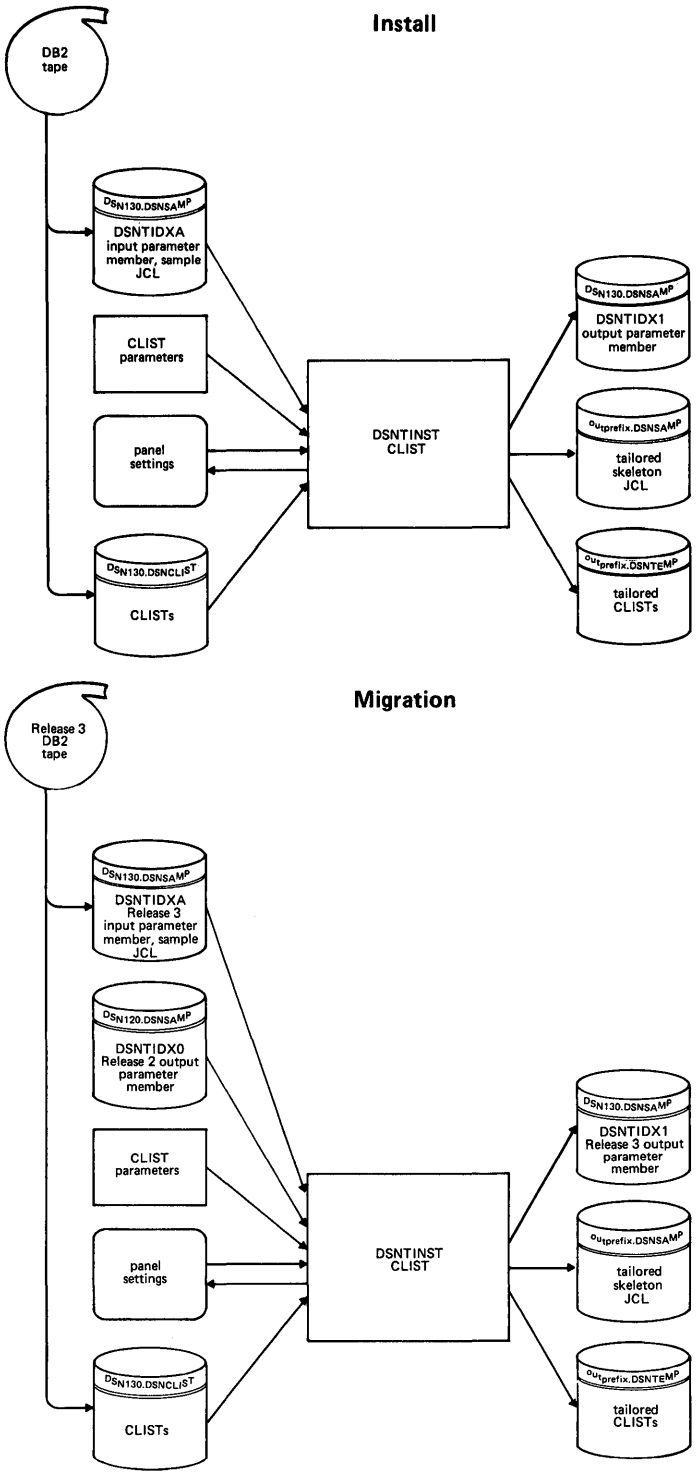


Figure 13. DSNTINST CLIST Input and Output

The Jobs

Using the values you specify on the ISPF panels, the DSNTINST CLIST tailors and loads the skeleton JCL provided in DSN130.DSNSAMP. The CLIST produces *outprefix*.DSNSAMP, which combines the JCL with the parameters you entered on the ISPF panels. This JCL constitutes the *install and migration jobs.*

Each of these jobs helps you perform a task in the install or migration process. Each is composed of one or more JCL procedures and/or job steps, and each is loaded as a separate member of the newly created *outprefix*.DSNSAMP.

The CLIST tailors and loads a different set of jobs for install than it does for migration. Consequently, you will run a different set of jobs, depending on whether you are installing or migrating DB2.

Installing DB2

The jobs automate much of the install process. However, there are steps you must perform manually.

Here is an overview of the process that installs DB2 on your system for the first time:

- Unload Release 3 SMP jobs from the distribution tape
- Allocate the Release 3 distribution and target libraries (Job DSNTIJAE if you are using SMP/E; Job DSNTIJAL if you are using SMP)
- SMP RECEIVE the Release 3 FMIDs (Job DSNTIJRC)
- SMP APPLY the Release 3 FMIDs (Job DSNTIJAP)
- SMP ACCEPT the Release 3 FMIDs (Job DSNTIJAC)
- Invoke the DSNTINST CLIST and specify DB2 parameters on the install panels
- Edit JCL jobs created by the CLIST
- Define Release 3 to MVS and build catalog procedures (Job DSNTIJMV)
- Optionally, define a new ICF catalog and ICF catalog alias (Job DSNTIJCA)
- Define DB2 data sets and data bases (Job DSNTIJIN)
- Initialize DB2 data sets and data bases (Job DSNTIJID)
- Define DB2 initialization parameters (Job DSNTIJUZ)
- Establish the DB2/TSO environment (Job DSNTIJVC)
- Optionally, add CICS routines to DB2 load modules (Job DSNTIJSU)
- IPL MVS
- START DB2 Release 3
- Define temporary table spaces (Job DSNTIJTM)
- Image copy DB2 catalog and DB2 directory (Job DSNTIJIC)
- Define the default data base storage group (Job DSNTIJSG)
- Run the install verification procedure to verify the DB2 system (Jobs DSNTEJ*xx*, except DSNTEJ6)

For a step-by-step description of the install procedure, see *IBM DATABASE 2 Install Guide.*

Migrating DB2

If you are currently running DB2 Release 2 and you wish to migrate your present applications and data to Release 3, the process is mainly a matter of changing existing JCL so that it references the new Release 3 libraries instead of Release 2. Logon procedures, the DB2 the DSNTINST CLIST, and JCL will all need to point to the new Release 3 libraries. In addition, you must install the Release 2 corrective service that will enable you to fall back to Release 2 if you encounter a severe error during or

after migration to Release 3. See the *IBM Database 2 Program Directory* shipped with the product for information about corrective service.

In brief, migration from Release 2 to Release 3 consists of the following procedure:

- Install the Release 2 corrective service to allow fall back (if not already installed)
- Unload Release 3 SMP jobs from the distribution tape
- Allocate the Release 3 distribution and target libraries (Job DSNTIJAE if you are using SMP/E; Job DSNTIJAL if you are using SMP)
- SMP RECEIVE the Release 3 FMIDs (Job DSNTIJRC)
- SMP APPLY the Release 3 FMIDs (Job DSNTIJAP)
- SMP ACCEPT the Release 3 FMIDs (Job DSNTIJAC)
- Image copy Release 2 (Job DSNTIJIC)
- STOP DB2 Release 2
- Invoke the DSNTINST CLIST in migrate mode
- Edit JCL and CLISTs (CLIST DSNTINST with MIGRATE option)
- Define Release 3 to MVS and build cataloged procedures (Job DSNTIJMV)
- Define DB2 initialization parameters (Job DSNTIJUZ)
- Establish the DB2/TSO environment (Job DSNTIJVC)
- Optionally, add CICS routines to DB2 load modules (Job DSNTIJSU)
- IPL MVS (see "DB2 Load Module Library Considerations" on page 25)
- START DB2 Release 3
- BIND SPUFI and DCLGEN (Job DSNTIJSG)
- Run the install verification procedure to verify the Release 3 system (Jobs DSNTEJ*xx*, except DSNTEJ6)

Most of the objects you created on DB2 Release 2 are unaffected by migration. The migration process adopts the applications and data (including views, indexes, temporary data bases, bootstrap data sets, and recovery logs) you created on Release 2, making no changes. It also adopts your Release 2 DB2 catalog and directory. Migration to Release 3 does not require that you recompile, reassemble, bind, or relink-edit Release 2 applications (as was necessary for migration to Release 2).

However, for Release 3, SOME is included as a new reserved keyword. If you used SOME as an identifier in a Release 2 application, then it is necessary for you to delimit it with SQL escape characters and then recompile the application.

Also, to gain the performance benefits of the SQL optimization available in Release 3, you might choose to rebind some of your Release 2 applications under Release 3. Applications which include subqueries, `ORDER BY`, and `GROUP BY` statements are likely candidates for optional rebinding.

In Release 3, the `WITH CHECK OPTION` cascades to dependent views. Views created under Release 2 are not affected and thus are not checked. You can use the following query against the DB2 catalog to identify those Release 2 views dependent upon checked views to which the `WITH CHECK OPTION` did not cascade:

```
SELECT DISTINCT DCREATOR, DNAME, BCREATOR, BNAME
       FROM SYSIBM.SYSVIEWDEP DEP, SYSIBM.SYSVIEWS B,
            SYSIBM.SYSVIEWS D
       WHERE BTYPE = 'V' and B.CHECK = 'Y' and D.CHECK = 'N'
            AND B.CREATOR = BCREATOR AND B.NAME = BNAME
            AND D.CREATOR = DCREATOR AND D.NAME = DNAME
```

For security reasons, you may wish to drop some of the views identified by this query and recreate them under Release 3. Note that the result of this query will include read-only views, which are not recorded as such in the catalog.

Unlike migration from Release 1 to Release 2, migration from Release 2 to Release 3 does not change the names of the following DB2 objects:

- The DB2 catalog is DSNDB06.
- The temporary data base is DSNDB07.
- The skeleton cursor table directory in the DB2 directory is SCT02.

Likewise, there is no need to unload and then reload the catalog for migration to Release 3. This is because the Release 3 catalog format is the same as that for Release 2. There are fewer jobs to run. Consequently, migration to Release 3 is significantly faster and simpler than it was for Release 2.

The definitions, plans, and authorizations that were recorded in your Release 2 catalog are adopted by Release 3. Users with DBADM, DBCTRL, DBMAINT or data base privileges for Release 2 have these privileges for Release 3. You need not GRANT authorization levels again.

Release 2 tables and indexes are also adopted by the Release 3 catalog. The catalog table SYSTABLES, for example, has the same name in the Release 3 catalog as it did in the Release 2 catalog.

Because Release 3 adopts the Release 2 bootstrap, active, and archive log data sets, as well as the catalog and directory, you cannot change their characteristics during migration. You can, however, use the standard update process to modify their characteristics after you complete a migration.

In addition, Release 3 adopts the Release 2 ICF catalog alias. You cannot change this value during either a migration or an update. The parameters that you cannot change during migration are listed in Figure 14. If you try to change any of these parameters, you will receive a warning message, and the CLIST will not change the parameter value. "Panel ID" identifies the panel on which the parameters appear.

Panel ID	Panel Parameter	Description	CLIST Parameter
DSNTIPA2	CATALOG ALIAS	Alias for ICF catalog	VCATALOG
DSNTIPA2	DEFINE CATALOG	Create an ICF catalog?	VCATSTAT
DSNTIPA2	VOLUME SERIAL 2	Second volume for DB2 data sets	VOLSDAT2
DSNTIPA2	VOLUME SERIAL 3	Third volume for DB2 data sets	VOLSDAT3
DSNTIPA2	VOLUME SERIAL 4	Fourth volume for DB2 data sets	VOLSDAT4
DSNTIPA2	VOLUME SERIAL 5	Fifth volume for DB2 data sets	VOLSDAT5
DSNTIPA2	VOLUME SERIAL 6	Sixth volume for DB2 data sets	VOLSDAT6
DSNTIPD	DATA BASES	Number of user data bases in your DB2 subsystem	NUMDATAB
DSNTIPD	TABLES	Average number of tables per data base	NUMTABLE
DSNTIPD	COLUMNS	Average number of columns per table	NUMCOLUM

Figure 14 (Part 1 of 2). Parameters that CANNOT Be Changed for Migration

Panel ID	Panel Parameter	Description	CLIST Parameter
DSNTIPD	TABLE SPACES	Average number of tables spaces per data base	NUMTABSP
DSNTIPD	PLANS	Number of application plans in your DB2 subsystem	NUMPLANS
DSNTIPD	PLAN STATEMENTS	Average number of SQL statements per application plan	NUMSTMTS
DSNTIPD	EXECUTED STATE-MENTS	Average number of SQL statements executed per application plan	NUMSTMTE
DSNTIPD	TABLES IN STMT	Average number of tables per SQL statement	NUMSTMTL
DSNTIPD	TEMPORARY 4K	Bytes of 4K-page work space	NUMTEMP1
DSNTIPD	TEMPORARY 32K	Bytes of 32K-page work space	NUMTEMP2
DSNTIPD	ARCHIVE LOG FREQ	Frequency with which active logs are off-loaded	NUMHRARC
DSNTIPD	UPDATE RATE	Updates, inserts and deletes per hour	NUMCOMHR
DSNTIPP	ICF CATALOG	ICF catalog password	PROTVCAT
DSNTIPP	BSDS PASSWORD	BSDS protection password	PROTBSDS
DSNTIPP	LOG PASSWORD	Active log protection password	PROTLOGS
DSNTIPP	ARCHIVE LOG PW	Archive log password	PROTARCH
DSNTIPP	ARCHIVE LOG RACF	RACF-protect archive log?	PROTARAC
DSNTIPP	DIRECTORY/CATALOG	DB2 catalog, directory, temporary data set password	PROTDIRC
DSNTIPL	NUMBER OF COPIES	Controls dual logging for the active log	LOGSTWO
DSNTIPL	DATA SET PREFIX 1	Prefix for first active log copy	LOGSPRE1
DSNTIPL	DATA SET PREFIX 2	Prefix for second active log copy	LOGSPRE2
DSNTIPL	NUMBER OF LOGS	Number of active log data sets	LOGSNUM
DSNTIPL	BOOTSTRAP NAME 1	Prefix for first BSDS copy	BSDSNAM1
DSNTIPL	BOOTSTRAP NAME 2	Prefix for second BSDS copy	BSDSNAM2

Figure 14 (Part 2 of 2). Parameters that CANNOT Be Changed for Migration

PERMANENT UNIT NAME (VOLSDEVT) on the Data Parameters Panel (DSNTIPD) can be changed during migration, but it does not affect the ICF catalog, DB2 catalog, directory, or logs. It is used for data sets created during migration.

Certain parameters affect the syntax of SQL, and changing them may cause problems. The parameters you should probably not change during migration are listed in Figure 15 on page 34.

Panel ID	Panel Parameter	Description	CLIST Parameter
DSNTIPF	DECIMAL POINT IS	period (.) or comma (,)	DECPOINT
DSNTIPF	SQL STRING DELIMITER	Default, ", or '	DEFSQSTR
DSNTIPF	MIXED DATA	Whether DB2 will accept DBCS data types	DEFMIXED
DSNTIPF	CHARACTER SET	Alphanumeric or Katakana	DEFCHARS

Figure 15. Parameters that SHOULD NOT Be Changed for Migration

Fall Back

Fall back is the process of returning to DB2 Release 2. If you encounter a severe error during or after migration, you can fall back to Release 2. To use Release 3 function after a fall back, you must perform another migration. Fall back for Release 3 is simpler and faster than it was for Release 2; there are fewer jobs to run.

Fall back for Release 3 includes the following steps:

- Recover data bases if necessary
- Stop DB2 Release 3
- Rename the JCL procedures (Release 3 Job DSNTIJFV)
- START DB2 Release 2
- BIND (REPLACE) SPUFI and DCLGEN (Release 3 Job DSNTIJFS)
- Verify the Release 2 subsystem (Jobs DSNTEJ*xx*, except DSNTEJ6)

Fall Back Restrictions

For Release 3, tables, views, and plans may be dependent on new external functions. If a value of B is in the IBMREQD column of catalog table SYSTABLES, SYSVTREE, SYSVIEWS, and SYSPLAN, then those tables, views, and plans are dependent on Release 3 and will be frozen after a fall back to Release 2. That is, most operations on these objects are prohibited until you remigrate to Release 3. Similarly, a plan containing any frozen items cannot be bound until you remigrate to Release 3. The Release 3 functions that can cause objects to be frozen after fall back to Release 2 are:

Substring and concatenation
Scalar functions
Date/time expressions
UNION ALL
Single precision floating point

Release 3 also adds SQL arithmetic error handling to DB2; although this new function does not result in frozen objects or bind errors after fall back, run time errors may occur under Release 2, depending on the data.

Release 3 support for user-defined linear data sets (LDS) is not effective when you fall back to Release 2. You will not be able to access DB2 data sets of the VSAM LDS type until you remigrate to Release 3. Similarly, Release 3 relaxes the Release 2 restrictions on views using `GROUP BY`, `HAVING`, and `DISTINCT`. After fall back, you will not be able to access views which exploit this flexibility until you remigrate to Release 3.

The SQL optimization added to Release 3 is temporarily lost if a plan is rebound under Release 2. If the plan is rebound after remigration to Release 3, optimization is incorporated into the plan again.

Likewise, SQL statements which Release 3 adds to formatted dumps will be lost if a plan is rebound under Release 2. After remigration to Release 3, the plan must be rebound in order for the statements to be saved for dump formatting.

Enhanced UNION compatibility and ANS SQL are functions new to Release 3. After falling back to Release 2, plans using these functions can be successfully executed. Attempted REBINDs or BINDs in Release 2 will fail and make the plans inoperative until they are bound in Release 3.

Remigration

Migration to Release 3 following a fall back to Release 2 (remigration) is like migration, only simpler. The only steps required for a remigration are:

- Run the Release 2 image copy job (Release 2 DSNTIJIC)
- STOP DB2 Release 2
- Run the step in Release 3 Job DSNTIJMV that creates the cataloged procedures.
- START DB2 Release 3
- BIND (REPLACE) SPUFI and DCLGEN (Release 3 Job DSNTIJFS)
- Verify the Release 3 subsystem (Release 3 Jobs DSNTEJ*xx*, except DSNTEJ6)

In addition, you must perform some tasks manually. These tasks, as well as the tasks performed by the jobs listed above, are explained in *IBM DATABASE 2 Install Guide*.

Updating DB2 Parameters

After you install DB2 or migrate to DB2 Release 3, you can modify the parameters you specified. Some of the parameters can be modified by running the DSNTINST CLIST in update mode using the update panels. You can also update parameters by invoking the CLIST and passing in parameter values without displaying the ISPF panels.

For information on how to use the update process, see Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

There are only two parameters that you cannot update: the CATALOG ALIAS parameter (VCATALOG) of the Data Parameters Panel (DSNTIPA2), and the associated DEFINE CATALOG parameter (VCATSTAT). These parameters affect the ICF catalog (not the DB2 catalog). To define a new ICF catalog, you must reinstall DB2. If you do this, you will lose your ICF catalog and consequently all of the data associated with your DB2 subsystem. Before you install DB2, carefully plan your value for CATALOG ALIAS (VCATALOG).

Certain other install parameters affect the syntax of SQL, and updates to these parameters should be made with caution. See Figure 15 on page 34 for a list.

Chapter 3. Estimating DB2 Storage Needs

This chapter provides information about how to estimate the amount of direct access and virtual storage required by DB2. The parameters you specify when you run the DSNTINST CLIST affect the sizes of some data sets and the amount of virtual storage needed. Other data sets have a fixed size.

This chapter explains how to calculate storage requirements for each of three typical sites: a large site, a medium-sized site, and a small site. These models are based on the following assumptions:

- The **small** site supports a small number of DB2 users and accepts the IBM-supplied parameter defaults for MVS/370. The small site has approximately 100 plans, 50 application data bases, and 300 tables.
- The **medium**-sized site supports more extensive use of DB2 data bases and accepts the parameter defaults for MVS/XA. The medium-sized site has approximately 200 plans, 200 application data bases, and 1200 tables.
- The **large** site supports heavy use of DB2. The large site has approximately 400 plans, 400 application data bases, and 2400 tables.

When you first install DB2, we recommend that you follow one of these three models. After you have had some experience with your subsystem, you will be able to modify parameters to suit your needs better.

Predefined Model Sites

This section summarizes the amount of DASD space needed for DB2 operation based on each of the three models. It assumes that, when running the CLIST, you accept the default values for the number of data bases, tables, and application plans expected at your site. You specify these values on the Sizes Panel (DSNTIPD).

If you do not accept the default values for these parameters, you can calculate the storage needed for each of the DB2 data sets by using the information in "Active Log Data Sets" on page 39. For other data sets, you can use the formulas in the CLIST. After you make the calculation for each data set, you can calculate the total requirements.

Total DB2 Space Requirements

To determine the space requirements based on your DASD device model, check the values listed in Figure 16 on page 37. The space requirements **do not** include space for user data bases, image copies, archive logs, or temporary data sets that you create during install or migration.

Site Size	3330	3350	3380
Small	918	462	366
Medium	1657	820	643
Large	10224	4894	3897

Figure 16. Estimated DASD Cylinder Requirements for DB2 by Site Size

Space Requirements for DB2 System Data Sets

Figure 17 provides space requirements in megabytes (Mb) for the different DB2 data sets. (The calculations were done with more precision than is shown in the figure; this accounts for the discrepancy between individual amounts and totals.) The DB2 libraries require a fixed amount of space, regardless of the size of your site. On the other hand, the DASD requirements for active logs and DB2 catalog increase significantly as the size of your site increases. You may need additional space for archives, logs, image copies and other working data sets. Device-specific estimates in cylinders for items in Figure 17 appear later in this chapter.

Site Size	DB2 Code	DB2 Catalog	Directory	Active Logs	Bootstrap Data Sets	Temporary Data Base	Total for DB2 Data Sets
Small	109	18	1	124	1	7	**261**
Medium	109	71	4	248	1	25	**458**
Large	109	142	7	2465	1	50	**2775**

Figure 17. Estimated Space Requirements (in Mb) for DB2 Data Sets

DASD Requirements for the DB2 Libraries

Requirements in cylinders by device for DB2 libraries, SMP data sets, and SMP/E data sets are shown in Figure 18, Figure 19 on page 38, and Figure 20 on page 38.

Library	3330	3350	3380
DB2 Distribution Libraries			
DSN130.DSNAMACS	47	25	19
DSN130.DSNAHELP	1	1	1
DSN130.DSNALOAD	48	24	20
Subtotal for Distribution Libraries	**96**	**50**	**40**
DB2 Target Libraries			
DSN130.DSNCLIST	3	2	2

Figure 18 (Part 1 of 2). Estimated Space Requirements (in Cylinders) for DB2 Distribution and Target Libraries

Library	3330	3350	3380
DSN130.DSNHELP	1	1	1
DSN130.DSNSPFM	1	1	1
DSN130.DSNSPFP	8	5	4
DSN130.DSNMACS	4	3	2
DSN130.DSNSAMP	29	15	12
DSN130.DSNLOAD	34	17	14
DSN130.DSNLINK	1	1	1
Subtotal for Target Libraries	**81**	**45**	**37**

Figure 18 (Part 2 of 2). Estimated Space Requirements (in Cylinders) for DB2 Distribution and Target Libraries

SMP Data Sets	3330	3350	3380
SMPACDS	20	10	8
SMPACRQ	1	1	1
SMPCDS	20	10	8
SMPCRQ	1	1	1
SMPLOG	10	5	7
SMPMTS	1	1	1
SMPPTS	6	3	3
SMPSCDS	6	3	3
SMPSTS	6	3	3
SMPTLIB	126	66	49
SMP 4 Total	**197**	**103**	**84**

Figure 19. Estimated Space Requirements (in Cylinders) for SMP Data Sets

SMP/E Data Sets	3330	3350	3380
SMPCSI	26	12	10
SMPLOG	10	5	7
SMPMTS	1	1	1
SMPPTS	6	3	3
SMPSCDS	6	3	3
SMPSTS	6	3	3

Figure 20 (Part 1 of 2). Estimated Space Requirements (in Cylinders) For SMP/E Data Sets

SMP/E Data Sets	3330	3350	3380
SMPTLIB	126	66	49
SMP/E Total	**181**	**93**	**76**

Figure 20 (Part 2 of 2). Estimated Space Requirements (in Cylinders) For SMP/E Data Sets

DASD Requirements for the DB2 Catalog

Requirements in cylinders by device for the entire set of DB2 catalog data sets and their indexes are shown in Figure 21.

Site Size	3330	3350	3380
Small	70	35	25
Medium	280	140	100
Large	560	280	200

Figure 21. Estimated DASD Cylinder Requirements for the DB2 Catalog by Site Size

For information about how to change the size of catalog data sets after you install or migrate DB2, see "Changing Catalog and Directory Size and Location" on page 201.

DB2 Directory

Directory space depends mainly on the number of user data bases, application plans, and tables in the DB2 subsystem.

Site Size	3330	3350	3380
Small	5	3	2
Medium	13	7	5
Large	26	13	10

Figure 22. Estimated DASD Cylinder Requirements for the DB2 Directory by Site Size

Active Log Data Sets

Active log data sets record significant events and data changes. They are periodically off-loaded to the archive log. Consequently, the space requirements for your active log data sets depend on how often DB2 data is changed at your site and how often you off-load the record of those changes to the archive log.

If you change data frequently and off-load it to the archive log infrequently, you will require a large amount of DASD space for the active log. If off-loading is to occur once each day, the active log data sets should be able to hold the log records your subsystem produces during one day of processing.

These are the assumptions concerning each of the three models:

- The small site has a data change rate of 1800 per hour, and the active log is off-loaded once each day.

- The medium-size site has a data change rate of 3600 per hour, and the active log is off-loaded once each day.
- The large site has a data change rate of 36,000 per hour, and the active log is off-loaded once each day.

As an example, here is how the DSNTINST CLIST calculates the amount of DASD space required by a small site:

1. During the ISPF tailoring session, you specified:
   - An archive period estimate of 24 hours—ARCHIVE LOG FREQUENCY parameter (NUMHRARC) on the Sizes Panel (DSNTIPD)
   - A data change rate estimate of 1800 changes per hour—UPDATE RATE parameter (NUMCOMHR) on the Size Panel (DSNTIPD)
2. DB2 takes the size of a typical row as 200 bytes. Each log record that records a data change includes a copy of the "before" row and a copy of the "after" row, so the length of an average data change log record is twice the length of the average row. Therefore, data change log records average 400 bytes.
3. Other types of log records are comparatively small in size, and are fixed length. The length of each type depends on the information it contains.
4. The size of the active log, including DASD track overhead, is estimated as:

```
Data set size = (data change log record size) x
                (data change rate per hour) x
                (archive period)
              = 400 bytes x 1800 per hour x 24 hours
              =  21 Mb
  dual logs   =  42 Mb
  3 data sets = 124 Mb
```

If you accept the defaults of three active log data sets and dual logging, DB2 creates six active log data sets. Figure 23 shows estimated storage requirements for active log data sets assuming dual logging. Figure 24 on page 41 shows the amount of space required for active log data sets on various IBM DASD devices. The estimates in both figures include DASD track overhead.

Site Size	Archive Period (hrs)	Data Change Rate (per hr)	Space for each Active Log Data Set (Mb)	Total Space for 6 Active Log Data Sets (Mb)
Small	24	1800	21	124
Medium	24	3600	41	248
Large	24	36,000	411	2465

Figure 23. Estimated Requirement (in Mb) for Active Log Data Sets

Site Size	3330	3350	3380
Small	456	222	174
Medium	912	438	348
Large	9096	4326	3462

Figure 24. Estimated DASD Cylinder Requirements for the Active Log by Site Size

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

- Tape utilization

  If the size of an active log data set is small compared to the size of a tape, tape utilization is fairly low. The planning sizes for tapes are 30 Mb for 1600 BPI tape, 100 Mb for 6250 BPI tape, and 200 Mb for 3480. Use of larger block sizes for archive logs can increase these figures by up to 40 percent. You specify block size for archive logs wiht the BLOCK SIZE (ARCHSIZE) parameter on the Archive Log Data Sets Panel (DSNTIPA).

- Checkpoint frequency

  You specify checkpoint frequency with the CHECKPOINT FREQ (OPCHKFRQ) parameter on the Operator Functions Panel (DSNTIPO). Restart speed is improved if there is space for at least 10 checkpoint intervals in a data set. Because we have assumed the average record takes 200 bytes, we can multiply the checkpoint frequency by 200, then multiply that product by 10 to get another recommended minimum. The default checkpoint frequency is 1000, giving a minimum log size of 2 Mb.

Bootstrap Data Sets (BSDSs)

Each BSDS requires 0.5 Mb per data set. If you are installing, DB2 automatically allocates two copies of the BSDS. If you are migrating, Release 3 adopts the BSDS characteristics you specified for Release 2. That is, if you specified 2 BSDS copies for Release 2, you will have 2 copies for Release 3. The total space requirement is approximately 1 Mb for both BSDSs. The BSDSs at any size site require approximately 4 cylinders for a 3330, 2 cylinders for a 3350, and 2 cylinders for a 3380 device.

Archive Log Data Sets

If you decide to place the archive log data sets on DASD or MSS, you need to reserve enough space on those devices. The active log data set and the BSDS are both written to the same location. Therefore, you must reserve space as large as the active log and the BSDS combined.

The DSNTINST CLIST uses values you specify on the install panels to compute archive primary and secondary space. Primary space for the archive log is the same as for the active log. Secondary space is small and is used if it is needed for cylinder rounding differences on different devices.

Temporary Data Base

The temporary data base is used as temporary space for processing SQL statements that require working storage. Figure 25 shows the DASD requirements for the temporary data base.

Site Size	3330	3350	3380
Small	25	12	10
Medium	90	45	35
Large	180	85	70

Figure 25. Estimated Space Requirements (in Cylinders) for Temporary Data Base by Device

Default Data Base

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

DB2 User Data Bases

There are two methods to estimate the DASD storage you need for the table spaces and index spaces that make up your DB2 user data bases. The first method can be used to make approximate estimates. The second method is more precise and more complicated.

In both estimates, you will see references to records and fields; these are DB2's internal representation of rows and columns.

Approximate Estimates: Use the following formula to make an approximate estimate of your DASD storage needs for DB2 user data bases.

```
Amount of space = 2 x (total amount of data)
```

In this formula, 2 is a combined multiplier that can be as small as 1.23, or as high as 4 or more. The combined multiplier results from multiplying the individual factors described below. Note that there is substantial variability and that some of the individual factors apply to all data sets on direct access devices.

- Record overhead—typical 1.10, minimum 1.01, maximum 4.

  This factor includes 8 bytes of record overhead; it depends on how well the records fit into the DB2 page. For long records that fit well on a page, the factor is about 1.01. If the records take just over half of the page or they are small (less than 22 bytes of data), this factor can be 2 or more.

- Free space within a page—typical 1.05, minimum 1.00, maximum 200.

  This factor depends upon your choice for the PCTFREE and FREEPAGE parameters on CREATE TABLESPACE and ALTER TABLESPACE statements. Free space is preserved during LOAD and REORG so that inserted rows and rows that need more space after updating can use the free space.

If you have specified 0 for FREEPAGE, the free space factor can be calculated in the following manner:

```
Free space within a page = 1 / (1 - PCTFREE / 100)
```

If you have chosen a value other than 0 for FREEPAGE, use the following calculation for the free space factor:

```
Free space
within
a page =  (1 / (1 - PCTFREE / 100))  X  ((FREEPAGE + 1) / FREEPAGE)
```

- Unusable space on a track—typical 1.16, minimum 1.02, maximum 1.16.

  This factor depends on the device type. DB2 uses 4096-byte records. The number of records that fit on a track and the amount of unusable space for various devices are:

Factor	3330	3350	3380
Records per track	13030	19069	47476
Blocks per track	3	4	10
Unusable space factor	1.06	1.16	1.16

Figure 26. Track Capacity by Device Type

- Unused space within an allocated data set—typical 1.20, minimum 1.02, maximum (none).

  Unused space occurs for all data sets as unused tracks or part of a track at the the end of the data set. The amount of unused space in a data set depends upon the volatility of the data, the amount of management done, and the data set size. Generally, large data sets can be managed more closely. Data sets that do not change in size are easier to manage. Careful size estimation and space management could reduce the unused space factor to 1.02 in some cases.

- Indexes—typical 1.20, minimum 1.01, maximum 2.

  The space for indexes depends upon the number of indexes and the record length of those indexes. For long records with one short index, the index factor might be as low as 1.01. For a row with every column indexed, the index factor is 2.

To combine the factors listed above, multiply them together. Figure 27 on page 44 shows calculations of the combined multiplier for three different data base designs. In each design, assume the device type to be 3350 or 3380; therefore, the unusable space factor is always 1.16.

- The *tight design* is carefully chosen to save space and allows only one index on one short field.
- The *loose design* allows a large value short of the maximum for each of the five factors. Free space adds 30 percent to the estimate, and indexes add 40 percent.
- The *medium* design has values between the other two designs and might serve as a rule-of-thumb in an early stage of data base design.

Factor	Tight Design	Medium Design	Loose Design
Record overhead X	1.02	1.10	1.30
Free space X	1.00	1.05	1.30
Unusable space X	1.16	1.16	1.16
Unused data set space X	1.02	1.25	1.50
Indexes =	1.02	1.20	1.40
Combined multiplier	**1.23**	**2.01**	**4.12**

Figure 27. Calculation of Combined Multiplier by Data Base Design

In addition, external storage devices are required for utility work data sets used for:

- Image copies of data sets (can be on tape)
- System libraries, system data bases, and log data sets
- Temporary work files for utility and sort jobs

A rough estimate of the additional external storage needed is three times the amount calculated above for DASD storage.

More Precise Estimates: A more precise way to estimate the space you need for data in table spaces and indexes is given below.

For **table spaces**, assume 1 table per table space:

```
The number of records per page immediately after load =
  FLOOR (usable page size/stored record length) x 95%
```

where:

- FLOOR means round down to an integer value
- Usable page size = 4096 - 22 (or 32768 - 22)
- Stored record length = sum of field lengths (use the average field length for VARCHAR, LONG VARCHAR, VARGRAPHIC, and LONG VARGRAPHIC).

  + 8 (record overhead)
  + 1 for each nullable column
  + 2 for each varying-length field

- Free space = 5% (assume defaults of 5% free space within each page and no free pages (PCTFREE and FREEPAGE parameters of CREATE TABLESPACE statement)

  This free space can be altered with the PCTFREE and FREEPAGE parameters of the CREATE TABLESPACE and ALTER TABLESPACE statements.

For **index space**, assume a unique index:

```
The number of index entries
per physical page immediately after load =
  FLOOR (usable page size/index entry length) x 90%
```

where:

- FLOOR means round down to an integer value
- Usable page size = (4096 - 29 - SUBPAGE overhead)

  where:

  - SUBPAGE overhead = 17 if SUBPAGES = 1, or
  - SUBPAGE overhead = (21 + KL) x (SUBPAGES) if SUBPAGES > 1

    SUBPAGES represents the index SUBPAGES specified on the CREATE INDEX statement.

    KL represents key length.
- Index entry length = KL + index overhead

  where index overhead = 4 + 1 per nullable column that is part of the key
- Free space = 10% (assume defaults of 10% free space within each SUBPAGE and no free pages (PCTFREE and FREEPAGE parameters of CREATE INDEX statement)

  This free space can be altered with the PCTFREE and FREEPAGE parameters of the CREATE INDEX and ALTER INDEX statements.

Notes:

1. Field length = number of bytes in the above calculations.
2. The field lengths in table space calculations should take into account any FIELDPROC or EDITPROC changes to the field lengths. For indexes, however, FIELDPROC and EDITPROC do not affect key length.
3. Nonunique index entries per physical page may be estimated by the formula:

   ```
   FLOOR ((usable page size/index entry length)
          x average number of records per key) x 90%
   ```

   where usable page size and index entry length are defined as above for unique indexes, except that:

   index overhead = 6 + (4 x average number of records per key) + 1 per nullable column that is part of the key
4. This calculation is for indexes created before load. Indexes created after load will require somewhat more index space because the index is built by the insert process rather than by the load process. Use a factor of 75 percent instead of 90 percent.
5. For indexes created over VARCHAR and VARGRAPHIC columns, use the maximum size as defined in the CREATE TABLE statement as the key length for index calculations.
6. The formulas as given apply to first-level or leaf index pages. For higher-level or nonleaf index pages, the unique index formula may be used with the following modifications:

   - SUBPAGES = 1, always
   - subtract an additional 3 bytes from usable page size
   - index overhead = 3 + 1 per nullable column that is part of the key

Example of Calculation of Data and Index Space: Assume defaults of 5 percent free space within each page and no free pages (PCTFREE and FREEPAGE parameters of CREATE TABLESPACE statement), and that the user data equals

```
   100000 records
   20 fixed-length columns, non nullable    Total:   200 bytes
    5 fixed-length columns, nullable        Total:    30 bytes
    1 varying-length field, nullable      Average:   100 bytes
    1 unique index with key length = 18 and no nullable columns

Create a unique index over 3 columns, totaling 18 bytes

Data physical page size = 4K bytes
Index SUBPAGES = 4

Data space calculation:

    Usable page size = 4074 (4096-22)
    Stored record length = 346
      User data = 330
      Record overhead = 8
      Null overhead = 6
      Variable field overhead = 2
    Usable page size/stored record length = 11.775

    Thus maximum records per page = 11

    11 x 95% = 10

    Thus records per page after load = 10

    100000 records/10 = 10000

    Thus data space = 10000 pages

Index space; unique index:

    Usable page size = 3911  (4096 - 29 - (21 + 18) x 4)
    Index entry length = 22  (18 + 4 + 0)
    Maximum index entries per physical page = 177  (3911/22)
    Index entries per page after load = 159  (177 x 90%)

Thus, 100000 rows/159 = 628 first-level index pages

    628/159 = 4 second-level index pages
    4/159 = 1 third-level index page

Total index space = 633 pages
Total data space = 10000 pages
```

Dump Data Set Size

Dump data set size should be based on the region sizes allocated to DB2 and the storage areas defined by DB2 for inclusion in the dump data set. The following guidelines apply:

- We recommend a minimum of two dump data sets.
- To ensure that all DB2 defined storage areas are included in the dump data set, 50 cylinders of 3380 DASD space is recommended for each SYS1.DUMP*xx* data set defined. This size is based on the storage areas defined during SVC dump invocation by DB2:

1. 1.5 Mb of DB2 volatile summary storage data
2. Dumps of the following address spaces:
   a. DB2 system services address space
   b. DB2 data base services address space
   c. Allied address space of the failing allied task
3. DB2 passes these parameters to the MVS SDUMP service aid through the SDATA keyword:

   SQA, ALLPSA, LSQA, SUMDUMP, and CSA (subpools 231 and 241).

Note: The DB2 resident trace table is obtained from CSA (subpool 231). Refer to *MVS/Extended Architecture System Programming Library: System Macros and Facilities, Volume 2* for more information on the MVS SDUMP service aid.

- Once DB2 SVC dump processing is complete, MVS message IEA911E indicates whether or not sufficient space was available in the dump data set to contain the requested storage areas. If this message indicates that a partial dump was taken, IBM may request that you recreate the problem if storage areas required for problem determination are not included in the dump.

Using the DSNTINST CLIST to Calculate Storage

If you choose not to use the estimates for the three model sites, you can use the detailed information for DASD storage estimates in the DSNTINST CLIST. We recommend you use the model site estimates the first time you install DB2. The model approach described in "Predefined Model Sites" on page 36 should provide you with adequate estimates for DB2 DASD usage. Later, after your site has had some experience in operating DB2, you may want to recalculate your DASD estimates.

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

To determine the algorithms that DB2 uses, merely print or edit the CLIST. You can find the information you are looking for in the CLIST, or you could run the CLIST to calculate the sizes.

DB2 Virtual Storage Layout

DB2 uses four private address spaces:

IRLM address space;
DB2 allied agent address space;
DB2 data base services address space; and
DB2 system services address space.

DB2 also uses common service area (CSA). Figure 28 on page 48 shows the DB2 virtual storage layout for both MVS/370 and MVS/XA.

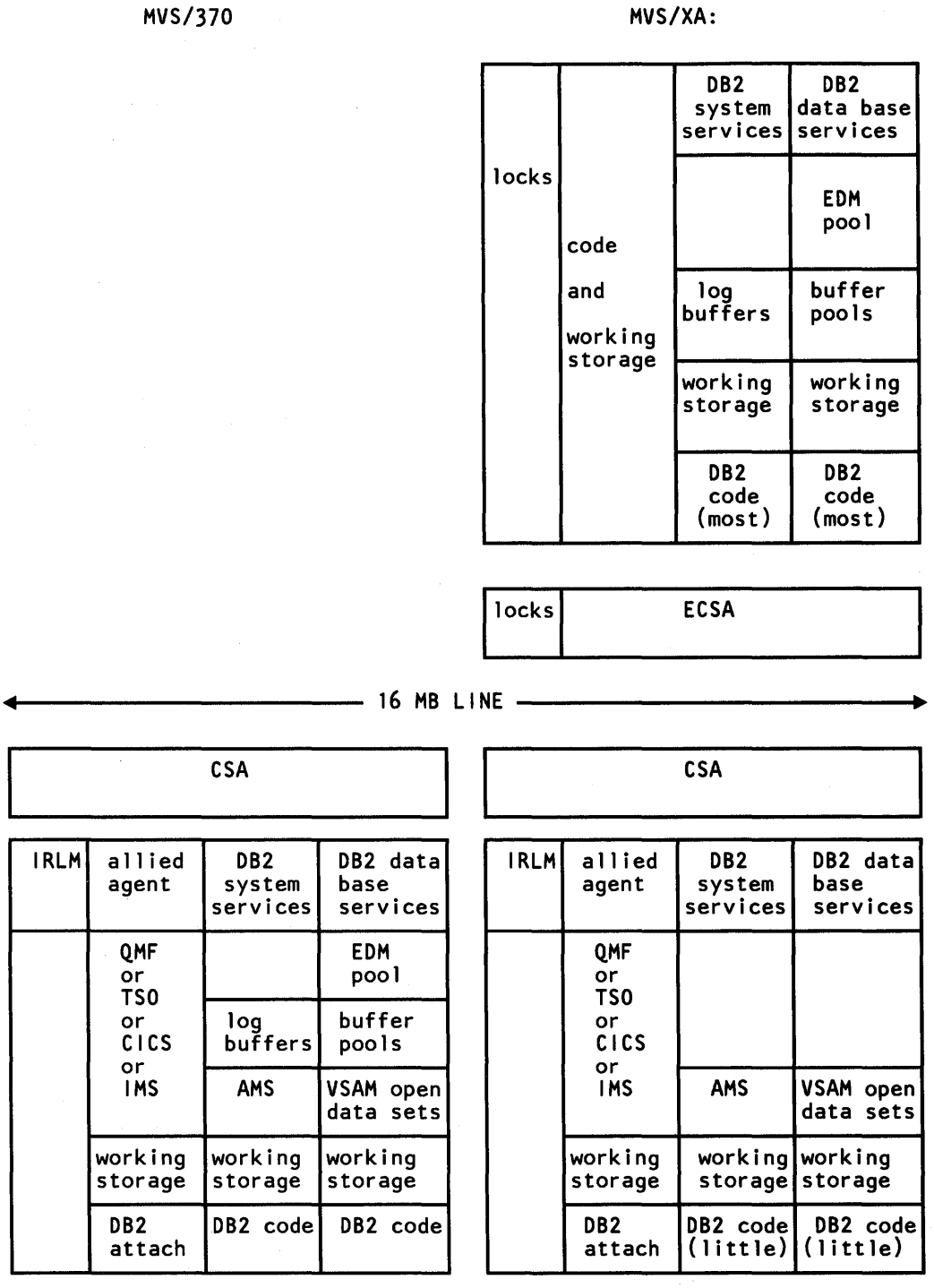


Figure 28. DB2 Virtual Storage Layout for MVS/370 and MVS/XA.

IRLM Address Space

The IRLM is required by DB2 to perform lock management. When you install DB2, if you specify YES for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), or if you specify PC=Y on the START IRLMPROC command, IRLM will place its control block structure in the IRLM private address space. If you are using MVS/370, you may need to accept this default to relieve DB2 storage constraints.

If you specify NO for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), or if you specify PC=N on the START IRLMPROC command, IRLM will place its control block structure in common service area (CSA). If your system is MVS/XA, extended common service area (ECSA) will be used if sufficient space is available. In this case, the amount of storage available to IRLM is limited by the value you specify for MAXIMUM CSA OR ECSA (IRLMMCSA) on IRLM Panel 2 (DSNTIPJ). The IBM-supplied default value for MAXIMUM CSA OR ECSA (IRLMMCSA) is 300K for MVS/370, or 5M for MVS/XA.

Using CSA or ECSA requires less processor time, but it can also reduce the size of private address spaces.

You can estimate the IRLM control block structure at 200 bytes per lock. Initially, plan 5 megabytes for the IRLM control block structure. (This can be in IRLM private address space, or in ECSA for MVS/XA systems, as described above.) Monitor DB2 lock usage and CPU usage at your site. Adjust the install parameter values for IRLM only after you have gained some experience with your DB2 subsystem.

Allied Agent Address Space

DB2 refers to the user address spaces as the allied agent address space. This can include TSO, IMS/VS, CICS, and batch address spaces. The size of the DB2 attach code in the allied agent address space depends on which attachment facilities you use. TSO requires about 130K for the DSN command. CICS, CAF, and IMS/VS require about 30K for the DB2 attach code. For all attachment facilities, the DB2 attach code must run below the 16 Mb line of MVS/XA virtual storage. Applications can run above the 16 Mb line.

DB2 Data Base Services Address Space

The DB2 data base services address space is the largest DB2 address space. Initially, plan for a minimum of 5200K bytes for this address space in MVS/370, and 3200K in MVS/XA. You will find a detailed discussion of the various components of the data base services address space in the remainder of this chapter. For more information, see "Main Storage Size" on page 50.

DB2 System Services Address Space

As shown in Figure 28 on page 48, the DB2 system services address space needs less space than the data base services address space. We recommend you specify 5000K bytes for the system services address space.

Common Service Area (CSA)

For MVS/XA, most of the DB2-required areas reside in the extended common service area (ECSA). The residual CSA requirement is approximately 160K bytes. Ensure that the amount of ECSA specified for MVS/XA is adequate. For more information, see "Ensure ECSA Size is Adequate" on page 194.

For MVS/370, DB2 requires a minimum of approximately 600K bytes. This should be sufficient to support approximately 40 to 50 concurrent connections. This estimate assumes that the IRLM lock control block structure resides in the IRLM private address space and is not in the CSA (that is, you specified YES for the CROSS MEMORY (IRLMPC) parameter on IRLM Panel 2, DSNTIPJ). Each additional connection will require approximately 1K bytes of additional storage in the CSA; fewer connections still require the minimum amount of 600K bytes.

Main Storage Size

One of your tasks as a system administrator is to specify values during the ISPF tailoring session that the DSNTINST CLIST will use to calculate main storage size. These values are especially important in an MVS/370 environment because of the virtual storage constraints on the private area of an address space. We recommend that you determine these values based upon your estimated application workload before you install or migrate DB2.

These values provide an estimate of the private area needed by the DB2 data base services address space, largest of the DB2 address spaces. If virtual address space is available in your MVS environment, it is a good idea to increase the value of the REGION parameter on the DB2 started tasks. If the estimated virtual address space is not available, you may need to reevaluate the sizes you have requested.

Note: These calculations are planning estimates. The values noted do not provide the exact limits, but indicate a reasonable range of values. More detailed information is provided in "Improving Virtual Storage Utilization" on page 193.

This section presents information about the values used to calculate region size:

- Buffer pool size
- EDM pool size
- Data set control block size
- Working storage size

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

Of these values, the sum of the buffer pool size, EDM pool size, data set control block size (for your table spaces and indexes), and working storage size must fit within your region size that is permitted to DB2. This sum is important for MVS/370, but less so for MVS/XA, because most of the space is allocated above the 16 Mb line of MVS/XA virtual storage.

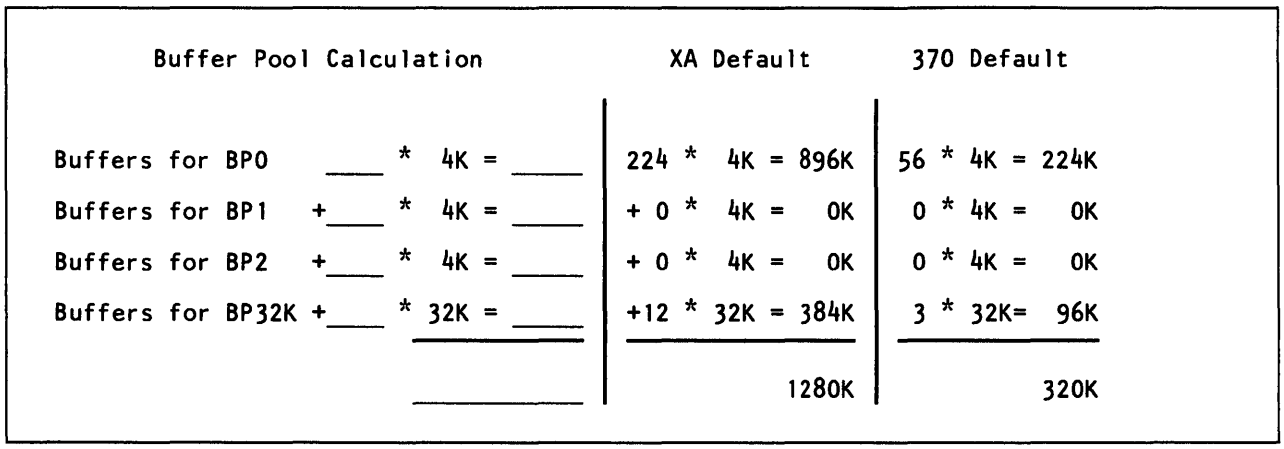
After you specify the values listed above, the CLIST calculates the EDM pool size and the size needed for the data set control blocks, then adds in the buffer pool size, the working storage size, and the fixed code size to update the region size used in the DB2 start procedures. The CLIST also displays this information as messages at the terminal, as shown in Figure 38 on page 56 and in Figure 39 on page 57.

Place your own estimates in the figures provided in this section and make the indicated calculations. For your reference, the figures include default values for MVS/370 and MVS/XA, where appropriate.

Buffer Pool Size Calculation

Buffer pools are areas of main storage reserved to satisfy the buffering requirements for one or more table spaces or indexes.

The amount of buffer pool space per concurrent user should be at least 40K bytes. We recommend a value of 60K bytes or more for improved performance. Very simple SQL statements accessing small amounts of data may require less than this. Complex SQL statements accessing large amounts of data may require more than this amount. Use Figure 29 to calculate buffer pool size for your site.



Buffer Pool Calculation		XA Default	370 Default
Buffers for BP0	____ * 4K = ____	224 * 4K = 896K	56 * 4K = 224K
Buffers for BP1	+____ * 4K = ____	+ 0 * 4K = 0K	0 * 4K = 0K
Buffers for BP2	+____ * 4K = ____	+ 0 * 4K = 0K	0 * 4K = 0K
Buffers for BP32K	+____ * 32K = ____	+12 * 32K = 384K	3 * 32K= 96K
	_____	1280K	320K

Figure 29. Buffer Pool Size Calculation

EDM Pool Size Calculation

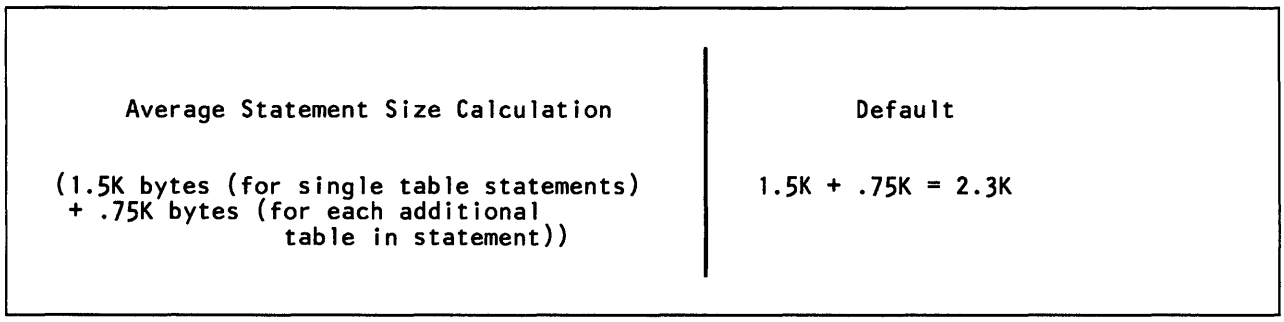
The environmental descriptor manager (EDM) pool contains active application plans and data base descriptors.

Refer to the EDMPOOL value in job DSNTIJUZ, which is generated by the CLIST. For an estimate, multiply the number of concurrent plans (one for each user), by the average statement size, by the average number of statements executed in a plan. The average plan size changes in proportion to the increase in the number and complexity of SQL statements.

To calculate the average statement size for a transaction, add 1.5K bytes for single table statements, and 0.75K bytes for each additional table in the statement.

Figure 30 shows a rough calculation for the average statement size. Because the default number of tables for each statement is 2, we added 1.5K bytes for the single table and 0.75K bytes for the additional table, which resulted in 2.25K bytes, rounded to 2.3K bytes.

If you estimated 1 table for each statement, you would multiply the number of SQL statements by 1.5K bytes; if you estimated 3 tables per statement, you would multiply the number of SQL statements by 3K bytes (1.5K bytes + 0.75K bytes + 0.75K bytes), and so forth.



Average Statement Size Calculation	Default
(1.5K bytes (for single table statements) + .75K bytes (for each additional table in statement))	1.5K + .75K = 2.3K

Figure 30. Average Statement Size Calculation

Use Figure 31 to calculate your average plan size.

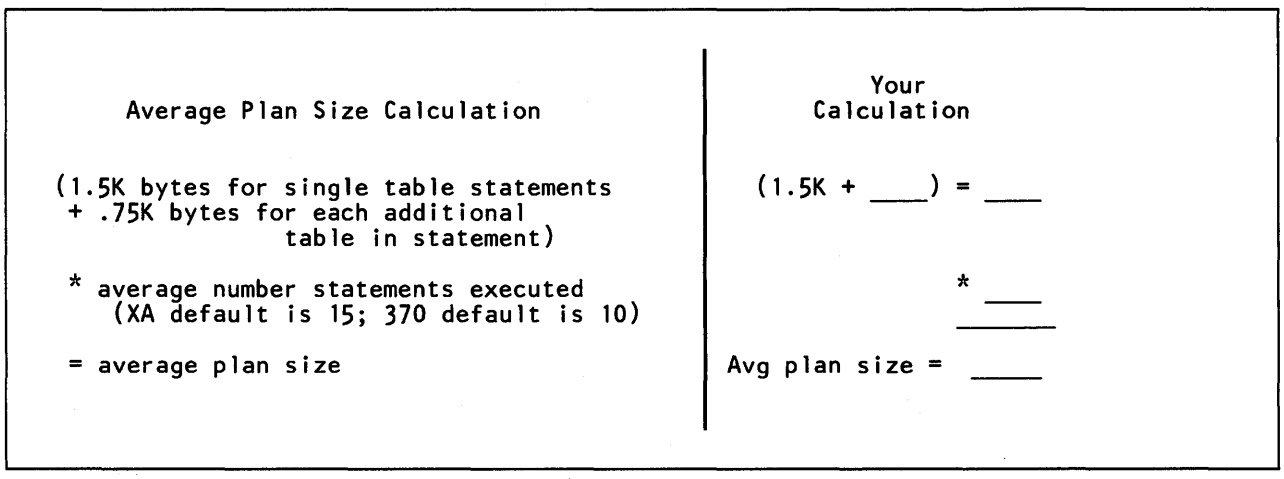


Figure 31. Average Plan Size Calculation for Your Site

The size of a QMF plan during execution will typically be 8K bytes.

After you bind a plan, you can check the size by listing the AVGSIZE and PLSIZE in the SYSPLAN catalog table. PLSIZE is the size of the base segment of the plan. AVGSIZE is the average size per section. To find the plan sizes and the total number of sections in a plan, use these SQL statements:

```
SELECT NAME, PLSIZE, AVGSIZE
       FROM SYSIBM.SYSPLAN
       ORDER BY NAME ;

SELECT PLNAME, NAME, MAX(SECTNO)
       FROM SYSIBM.SYSSTMT
       GROUP BY PLNAME, NAME
       ORDER BY PLNAME, NAME;
```

Of course, you can also use similar statements with WHERE clauses to specify the plans you want. The number of sections executed per plan can be estimated only from execution of the plans. Consequently, the amount of storage needed during execution for a plan is the base segment size, plus the size for the sections being executed.

The second part of the EDMPOOL calculation involves the space for data base descriptors (DBDs). To determine this value, multiply the number of concurrently open data bases by the average size of the data base descriptor.

The data base descriptor size is 9K bytes for the default values. The data base descriptor size depends on the number of table spaces, tables, indexes, columns, partitions, and index keys in the data base. The DSNTINST CLIST contains the algorithm for calculating the DBD size.

You can display the DBD size for your existing data bases with the DISPLAY DATABASE command.

Using Figure 32, you can estimate DBD size based on an estimate of columns per table and tables per data base for your site. The DBD sizes shown in Figure 32 reflect the larger DBDs permitted in Release 3 and assume:

- the same number of tables as table spaces
- 1 index per table (Release 2 of this book assumed 2 indexes per table)
- 2 partitions per table space
- 3 keys per index

These values are the defaults built into the CLIST and are reasonably typical for data bases. The DBD sizes are shown in kilobytes.

Columns per Table	10 Tables per Data Base	20 Tables per Data Base	30 Tables per Data Base	40 Tables per Data Base	50 Tables per Data Base
10	9K	17K	26K	34K	42K
20	10K	19K	29K	38K	47K
30	11K	21K	32K	42K	52K
40	12K	23K	34K	46K	57K
50	13K	25K	37K	50K	62K
75	15K	30K	45K	59K	74K
100	18K	35K	52K	69K	86K
200	27K	54K	81K	108K	135K
300	37K	74K	111K	147K	184K

Figure 32. DBD Sizes for Ranges of Columns and Tables

As shown in Figure 33, add your estimates for space for active plans and DBDs. Then add another 50K bytes for overhead. Finally, multiply the total by 1.25 to estimate fragmentation loss.

```
EDM Pool Calculation      yours          XA default     370 default
(Concurrent plans          ___ *           30 *           10 *
* average stmt size        ___ *         2.3K *         2.3K *
* average stmt exec)       ___ =  ___      15  = 1020K     10  =  230K
+ (concurrent data bases    ___ *           50 *            7 *
* average DBD size)        ___ =  ___     9K =  450K      9K = + 63K
+ 50K overhead                 +  50K =       + 50K =        + 50K =
                                  ___        1520K          343K
* 1.25 fragmentation loss        * 1.25 =     * 1.25 =       * 1.25 =
= EDM Pool Size                  _____        1900K          428K
```

Figure 33. EDM Pool Size Calculation

Data Set Control Block Size Calculation

To estimate the total number of data sets that are open at any time (one for each table space and one for each index), add the number of table spaces and the number of indexes concurrently in use. The number of indexes per data base is calculated as the average number of indexes per table multiplied by the average number of tables for each data base. The totals are calculated as the average number of indexes and table spaces for each data base multiplied by the number of data bases concurrently in use.

The default values of 10 table spaces and 10 tables per data base give 20 data sets per data base, because the CLIST assumes 1 index per table (1 index per table times 10 tables per data base plus 10 table spaces per data base). The maximum number of data sets that can be allocated to a task is 1635, so the maximum number of concurrently open data bases is 81 (1635 / 20) if the default of 20 data sets is used. For MVS Version 2 Release 2, the maximum number of data sets that can be allocated to a task is 3273, so the maximum number of concurrently open data bases is 163 (3273 / 20). Figure 34 shows the maximum number of concurrently open data bases for:

- Releases of MVS prior to MVS Version 2 Release 2 (including MVS/370 and earlier releases of MVS/XA)
- MVS Version 2 Release 2

Tables per Data Base	Data Sets per Data Base	Pre-MVS 2.2 Max. Concur. Open Data Bases	MVS 2.2 (and later) Max. Concur. Open Data Bases
10	20	81	163
20	40	40	81
30	60	27	54
40	80	20	40

Figure 34. Maximum Number of Concurrent Data Bases

This method of calculation ignores partitioned table spaces and partitioned index spaces. It also assumes that all data sets in the data base are open if the data base is in use. Use of CLOSE YES on table spaces and indexes can change this assumption. You may want to enter a smaller value for the number of concurrent data bases if CLOSE YES is used extensively at your site, or if typically only a few of the data sets in a data base are opened.

To calculate the main storage required for data set control blocks, place your estimate for the number of concurrently open data sets in the space provided in Figure 35, and multiply by 1.8K for MVS/XA or by 2.3K for MVS/370.

```
                Data Set Control Block Calculation

                        yours          XA          370

Number of data sets ____ *____    1000 * 1.8K   140 * 2.3K

                          = ____       = 1800K      = 322K
```

Figure 35. Data Set Control Block Size Calculation

DB2 support of MVS Version 2 Release 2 provides substantial virtual storage constraint relief. With MVS Version 2 Release 2, you have the option of moving the scheduler work area (SWA) above the 16 Mb line of MVS/XA virtual storage. In this way, you can save 600 bytes per open data set in virtual storage below the 16 Mb line. For more information refer to *MVS/Extended Architecture System Programming Library JES2 Initialization and Tuning Guide* or *MVS/Extended Architecture System Programming Library JES3 Initialization and Tuning Guide.*

Working Storage Calculation

Working storage is that portion of main storage DB2 needs in the data base services address space to hold data temporarily. To estimate the amount of working storage needed, start with 648K bytes. For MVS/XA, add 40K bytes. For MVS/370, add 72K bytes if your site uses 32K buffers. Add 40K bytes for each concurrent DB2 user, as shown in Figure 36. Place your estimates in the spaces provided and make the indicated calculations.

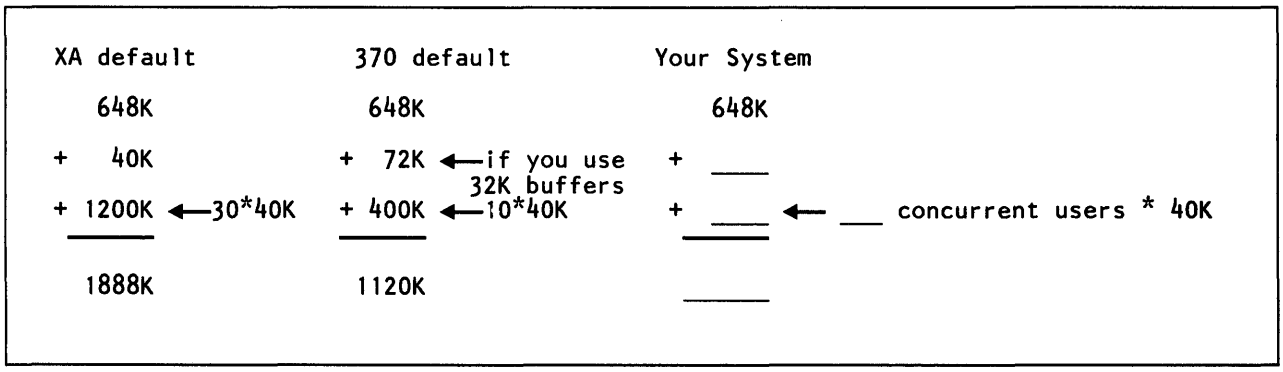


Figure 36. Working Storage Size Calculation for Data Base Services Address Space

There is a difference between dynamic SQL users and static SQL users. Dynamic SQL users need more working storage and less EDMPOOL space than static SQL users. QMF users have a very small plan in EDMPOOL, usually 4K or 8K bytes. Users of static SQL have larger plan sizes as noted above, varying from 10K to 80K bytes typically. Because the static SQL program does not need to prepare the statement or to sort as often, less working storage is required for static SQL users. Typical sites would use about 76K bytes per thread of working storage for dynamic SQL users, and 24K bytes per thread for static SQL users.

Total Main Storage Calculation for Region Size

This calculation will produce an estimate of the region size needed in the DB2 data base services address space. This calculation is important for MVS/370, but less so for MVS/XA, because most of the space is in extended private storage.

To calculate the region size for MVS/370, add the total space for buffer pools, EDM pool, data set control blocks, and working storage to that of the fixed code size.

For MVS/XA, most of the needed virtual storage is in extended private storage, including the buffer pool, the EDMPOOL, almost all of the code, and a significant amount of working storage. This is the difference between the total storage and the estimated region size in MVS/XA. The region size estimate does not include extended private storage—it includes only the data set storage size and part of the working storage. To calculate the estimated region size for MVS/XA, add 720K bytes plus 20K bytes per concurrent user plus the data set storage size.

Figure 37 shows total main storage calculation. Place your estimates in the spaces provided and make the indicated calculations.

Category	Your Size	XA Default	370 Default
Buffer size	_____	1280K	320K
EDMPOOL	_____	1900K	428K
Data set size	_____	1800K	322K
Code size	3200K	3200K	3200K
Working storage	_____	1888K	1120K
	_____	_____	_____
Total size	_____	10068K	5390K
Region size	_____	3088K	5390K
Region size (SWA above line)	_____	2488K	N/A

Figure 37. Main Storage Size Calculation

CLIST Messages for Storage Calculations

The DSNTINST CLIST will produce messages that indicate the sizes calculated.

The messages for MVS/XA show that most of the needed virtual storage is in extended private storage (including the buffer pool, the EDM pool, most of the code, and a significant amount of working storage).

The CLIST messages in Figure 38 show the default sizes for MVS/XA.

```
DSNT473I BEGINNING CHECK PHASE
DSNT485I BUFFER POOL SIZE = 1280 K
DSNT485I EDMPOOL STORAGE SIZE = 1900 K
DSNT485I DATA SET STORAGE SIZE = 1800 K
DSNT485I CODE STORAGE SIZE = 3200 K
DSNT485I WORKING STORAGE SIZE = 1888 K
DSNT486I TOTAL MAIN STORAGE = 10068 K
DSNT487I ESTIMATED REGION SIZE = 3088 K
DSNT487I ESTIMATED REGION SIZE SWA ABOVE LINE = 2488 K
```

Figure 38. CLIST Messages (Showing Default Sizes in Bytes for MVS/XA)

During the install session the DSNTINST CLIST issues the following message:

```
DSNT438I IRLM LOCK MAXIMUM SPACE = 58593 K,
         AVAILABLE SPACE = 20000 K
```

This message shows that the IRLM could request a total amount of space larger than the available space, causing an abend. The CLIST calculates the maximum space value in this message based on the values you specify for LOCKS PER USER (IRLMLKUS) on IRLM Panel 2 (DSNTIPJ), and for MAX USERS (NUMCONCR) on the Storage Sizes Panel (DSNTIPE). The CLIST calculates maximum IRLM lock space according to the formula:

```
MAX USERS x LOCKS PER USER x 200 bytes per lock
```

The CLIST assumes that the private region available for IRLM locks is 5,000K for MVS/370, and 20,000K for MVS/XA because extended private address space is used.

The CLIST messages in Figure 39 show the default sizes for MVS/370. The MVS/XA defaults assume three to five times the capacity of an MVS/370 system.

```
DSNT473I BEGINNING CHECK PHASE
DSNT485I BUFFER POOL SIZE = 320 K
DSNT485I EDMPOOL STORAGE SIZE = 428 K
DSNT485I DATA SET STORAGE SIZE = 322 K
DSNT485I CODE STORAGE SIZE = 3200 K
DSNT485I WORKING STORAGE SIZE = 1120 K
DSNT486I TOTAL MAIN STORAGE = 5390 K
DSNT487I ESTIMATED REGION SIZE = 5390 K
```

Figure 39. CLIST Messages (Showing Default Sizes in Bytes for MVS/370)

Chapter 4. Install and Migration Parameters

This chapter contains information about each parameter that you will specify when you install Release 3 or migrate to Release 3. The detailed parameter-by-parameter information will help you plan your choices before you actually install or migrate. Select your parameter values based on the information here, then turn to *IBM DATABASE 2 Install Guide*. There you will find Chapter 3, "Tailoring the Install and Migration Jobs," which provides panel summary tables, with space to record your choice for each parameter. This record of your parameter choices will greatly simplify your install or migration task. It will also be useful to you later for tuning purposes.

Both books are organized to follow the sequence in which DB2 presents the panels to you. For both install and migration, DB2 presents the panels in the following sequence:

Main Panel: DSNTIPA1

The parameters on the Main Panel (DSNTIPA1) control input to and output from the DSNTINST CLIST. On this panel you can:

1. Name the input data sets you want the CLIST to process:
   - DSN130.DSNSAMP (JCL, sample applications and data, and so forth)
   - DSN130.DSNCLIST
   - The input parameter member (DSNTIDXA or DSNTID00)
2. Name the output data sets you want the DSNTINST CLIST to produce:
   - *outprefix*.DSNSAMP (edited versions of the objects in DSN130.DSNSAMP)
   - *outprefix*.DSNTEMP (edited versions of some of the CLISTs in DSN130.DSNCLIST)
   - the output parameter member

Note that this book generally uses the default input prefix of DSN130 when constructing data set names. See Figure 13 on page 29 for an illustration of this process.

The Main Panel is the first and last panel you use during the tailoring session. When you complete the processing, this panel is displayed again. This panel uses ISPF panel services to remember certain panel values and re-displays them each time you run the CLIST. Upon completion of successful CLIST runs, the OUTPUT MEMBER NAME (OUTMEM) and DATA SET NAME PREFIX (OUTPUT) will not be re-displayed as you end your session and then re-invoke the CLIST. This will help you avoid accidentally overwriting your previous output.

The defaults for the parameters on the Main panel (DSNTIPA1) are the same for MVS/370 and MVS/XA.

```
DSNTIPA1           INSTALL, UPDATE, AND MIGRATE DB2 - MAIN PANEL
===>

Check parameters and reenter to change:

 1  INSTALL TYPE         ===> I
                                    I for new Install
                                    U for Update
                                    Data set name(member) to migrate
Enter name of your input data sets (DSNLOAD, DSNMACS, DSNSAMP, DSNCLIST):
 2  PREFIX               ===> DSN130
 3  SUFFIX               ===>

Enter to set or save panel values (by reading or writing the named members):
 4  INPUT MEMBER NAME    ===> DSNTIDXA  Enter to read old panel values
 5  OUTPUT MEMBER NAME   ===>           Enter to write new panel values

Enter name to create edited JCL and CLIST output:
 6  DATA SET NAME PREFIX ===>

 7  DSNZPARM NAME        ===> DSNZPARM  Enter name of new DB2 parameter module

PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 40. Main Panel (DSNTIPA1)

1. INSTALL TYPE (INSTYPE)

Choose one of the three modes in which the DSNTINST CLIST runs: install, update, or migrate.

- **I** to install DB2 for the first time. This is the default for the first run of the DSNTINST CLIST.
- **U** to update parameters for your current DB2 subsystem. This is the default after you make a successful install run of the CLIST.

  The update panels are used to tune DB2; they are different from the install and migration panels. They are described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

  You can also run the CLIST in either install or update mode to recheck the values you have previously specified. To do this, specify either U or I for INSTALL TYPE (INSTYPE), and specify either a new or a null DATA SET NAME PREFIX (OUTPUT) for parameter 6 on this panel. If you specify the same data set name prefix you used on your last run of the CLIST, this run will overwrite the *outprefix*.DSNSAMP and *outprefix*.DSNTEMP data sets you created previously. If you specify no data set name prefix, no new output will be created (and all previous output will remain intact). If you specify a new prefix, new output will be created (and all previous output will remain intact).
- An input parameter member name to migrate from DB2 Release 2 to DB2 Release 3.

  This is the name of the output parameter member containing the parameters you used when you last installed, updated, or migrated DB2. Specify the fully qualified data set name in the form:

  ```
  any.data.set.name(member)
  ```

  If you no longer have this data set member or the one you have is incorrect, enter the correct values for parameters on the panels; these new values will be written to the output parameter member. Alternately, if you do not have ISPF, you can create a new data set member from either a Release 2 or a Release 3 DSNTID*xx* member. Enter in this member any default values you want to use, with CLIST parameter names the first item on each line and current parameter values the last item on each line.

  Certain parameters cannot be changed during a migration. Consequently, be sure to specify them correctly in the data set member you provide. These parameters are listed in Figure 14 on page 32.

  During a migration, the default values that appear on the ISPF panels are not the IBM-supplied defaults; they are the values contained in the input parameter member you specify for INSTALL TYPE (INSTYPE). There are exceptions: parameters VOLUME SERIAL 1 - VOLUME SERIAL 6 on panel DSNTIPA2. These parameters are not copied, so you will not see your defaults in these fields.

2. PREFIX (LIBPREFX)

Specify the input prefix for the DSNLOAD, DSNMACS, DSNSAMP, and DSNCLIST libraries. The value you specify must be 18 characters or less. The default input prefix is DSN130.

This is also the prefix for several non-VSAM data sets created during the tailoring session. The DSNTINST CLIST assumes that these data sets have the same prefix and the same suffix (parameter 3 on this panel). If they do not, the CLIST cannot tailor the install and migration jobs completely and you must edit the jobs to ensure that the appropriate names are used.

A list of the data sets that use the same input prefix and suffix appears below. The default input prefix DSN130 is used in the list and generally throughout this book.

DSN130.DSNAHELP.*suffix*
DSN130.DSNAMACS.*suffix*
DSN130.DSNALOAD.*suffix*
DSN130.DSNCLIST.*suffix*
DSN130.DSNLINK.*suffix*
DSN130.DSNLOAD.*suffix*
DSN130.DSNMACS.*suffix*
DSN130.DSNSAMP.*suffix*
DSN130.DBRMLIB.DATA.*suffix*
DSN130.RUNLIB.LOAD.*suffix*
DSN130.SRCLIB.DATA.*suffix*

Three of these data sets are referenced infrequently in the install and migration jobs; consequently, changing the prefix or suffix for these three data sets is relatively easy:

DSN130.DSNALOAD appears only in job DSNTIJUZ
DSN130.DSNLINK appears only in job DSNTIJMV
DSN130.DSNMACS appears only in jobs DSNTIJUZ and DSNTEJ1

3. SUFFIX (LIBSUFFX)

Specify a suffix to append to the libraries listed in the section on PREFIX (parameter 2 on this panel). The CLIST assumes that these libraries have a common suffix. If they do not, the CLIST cannot tailor the install and migration jobs completely. You must edit the jobs to ensure that the appropriate library names are used.

Using a suffix is optional; the default value for this parameter is null. The suffix can be up to 17 characters long minus the length of the prefix. The complete data set name may not exceed 44 characters. In addition, names exceeding 8 characters must be in groups of no more than 8 characters, separated by periods.

4. INPUT MEMBER NAME (INMEM)

Specify the input parameter member that contains the parameter values you want to use as defaults during install and migration.

If you are installing DB2 for the first time, use the IBM-supplied parameter defaults. For MVS/XA, they are stored in data set DSNTIDXA. This is the default. For MVS/370, the IBM-supplied defaults are stored in data set DSNTID00. If you are installing on MVS/370, you must change DSNTIDXA to DSNTID00.

To install DB2 for both MVS/XA and MVS/370, you can use the panels twice, assessing all panels and specifying all parameters values twice. Alternatively, you can install for MVS/XA, and then install for MVS/370, using the OUTPUT MEMBER NAME from your install on MVS/XA as the INPUT MEMBER NAME for your install on MVS/370. You should reevaluate parameters whose defaults vary from MVS/XA to MVS/370, but you need not change all parameters.

If you are migrating, you specify two input parameter members: one for INPUT MEMBER NAME, and one for INSTALL TYPE (INSTYPE), parameter 1 on this panel. The member you specify for INPUT MEMBER NAME should contain the default parameter values. The member you specify for INSTALL TYPE should contain the current Release 2 parameter values at your site.

As you change the default parameters, they are saved in the output data set that you specify for OUTPUT MEMBER NAME (parameter 5 on this panel). DB2 saves the output member name from your last tailoring session and supplies it for INPUT MEMBER NAME at the beginning of your next tailoring session.

5. OUTPUT MEMBER NAME (OUTMEM)

Specify the output parameter member in which you want to save the values you enter on the ISPF panels. This member will be stored in the existing version of DSN130.DSNSAMP (not the one created by the DSNTINST CLIST). To avoid replacing any members of the existing DSN130.DSNSAMP, specify DSNTID*xx* as the value of OUTPUT MEMBER NAME, where *xx* is any alphanumeric value except 00, XA, or VB.

The default is null. In order to save the panel values you enter, you must specify a value for OUTPUT MEMBER NAME before you start the tailoring session. If you do not, the values you enter on the ISPF panels will not be saved.

Always specify a new value for OUTPUT MEMBER NAME if you resume or start a new tailoring session. DB2 will supply the OUTPUT MEMBER NAME from your current session as the INPUT MEMBER NAME for your next tailoring session. Do not specify the same member name for OUTMEM as for INMEM. Save your output parameter member.

6. DATA SET NAME PREFIX (OUTPUT)

Specify the prefix that you want to use for the output data sets produced by the DSNTINST CLIST. Using the values you enter during the tailoring session, DSNTINST creates two output data sets:

- *outprefix*.DSNTEMP, which contains a set of tailored CLISTs for DB2
- *outprefix*.DSNSAMP, which contains the tailored jobs

The default is null. You must specify a value for DATA SET NAME PREFIX before starting the tailoring session, or no tailored jobs or CLISTs are produced. *Unless you have decided to overwrite previous output data sets,* you must select a name that is different from that of any existing DSNTEMP, DSNCLIST, and DSNSAMP libraries. This means that you cannot specify the same value for this parameter as you specify for PREFIX (LIBPREFX), parameter 2 on this panel. The CLIST will allocate the output data set for you.

7. DSNZPARM NAME (OUTZPRM)

Specify the member name of the DB2 initialization parameter module. This module contains DB2 execution time options and default values. The initialization parameter module is stored in DSN130.DSNLOAD. (For more information about the contents of this module, see Appendix A, "Initialization Parameter Module (DSNZPARM)" on page 226.)

To avoid replacing any existing members of DSN130.DSNLOAD, specify DSNZPAR*x* for this parameter, where *x* is any alphanumeric character.

You use the name you specify for this parameter when you start DB2. For instance, if you specify DSNZPAR1 as the value of this parameter, you start DB2 with the following command:

```
-START DB2,PARM(DSNZPAR1)
```

The default for DSNZPARM NAME is DSNZPARM. If you accept the default, you need not specify the PARM option when you start DB2.

To create a new DSNZPARM member, you must also specify a value for parameter DATA SET NAME PREFIX (OUTPUT), parameter 6 on this panel.

Data Parameters Panel: DSNTIPA2

The VSAM data sets you create during install or migration must be cataloged on an ICF (Integrated Catalog Facility) catalog. (This is not the same as the DB2 catalog.)

DB2 uses the information you specify on this panel to define the DASD volumes for the subsystem data bases and data sets. The parameters you enter on this panel and each of the following panels are saved in the output parameter member you specified for OUTPUT MEMBER NAME (OUTMEM) on the Main Panel (DSNTIPA1).

The defaults for the parameters on the Data Parameters Panel (DSNTIPA2) are the same for MVS/370 and MVS/XA.

```
DSNTIPA2        INSTALL DB2 - DATA PARAMETERS
===>

Check parameters and reenter to change:

 1  CATALOG ALIAS        ===> DSNC130             Alias of ICF catalog for
                                                  DB2 subsystem data sets
 2  DEFINE CATALOG       ===> YES                 YES or NO
 3  VOLUME SERIAL 1      ===> DSNV01              Volume 1 for DB2 data sets
 4  VOLUME SERIAL 2      ===> DSNV02              Volume 2 for DB2 data sets
 5  VOLUME SERIAL 3      ===>                     Volume 3 for DB2 data sets
 6  VOLUME SERIAL 4      ===>                     Volume 4 for DB2 data sets
 7  VOLUME SERIAL 5      ===>                     Volume 5 for DB2 data sets
 8  VOLUME SERIAL 6      ===>                     Volume 6 for DB2 data sets
 9  PERMANENT UNIT NAME  ===> 3380                Device type for ICF catalog
                                                  and partitioned data sets
10  TEMPORARY UNIT NAME  ===> SYSDA               Device type for
                                                  temporary data sets



PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 41. Data Parameters Panel (DSNTIPA2)

1. CATALOG ALIAS (VCATALOG)

Specify the alias of the ICF catalog in which you wish to catalog the DB2 VSAM data sets created by the install process.

VSAM Data Set Cataloging Options: The install jobs catalog the DB2 VSAM data sets. This includes recovery log, subsystem, and user data sets. The catalog that defines these data sets must be created through ICF. It may be either an ICF master catalog or an ICF user catalog. We recommend an ICF user catalog because it is more flexible and easier to manipulate.

If you are installing DB2 for the first time, the default value for CATALOG ALIAS (VCATALOG) is DSNC130. You can change this value if you are installing for the

first time. You should ensure that the alias you use conforms to your site's naming conventions. If you are updating, you cannot change this parameter. To change this parameter for a previously installed DB2 subsystem, you must reinstall DB2. You cannot change this parameter during migration. You must use the same ICF catalog alias for Release 3 that you used for Release 2 because Release 3 adopts many Release 2 data sets that are already cataloged using the Release 2 ICF catalog alias.

Whether you are installing or migrating, DB2 does not require that you catalog all DB2 VSAM data sets in the same ICF catalog. This book refers to the catalog that you create during the install process as the *primary* ICF catalog. You must catalog certain data sets in this primary ICF catalog. Other data sets can be cataloged in different ICF catalogs. Still other data sets need not be cataloged at all. Figure 42 lists the ICF cataloging options that DB2 allows for each of its VSAM data sets.

DB2 VSAM Data Sets	Options
DB2 Directory (DSNDB01) DB2 Catalog (DSNDB06) Default Data Base (DSNDB04) Temporary Data Base (DSNDB07)	You must catalog these data sets in the primary ICF catalog.
Active Logs Bootstrap Data Set	You can catalog these data sets in a different ICF catalog and give them a prefix different from the VSAM data sets on the primary ICF catalog.
Archive Logs	You do not have to catalog these data sets at all. You can, however, catalog them in either the primary or an alternate ICF catalog. Cataloging archive logs simplifies DASD volume management using, for example, DFHSM.
User Table Spaces User Index Spaces	You do not have to put user table spaces and index spaces in the primary ICF catalog. You can also catalog different user table spaces and index spaces in different ICF catalogs.

Figure 42. Relationship Between the ICF Catalog and DB2 VSAM Data Sets

You must still provide any catalog connections for the active log, archive log, and bootstrap data sets that you do not catalog on the primary DB2 ICF catalog. You can provide the connection by doing one of the following:

- Adding a STEPCAT statement to the start procedure
- Defining an alias for the proper catalog
- Cataloging the data sets in the master catalog (not recommended)

Although two DB2 subsystems can be cataloged on the same ICF catalog, two DB2 subsystems must not share the same ICF catalog alias, because this is the only parameter that makes the data set names unique. If the same ICF catalog alias were used, the two DB2 subsystems could generate the same name for several different data sets.

Data Set Naming Conventions: The value you specify as the ICF catalog alias is also used as the high-level qualifier for DB2 VSAM data sets.

The data sets for the DB2 directory and catalog data bases, the default data base, and the temporary data base are all VSAM entry-sequenced data sets (ESDSs). Their data set names conform to the following format:

ddddddd.DSNDB*n*.*bbbbbbbb*.*xxxxxxxx*.I0001.A*ccc*

where:

ddddddd is the high-level qualifier. This is the value you specified for the CATALOG ALIAS (VCATALOG) parameter. DSNC130 is the default.

DSNDB*n* is a constant (C or D), identifying this as a DB2 data set where *n* is C for a cluster name or D for a data name.

bbbbbbbb is the data base name. The system data base names are:

DSNDB01—the DB2 directory data base

DSNDB04—the default data base

DSNDB06—the DB2 catalog data base

DSNDB07—the temporary data base

xxxxxxxx is the name of the individual table space or index space.

I0001.A*ccc* is used by DB2 to identify the data set. *ccc* identifies the partition number of a partitioned table space or index space, or the relative data set number of a simple table space or index space.

For example, one of the DB2 directory data sets is named:

DSNC130.DSNDB*D*.DSNDB*01*.*DBD01*.I0001.A001

Similarly, one of the DB2 catalog data sets is named:

DSNC130.DSNDB*D*.DSNDB*06*.*SYSDBASE*.I0001.A001

While all these data sets (including the DB2 catalog and directory) are usually described as VSAM data sets, they have a non-VSAM format and cannot be accessed by access method services or other VSAM record management.

2. DEFINE CATALOG (VCATSTAT)

Specify whether you want to create a new ICF catalog: YES or NO. YES is the default.

If you specify YES, a new ICF catalog is built using the ICF catalog alias you specified for CATALOG ALIAS (VCATALOG), parameter 1 on this panel. If you specify NO, the DSNTINST CLIST assumes that the ICF catalog identified by parameter 1 on this panel already exists; the CLIST does not create a new one.

You can change this value if you are installing for the first time. If you are updating, you cannot change this parameter. To change this parameter for a previously installed DB2 subsystem, you must reinstall DB2. You cannot change this parameter during migration; you must use the same ICF catalog alias for Release 3 that you used for Release 2.

3-8. VOLUME SERIAL 1-VOLUME SERIAL 6 (VOLSDAT1-VOLSDAT6)

Use these fields to specify the volume serial numbers for the data sets defined by install or migration.

You may specify up to six volumes. You must specify at least one. Specifying more volumes helps recovery and performance. If you specify fewer than six volumes, data is combined on them. If you specify six volumes, data is distributed as follows:

- VOLSDAT1 is used for non-VSAM data. It is used for *outprefix*.DSNSAMP and *outprefix*.DSNTEMP, the data sets generated by the DSNTINST CLIST. It is also used for DSN130.DBRMLIB.DATA, DSN130.RUNLIB.LOAD, and DSN130.SRCLIB.DATA. The default is DSNV01.
- VOLSDAT2 is used for the temporary data sets, the default and sample storage group, and the ICF catalog (if a new one is created). The default is DSNV02.

  You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.
- VOLSDAT3 is used for the DB2 catalog and directory. If you do not specify a value, the volume serial you specified for VOLSDAT1 is used for the DB2 catalog and directory.

  You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.
- VOLSDAT4 is used for DB2 catalog indexes and directory indexes. If you do not specify a value, the volume serial you specified for VOLSDAT2 is used for the DB2 catalog indexes and directory indexes.

  You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.
- VOLSDAT5 is used for the first copy of the active log and the second copy of the BSDS. If you do not specify a value, the volume serial you specified for VOLSDAT3 is used for the first copy of the active log and the second copy of the BSDS.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

- VOLSDAT6 is used for the second copy of the active log and the first copy of the BSDS. If you do not specify a value, the volume serial you specified for VOLSDAT4 is used for the second copy of the active log and the first copy of the BSDS.

  You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

  If you accept the default of dual active logging, specify different volume serials for VOLUME SERIAL 5 and VOLUME SERIAL 6. This places the active logs on different DASD devices.

9. PERMANENT UNIT NAME (VOLSDEVT)

Specify the device type or unit name that will be used to allocate the following data sets:

- ICF catalog
- DSN130.DBRMLIB.DATA
- DSN130.RUNLIB.LOAD
- DSN130.SRCLIB.DATA
- The two data sets generated by the DSNTINST CLIST:
  - *outprefix*.DSNTEMP
  - *outprefix*.DSNSAMP

The value identifies a direct access or disk unit name, because it is used for partitioned data sets and the ICF catalog. If you want to use different device types for different data sets, edit the install or migration jobs after you complete the tailoring session. Some common device types are 3330, 3340, 3350, 3375, and 3380. The default is 3380.

If you change this parameter during a migration to DB2 Release 3, it will not affect the ICF catalog, DB2 catalog, directory, or logs. The new value will be used for data sets created during migration.

10. TEMPORARY UNIT NAME (VOLTDEVT)

Specify the device type or unit name that will be used to allocate temporary data sets. This is the direct access or disk unit name used for the precompiler. The default is SYSDA.

Sizes Panel: DSNTIPD

DB2 uses the information you specify on this panel to establish the size of the DB2 catalog, directory, temporary data base, and log data sets.

The values you supply on this panel are only estimates. They do not preclude any application or user from exceeding these estimates, within reasonable limits. For example, if you specify 50 data bases you can still have more than 50. However, if you exceed the value by too much, you may run out of main storage or use many secondary extents.

The DSNTINST CLIST contains formulas that calculate the space needed for each catalog data set and the indexes each data set requires. Data entered on this panel is used as input to these formulas. Use integers for these values; do not specify fractions.

Many of the parameters on this panel affect the value of the EDMPOOL parameter in macro DSN6SPRM.

The defaults for the parameters on Sizes Panel (DSNTIPD) are different for MVS/370 and MVS/XA. These defaults correspond to the storage sizes for the site models discussed in Chapter 3, "Estimating DB2 Storage Needs" on page 36. The MVS/370 defaults correspond to the small site model; the MVS/XA defaults correspond to the medium site model. Large sites should increase the default values according to their needs.

The capacity you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might, during rollback and restart, need to access data in the archive log. Two parameters on Panel DSNTIPD affect the capacity of the active log:

- ARCHIVE LOG FREQ (NUMHRARC) provides an estimate of how often active logs data sets are to be copied to the archive log.
- UPDATE RATE (NUMCOMHR) estimates the number of data base changes (updates, inserts, deletes) expected per hour.

In each case, increasing the value you specify for the parameter increases the capacity of the active log. See "Determining the Size of Active Logs" on page 199 for a complete list of factors affecting the active and archive logs.

If you are migrating, DB2 Release 3 adopts your Release 2 DB2 catalog, directory, temporary data bases, BSDS, and active and archive logs. Consequently, during migration, you cannot change any of the parameters that affect the characteristics of these objects. This means that during migration you cannot change any of the parameters on the Sizes Panel (DSNTIPD) except:

- VIEWS (NUMVIEWS) is the average number of views per table.

See "Migrating DB2" on page 30 for a complete list of parameters that cannot be changed during migration.

You can alter the characteristics of the DB2 catalog, directory, temporary data bases, BSDS, and active and archive logs by using the update process after you complete the migration process. See Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

```
DSNTIPD         INSTALL DB2 - SIZES
===>

Check numbers and reenter to change:

 1  DATA BASES        ===> 200        In this subsystem
 2  TABLES            ===> 10         Per data base (average)
 3  COLUMNS           ===> 10         Per table (average)
 4  VIEWS             ===> 3          Per table (average)
 5  TABLE SPACES      ===> 10         Per data base (average)
 6  PLANS             ===> 200        In this subsystem
 7  PLAN STATEMENTS   ===> 30         SQL statements per plan (average)
 8  EXECUTED STMTS    ===> 15         SQL statements executed (average)
 9  TABLES IN STMT    ===> 2          Tables per SQL statement (average)
10  TEMPORARY 4K      ===> 16M        Bytes of 4K-page work space
11  TEMPORARY 32K     ===> 4M         Bytes of 32K-page work space
12  ARCHIVE LOG FREQ  ===> 24         Hours per archive run
13  UPDATE RATE       ===> 3600       Updates, inserts, and deletes per hour



PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 43. Sizes Panel: DSNTIPD

1. DATA BASES (NUMDATAB)

Estimate the number of user data bases in your subsystem. The default is 50 for MVS/370 and 200 for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

2. TABLES (NUMTABLE)

Estimate the average number of tables per data base in your subsystem.

Reasonable values for this parameter range from 1 to 180. The value you specify depends on the average number of columns your tables will have, and whether you will use indexes and partitions. If you plan to have tables with 20 to 40 columns each, the value you should use for this parameter typically ranges from 60 to 80. If you plan to have tables with 1 to 10 columns, and no indexes or partitions, the value you specify can be higher.

The default is 10.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

3. COLUMNS (NUMCOLUM)

Estimate the average number of columns per table in your subsystem.

The maximum value for this parameter is 300; the minimum is 1. The default is 10.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

4. VIEWS (NUMVIEWS)

Estimate the average number of views per table in your subsystem. The default is 3.

5. TABLE SPACES (NUMTABSP)

Estimate the average number of table spaces per data base in your subsystem. The default is 10.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

6. PLANS (NUMPLANS)

Estimate the number of application plans in your subsystem. Each program requires a separate application plan. The default is 100 for MVS/370 and 200 for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

7. PLAN STATEMENTS (NUMSTMTS)

Estimate the average number of SQL statements per application plan. The maximum value is 32,000; The default is 20 for MVS/370 and 30 for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

8. EXECUTED STMTS (NUMSTMTE)

Estimate the average number of SQL statements executed per plan.

For a plan that specifies or defaults to ACQUIRE(USE) on the BIND statement, the number of SQL statements executed may be smaller than the total number of SQL statements. The maximum value is 32,000; The default is 10 for MVS/370 and 15 for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

9. TABLES IN STMT (NUMSTMTL)

Estimate the average number of tables used per SQL statement.

Certain types of statements require additional tables to be referenced; this includes statements that use joins, unions, subselect clauses, the DISTINCT keyword, and/or the ORDER BY keyword. Consider how often you plan to use these types of statements, and specify an appropriate value for this parameter. The maximum value is 16; the default is 2.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

10. TEMPORARY 4K (NUMTEMP1)

Specify the size of the temporary data base for 4K-byte page data bases.

The temporary data base (DSNDB07) is used as temporary space for SQL statements that require working storage. The SQL statements that always require working storage are:

- GROUP BY or HAVING (without index)
- ORDER BY (without index)
- DISTINCT (without index)
- UNION (except UNION ALL)
- EXISTS (subselect)
- IN (subselect)
- ANY (subselect)
- ALL (subselect)
- Some joins

The ISPF tailoring session allows you to create two table spaces within the temporary data base: one for tables spaces that require 4K-byte buffering, and one for table spaces that require 32K-byte buffering. The names of the data sets for these table spaces are:

- DSNC130.DSNDBD.DSNDB07.DSNTMP01.I0001.A001
- DSNC130.DSNDBD.DSNDB07.DSNTMP02.I0001.A001

All DB2 users share these table spaces. Utilities cannot be used on these table spaces.

You can control the size of these table spaces using the TEMPORARY 4K (NUMTEMP1) and TEMPORARY 32K (NUMTEMP2) parameters. TEMPORARY 4K controls the size of the table space for 4K-byte buffering.

You can use the abbreviations K and M for multiples of 1024 and 1048576 bytes, respectively. The default is 4M for MVS/370 and 16M for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

You can create additional temporary table spaces at any time except during migration. This improves DB2 performance by reducing device contention among applications that require working storage. In addition, you can concatenate temporary table spaces. This allows you to support large temporary files. For information about creating additional temporary table spaces, see "Allocate Additional Temporary Table Spaces" on page 189.

11. TEMPORARY 32K (NUMTEMP2)

Specify the size of the temporary data base for 32K-byte page data bases. (See the description under TEMPORARY 4K, parameter 10 on this panel.)

You can use the abbreviations K and M for multiples of 1024 and 1048576, respectively. The default is 1M for MVS/370 and 4M for MVS/XA.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration

process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

If you do not need working storage for 32K-byte buffering, specify 0 for parameter TEMPORARY 32K. If you specify 0 for this parameter, you must also specify 0 for the MIN BP32K BUFFERS (BUFMIN32) and MAX BP32K BUFFERS (BUFMAX32) fields on the Storage Sizes Panel (DSNTIPE).

12. ARCHIVE LOG FREQ (NUMHRARC)

Estimate the interval in hours at which the active log is off-loaded to the archive log. If you accept the default value of 24, the active log is off-loaded approximately once each day.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

13. UPDATE RATE (NUMCOMHR)

Estimate the number of inserts, updates, and deletes expected per hour in your subsystem. The default value is 1800 for MVS/370 and 3600 for MVS/XA.

The size calculations in the DSNTINST CLIST assume that approximately 400 bytes of data are logged for each insert, update, and delete. The amount of data logged for these changes may be different at your site. Therefore, you should update this parameter after you gain some experience with DB2 and have a better idea of how many bytes of data are logged for each change.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

Storage Sizes Panel: DSNTIPE

DB2 uses the information you specify on this panel to calculate main storage sizes for MVS/370 and MVS/XA.

The DSNTINST CLIST contains formulas that calculate the space needed. The values entered on this panel are used as input to these formulas.

The defaults for the parameters on this panel are different for MVS/370 and MVS/XA.

```
DSNTIPE         INSTALL DB2 - STORAGE SIZES
===>

Check numbers and reenter to change:

 1  DATA BASES         ===> 50         Concurrently in use
 2  MAX USERS          ===> 30         Concurrently running in DB2
 3  MAX TSO CONNECT    ===> 100        Users on QMF or in DSN command
 4  MAX BATCH CONNECT  ===> 20         Users in DSN command or utilities
 5  MIN BP0 BUFFERS    ===> 224        Minimum number of buffers in BP0
 6  MAX BP0 BUFFERS    ===> 224        Maximum number of buffers in BP0
 7  MIN BP1 BUFFERS    ===> 0          Minimum number of buffers in BP1
 8  MAX BP1 BUFFERS    ===> 0          Maximum number of buffers in BP1
 9  MIN BP2 BUFFERS    ===> 0          Minimum number of buffers in BP2
10  MAX BP2 BUFFERS    ===> 0          Maximum number of buffers in BP2
11  MIN BP32K BUFFERS  ===> 12         Minimum number of buffers in BP32K
12  MAX BP32K BUFFERS  ===> 12         Maximum number of buffers in BP32K



PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 44. Storage Sizes Panel: DSNTIPE

1. DATA BASES (NUMCONDB)

Specify the maximum number of data bases that can be open at one time.

The default value is 7 for MVS/370 and 50 for MVS/XA. The maximum acceptable value for both MVS/370 and MVS/XA is 800.

This number is affected by the CLOSE YES or CLOSE NO specification on the CREATE TABLESPACE and CREATE INDEX statements as well as by START and STOP commands. For instance, you may want to specify a smaller value for this parameter if your site uses CLOSE YES extensively.

2. MAX USERS (NUMCONCR)

Specify the maximum number of threads that can be allocated concurrently. Count the following as separate users:

- Each TSO user (whether running a DSN command, or a DB2 request from QMF)
- Each batch job (whether running a DSN command or a DB2 utility)
- Each IMS/VS region that can access DB2
- Each CICS transaction that can access DB2
- Each task connected to DB2 through the call attachment facility

When the number of users attempting to access DB2 exceeds the number you specify, excess plan allocation requests are queued.

The default value is 10 for MVS/370 and 30 for MVS/XA. The maximum acceptable value is 200 for both MVS/370 and MVS/XA.

This parameter also sets the CTHREAD parameter of macro DSN6SYSP.

3. MAX TSO CONNECT (NUMCONTS)

Specify the maximum number of users allowed to access DB2 from TSO foreground at the same time. Count each of the following as a separate user:

- Each TSO user executing a DSN command
- Each QMF user (whether executing a DB2 request or not)
- Each task connected to DB2 through the call attachment facility

When the number of TSO users attempting to access DB2 exceeds the number you specify, excess connection requests are rejected.

The default value is 40 for MVS/370 and 100 for MVS/XA. The maximum acceptable value is 1000 for both MVS/370 and MVS/XA.

The install and migration panels do not have a parameter for controlling the maximum concurrent connections for IMS/VS and CICS. You can control those limits by using IMS/VS and CICS facilities.

This parameter also sets the IDFORE parameter of macro DSN6SYSP.

4. MAX BATCH CONNECT (NUMCONBT)

Specify the maximum number of concurrent connections to DB2 from batch. Count each batch job (using QMF, the DSN command, task connected to DB2 through the call attachment facility, or running a utility) as a separate connection.

Requests to access DB2 by batch jobs that exceed this limit are rejected. The default value is 10 for MVS/370 and 20 for MVS/XA. The maximum acceptable value is 100 for both MVS/370 and MVS/XA.

This parameter also sets the IDBACK parameter of macro DSN6SYSP.

5. MIN BP0 BUFFERS (BUFMIN00)

The minimum number of buffers for buffer pool BP0. For MVS/XA, specify a value from 14 to 500,000; the default value is 224. For MVS/370, specify a value from 14 to 16,000; the default value is 56.

Parameters 5 through 12 on this panel allow you to specify the minimum and maximum number of buffers in each of the buffer pools. For parameters 5 through 10, each buffer is 4K bytes in size; for parameters 11 and 12, each buffer is 32K bytes in size. The values specified for these parameters will affect the amount of MVS virtual storage used in the system. For more information, see "Improving Virtual Storage Utilization" on page 193.

These parameters set the BUFFER parameter of macro DSN6SPRM.

6. MAX BP0 BUFFERS (BUFMAX00)

The maximum number of buffers for buffer pool BP0. For MVS/XA, specify a value from 14 to 500,000; the default value is 224. For MVS/370, specify a value from 14 to 16,000; the default value is 56.

7. MIN BP1 BUFFERS (BUFMIN01)

The minimum number of buffers for buffer pool BP1. For MVS/XA, specify a value from 0 to 500,000; the default value is 0. For MVS/370, specify a value from 0 to 16,000; the default value is 0.

8. MAX BP1 BUFFERS (BUFMAX01)

The maximum number of buffers for buffer pool BP1. For MVS/XA, specify a value from 0 to 500,000; the default value is 0. For MVS/370, specify a value from 0 to 16,000; the default value is 0.

9. MIN BP2 BUFFERS (BUFMIN02)

The minimum number of buffers for buffer pool BP2. For MVS/XA, specify a value from 0 to 500,000; the default value is 0. For MVS/370, specify a value from 0 to 16,000; the default value is 0.

10. MAX BP2 BUFFERS (BUFMAX02)

The maximum number of buffers for buffer pool BP2. For MVS/XA, specify a value from 0 to 500,000; the default value is 0. For MVS/370, specify a value from 0 to 16,000; the default value is 0.

11. MIN BP32K BUFFERS (BUFMIN32)

The minimum number of buffers for buffer pool BP32K. For MVS/XA, specify a value from 0 to 62,500; the default value is 12. For MVS/370, specify a value from 0 to 16,000; the default value is 3.

If you do not need working storage for 32K-byte buffering, specify 0 for parameter MIN BP32K BUFFERS (BUFMIN32). You must also specify 0 for MAX BP32K BUFFERS (BUFMAX32) on this panel, as well as 0 for parameter TEMPORARY 32K on the Sizes Panel (DSNTIPD).

12. MAX BP32K BUFFERS (BUFMAX32)

The maximum number of buffers for buffer pool BP32K. For MVS/XA, specify a value from 0 to 62,500; the default value is 12. For MVS/370, specify a value from 0 to 16,000; the default value is 3.

If you do not need working storage for 32K-byte buffering, specify 0 for parameter MAX BP32K BUFFERS (BUFMAX32). You must also specify 0 for MIN BP32K BUFFERS (BUFMIN32) on this panel, as well as 0 for parameter TEMPORARY 32K on the Sizes Panel (DSNTIPD).

Operator Functions Panel: DSNTIPO

DB2 uses the information you specify on this panel to set various operator functions, such as write-to-operator route codes, collection of statistics and accounting data, and checkpoint frequency.

The capacity you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might, during rollback and restart, need to access data in the archive log. One parameter on this panel affects the capacity of the active log:

- CHECKPOINT FREQ (OPCHKFRQ) determines the number of log records written between checkpoints.

Increasing the value you specify for the parameter increases the capacity of the active log. See "Determining the Size of Active Logs" on page 199 a complete list of factors affecting the active and archive logs.

Some of the defaults for the parameters on the Operator Functions Panel (DSNTIPO) are different for MVS/370 and MVS/XA.

```
DSNTIPO         INSTALL DB2 - OPERATOR FUNCTIONS
===>

Enter data below:

 1  WTO ROUTE CODES     ===> 1
                                  Routing codes for WTORs
 2  TRACE SIZE          ===> 64K  Trace table size in bytes
 3  TRACE AUTO START    ===> YES  Start trace automatically. YES or NO
 4  SMF STATISTICS      ===> NO   Gather SMF statistics information
 5  SMF ACCOUNTING      ===> NO   Gather SMF accounting information
 6  CHECKPOINT FREQ     ===> 1000 Number of log records per checkpoint
 7  MVS LEVEL           ===> XA   370 or XA
 8  STATISTICS TIME     ===> 30   Time interval in minutes (1-1440)
 9  WAIT FOR RECALL     ===> NO   Open processing to wait for recall




PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 45. Operator Functions Panel: DSNTIPO

1. WTO ROUTE CODES (OPROUTCD)

Specify the MVS console route codes assigned to messages that are not solicited from a specific console. The default is 1 for both MVS/370 and MVS/XA.

You must specify at least one number. Separate numbers in the list by commas only, not by blanks. For example:

```
1,3,5,7,9,10,11
```

This parameter also sets the ROUTCDE parameter of macro DSN6SYSP.

For more information on routing codes, refer to *MVS/Extended Architecture Message Library: Routing and Descriptor Codes.*

2. TRACE SIZE (OPTRCSIZ)

Specify the size, in bytes, of the trace table.

The global trace facility allows tracing of DB2 events in a wraparound table.

Calculate this value taking into account that most trace records require 32-byte entries. Events with more than three data items require 64-byte entries.

You can use the abbreviation K for multiples of 1024 bytes. The actual value is rounded up to a multiple of 4096 bytes. If you use 50K, for example, the actual table size will be 52K bytes.

The default is 32K for MVS/370 and 64K for and MVS/XA. The minimum value for this parameter is 4K; the maximum is 396K.

This parameter also sets the TRACTBL parameter of macro DSN6SYSP.

3. TRACE AUTO START (OPTRCAUT)

Specify whether DB2 will start collection of trace data automatically: YES or NO. The default is YES for both MVS/370 and MVS/XA.

After you complete the tailoring session, you can control trace activity with the -START TRACE, -DISPLAY TRACE, and -STOP TRACE commands. For the syntax of these commands, see *IBM DATABASE 2 SQL Reference.* For information about how to use these commands to monitor DB2, see Chapter 7, "Monitoring DB2" on page 167. For information about how to use these commands to diagnose DB2 problems, see *IBM DATABASE 2 Diagnosis Reference Volume 1.*

This parameter also sets the TRACSTR parameter of macro DSN6SYSP.

4. SMF STATISTICS (OPSMFSTA)

Specify whether DB2 will send statistics data to SMF: YES or NO. The default is NO for both MVS/370 and MVS/XA.

If you specify YES, you may also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write these records. For more information see *IBM DATABASE 2 Install Guide* and *MVS/370 System Programming Library: System Macros and Facilities Volume 2*.

This parameter also sets the SMFSTAT parameter of macro DSN6SYSP.

5. SMF ACCOUNTING (OPSMFACT)

Specify whether DB2 will send accounting data to SMF: YES or NO. The default is NO for both MVS/370 and MVS/XA.

If you specify YES, you may also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write these records. For more information see *IBM DATABASE 2 Install Guide* and *MVS/370 System Programming Library: System Macros and Facilities Volume 2*.

This parameter also sets the SMFACCT parameter of macro DSN6SYSP.

6. CHECKPOINT FREQ (OPCHKFRQ)

Specify the number of log records that DB2 will write between the start of successive checkpoints.

If you accept the default, which is 1000 for both MVS/370 and MVS/XA, DB2 will start a new checkpoint every time 1000 log records have been written.

The number of log records DB2 writes varies based on several factors; these include application mix and complexity, processor and DASD configurations and speeds, and acceptable performance/restart time trade off.

The minimum value for this parameter is 200; the maximum is 500000.

This parameter also sets the LOGLOAD parameter of macro DSN6SYSP.

7. MVS LEVEL (OPMVSLV)

Specify the MVS level on which you will use DB2: 370 or XA. The default is 370 for MVS/370 and XA for MVS/XA. Your MVS level determines the size information and defaults used within the initialization macros.

This parameter also sets parameter MVS of macro DSN6ENV.

8. STATISTICS TIME (OPSTATIM)

Specify the the time interval, in minutes, between statistics collection. Statistics records are written approximately at the end of this interval. Specify a value from 1 to 1440. The default is 30 for both MVS/370 and MVS/XA.

9. WAIT FOR RECALL (OPRECALL)

Specify whether to wait for automatic recall of migrated data sets at data set open time, using Data Facility Hierarchical Storage Manager (DFHSM). NO indicates that if a DB2 table or index space has been migrated, DB2 will initiate the recall and indicate resource unavailable at data set open time. YES indicates that DB2 will wait for the automatic recall to be performed at data set open time. The default is NO for both MVS/370 and MVS/XA.

If you specify YES for this parameter and a data set is being recalled, all subsequent data set open and close operations must wait until the recall of the migrated data set is complete. Since recall may take several seconds, this could have a significant effect on system response time.

Never specify YES for this parameter if data sets are migrated to tape, as the recall would require waiting for the tape to be mounted.

Note that the DB2 directory, catalog, and associated indexes should not be placed on volumes that are managed by DFHSM.

This parameter also sets the RECALWT parameter of macro DSN6SPRM.

Application Programming Defaults Panel: DSNTIPF

DB2 uses the information you specify on this panel to set application programming defaults. The DSNTINST CLIST builds JCL (Job DSNTIJUZ) which assembles the values from this panel, the subsystem name from DSNTIPM, and the current DB2 release level into a data-only load module called DSNHDECP. DSNDDECP resides in DSN130.DSNMACS (the DB2 macro library) and maps the DSNHDECP load module, which resides in DSN130.DSNLOAD (a DB2 load-module library). Applications can load and access DSNHDECP. When modifying data in DSNHDECP, do so only in the approved UPDATE fashion. Do not ZAP the data in DSNHDECP. For more information, see *IBM DATABASE 2 Advanced Application Programming Guide.*

DSNHDECP contains the current DB2 release level, the default programming language, the default subsystem identifier, and other fields that may be of interest to application programmers.

The values you specify on this panel are used as default values by the program preparation panels, the program preparation CLIST (DSNH), and the precompiler. They may also be used as defaults by other programs, such as QMF (Query Management Facility).

The defaults for the parameters on this panel are the same for MVS/370 and MVS/XA.

```
DSNTIPF         INSTALL DB2 - APPLICATION PROGRAMMING DEFAULTS
===>

Enter data below:

 1  LANGUAGE DEFAULT      ===> COBOL    COBOL, COB2, PLI, FORTRAN, ASM, ASMH
 2  DECIMAL POINT IS      ===> .        . or ,
 3  MINIMUM DIVIDE SCALE  ===> NO       NO or YES for a minimum of 3 digits
                                        to right of decimal after division
 4  STRING DELIMITER      ===> DEFAULT  DEFAULT, " or '
 5  SQL STRING DELIMITER  ===> DEFAULT  DEFAULT, " or '
 6  MIXED DATA            ===> NO       NO or YES for mixed DBCS data
 7  CHARACTER SET         ===> ALPHANUM ALPHANUM or KATAKANA
 8  DATE FORMAT           ===> ISO      ISO, JIS, USA, EUR, LOCAL
 9  TIME FORMAT           ===> ISO      ISO, JIS, USA, EUR, LOCAL
10  LOCAL DATE LENGTH     ===> 0        0 (no exit), or value between 10 and 254
11  LOCAL TIME LENGTH     ===> 0        0 (no exit), or value between  8 and 254



PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 46. Application Programming Defaults Panel: DSNTIPF

1. LANGUAGE DEFAULT (DEFLANG)

Specify the default programming language for your site. The default value for this parameter is COBOL. You can specify any of the following:

ASM for Assembler F
ASMH for Assembler H
COBOL for COBOL
COB2 for VS COBOL II
FORTRAN for Fortran
PLI for PL/I

2. DECIMAL POINT IS (DECPOINT)

Specify whether the decimal point for numbers is the comma (,) or the period (.). The default is the period (.). Some nations customarily signify the number "one and one-half", for instance, as "1.5"; other nations use "1,5" for the same value.

This parameter is used for dynamic SQL. It also specifies the default precompiler option (PERIOD or COMMA) for COBOL and VS COBOL II programs. It is not used in or supported by other languages.

If you are migrating, consider retaining the value you specified for Release 2 for compatibility. You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. You should investigate the impact of an update carefully.

3. MINIMUM DIVIDE SCALE (DECDIV3)

Specify YES if at least three digits should be displayed to the right of the decimal point after decimal division. Certain accounting applications may choose this option. If the default NO is accepted, the rules for decimal division in SQL are followed. See *IBM DATABASE 2 SQL Reference* for more information about decimal division in SQL.

4. STRING DELIMITER (DEFSTRNG)

Specify whether the string delimiter for COBOL or VS COBOL II is to be an apostrophe (') or a quotation mark ("). Specify one of the following:

DEFAULT to make the delimiter for strings in COBOL or VS COBOL II programs a quotation mark (").

" to make the host language string delimiter a quotation mark

' to make the host language string delimiter an apostrophe

This option is effective only for COBOL and VS COBOL II languages. It is not used by dynamic SQL. See SQL STRING DELIMITER (DEFSQSTR).

If you are migrating, consider retaining the value you specified for Release 2 for compatibility. You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. You should investigate the impact of an update carefully.

5. SQL STRING DELIMITER (DEFSQSTR)

Specify whether the SQL string delimiter is an apostrophe (') or quotation mark ("). This delimiter is used to delimit strings in dynamic SQL and in SQL statements within COBOL or VS COBOL II programs. Specify one of the following:

DEFAULT to make the delimiter for strings in SQL statements within COBOL or VS COBOL II programs a quotation mark ("), and the string delimiter in dynamic SQL an apostrophe

" to make the string delimiter in SQL statements a quotation mark

' to make the string delimiter in SQL statements an apostrophe

This option is effective only for COBOL and VS COBOL II languages. It also sets the value for dynamic SQL. The value you specify is used by the program preparation panels (as the APOSTSQL/QUOTESQL default), the DSNH CLIST, and the precompiler.

If you are migrating, consider retaining the value you specified for Release 2 for compatibility. You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. You should investigate the impact of an update carefully. You may need to change queries in data sets and in QMF. Use Figure 47 to determine the values you should specify.

PURPOSE	STRING DELIMITER	SQL STRING DELIMITER
Compatibility with Release 1	DEFAULT	DEFAULT
Force APOST default (even in COBOL and VS COBOL II) and provide a default similar to APOST in SQL/DS	'	'
Change dynamic query string delimiter to the quote. Helpful if you use COBOL or VS COBOL II with the quote option— allows queries to be tested with dynamic SQL and moved into the program more easily.	"	"
Compatibility with the SQL/DS QUOTE option	"	'

Figure 47. Values for STRING DELIMITER and SQL STRING DELIMITER

Figure 48 provides guidance for selecting the values of STRING DELIMITER (DEFSTRING) and SQL STRING DELIMITER (DEFSQSTR) depending on the string delimiters you currently use in dynamic SQL, COBOL, VS COBOL II, and SQL embedded in either COBOL. The SQL escape character is a quotation mark when the SQL string delimiter is an apostrophe; it is an apostrophe when the SQL string delimiter is a quotation mark.

SPECIFY: STRING DELIMITER	SQL STRING DELIMITER	TO GET: DYNAMIC SQL DELIMITER	COBOL DELIMITER	SQL DELIMITER IN COBOL
DEFAULT	DEFAULT	'	"	"
'	'	'	'	'
"	"	"	"	"
"	'	'	"	'

Figure 48. STRING DELIMITER and SQL STRING DELIMITER Combinations

Figure 49 shows how the precompiler options interact with the values you specify for STRING DELIMITER and SQL STRING DELIMITER. This chart allows you to select useful combinations so that you do not have to set both sets of parameters (precompiler and install parameters). The last three columns show the precompiler options in effect for three different parameter settings.

SPECIFY: STRING DELIMITER	SQL STRING DELIMITER	TO GET: NO PARM	PARM=APOST	PARM=QUOTE
DEFAULT	DEFAULT	QUOTE QUOTESQL	APOST APOSTSQL	QUOTE QUOTESQL
'	'	APOST APOSTSQL	APOST APOSTSQL	QUOTE QUOTESQL
"	"	QUOTE QUOTESQL	APOST APOSTSQL	QUOTE QUOTESQL
"	'	QUOTE APOSTSQL	APOST APOSTSQL	QUOTE QUOTESQL

Figure 49. String Delimiter and Precompiler Option Combinations

6. MIXED DATA (DEFMIXED)

The major use for this parameter is to control DB2 recognition of Katakana characters. Specify whether the characters X'0E' and X'0F' are to have a special meaning as the shift-out and shift-in controls for the double-byte character set (DBCS): YES or NO.

If you accept the default value NO, the characters may be used in a string without being used as control characters. If you specify YES, DB2 treats X'0E' and X'0F' as control characters and assumes that all character data may be mixed DBCS.

If you are migrating, consider retaining the value you specified for Release 2 for compatibility. You can update this parameter using the process described in

Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. You should investigate the impact of an update carefully.

7. CHARACTER SET (DEFCHARS)

Specify whether DB2 will accept only the alphanumeric character set or accept also the Katakana set without delimiters.

Specify one of the following:

ALPHANUM (the default) for the alphanumeric character set of 10 digits and 26 letters. Lowercase letters are folded to uppercase.

KATAKANA for the Katakana character set. This value allows tables and columns to have Katakana names without use of the SQL escape characters (that is, quotation marks or apostrophes, depending on what you specified for STRING DELIMITER (DEFSTRING), parameter 4 on this panel, and the APOST | QUOTE precompiler option). Characters are not folded from lowercase to uppercase when you specify this value.

If you are migrating, consider retaining the value you specified for Release 2 for compatibility.

You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. You should investigate the impact of an update carefully.

8. DATE FORMAT (DEFDATE)

Specify which of the following formats will be used as a default output format to represent dates:

ISO for yyyy-mm-dd
USA for mm/dd/yyyy
EUR for dd.mm.yyyy
JIS for yyyy-mm-dd
LOCAL for user-defined format

Figure 50 on page 88 illustrates the four IBM-supplied formats and provides examples.

Format Name	Abbreviation	Format	Example
International Standards Organization	ISO	yyyy-mm-dd	1986-12-25
IBM USA standard	USA	mm/dd/yyyy	12/25/1986
IBM European standard	EUR	dd.mm.yyyy	25.12.1986
Japanese Industrial Standard Christian Era	JIS	yyyy-mm-dd	1986-12-25

Figure 50. IBM-Supplied Date Formats

Note that DB2 can accept a date in any format as input. DB2 will interpret the input date based upon its punctuation, then provide date output in the format you specify for this parameter.

ISO is the default. If you want your own date format to be the default, enter LOCAL for this parameter. If you specify LOCAL, you must provide a date exit routine to perform date formatting.

You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. Updating this parameter can affect how your applications run; investigate the impact of an update carefully.

9. TIME FORMAT (DEFTIME)

Specify which of the following formats will be used as default output to represent times:

ISO for hh.mm.ss
USA for hh:mm xM or hh xM
EUR for hh.mm.ss
JIS for hh:mm:ss
LOCAL for user-defined form

Figure 51 illustrates the four IBM-supplied formats and provides examples.

Format Name	Abbreviation	Format	Example
International Standards Organization	ISO	hh.mm.ss	13.30.05
IBM USA standard	USA	hh:mm AM or hh PM	1:30 PM or 1 PM
IBM European standard	EUR	hh.mm.ss	13.30.05
Japanese Industrial Standard Christian Era	JIS	hh:mm:ss	13:30:05

Figure 51. IBM-Supplied Time Formats

Note that DB2 can accept a time in any format as input. DB2 will interpret the input time based upon its punctuation, then provide time output in the format you specify for this parameter.

ISO is the default. If you want your own time format to be the default, enter LOCAL for this parameter. If you specify LOCAL, you must provide a time exit routine to perform time formatting.

You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. Updating this parameter can affect how your applications run; investigate the impact of an update carefully.

10. LOCAL DATE LENGTH (DEFDATEL)

Accept the default value **0** if you want to use one of the four IBM-supplied formats above (ISO, JIS, USA or EUR). This indicates that no user-defined date format exists in your system. If you want to define your own date format, enter a value between 10 and 254. If you want your own date format to be the default, enter LOCAL for parameter 8 on this panel.

You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. Updating this parameter can affect how your applications run; investigate the impact of an update carefully. Note that increasing LOCAL DATE LENGTH (DEFDATEL) requires you to rebind all affected application plans.

For more information on writing exit routines, see *IBM DATABASE 2 Data Base Planning and Administration Guide*. For more information on installing a user exit, see *IBM DATABASE 2 Install Guide*.

11. LOCAL TIME LENGTH (DEFTIMEL)

Accept the default value **0** if you want to use one of the four IBM-supplied formats above (ISO, JIS, USA or EUR). This indicates that no user-defined time format exists in your system. If you want to define your own time format, enter a value between 8 and 254. If you want your own time format to be the default, enter LOCAL for parameter 9 on this panel.

You can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204. Updating this parameter can affect how your applications run; investigate the impact of an update carefully. Note that increasing LOCAL TIME LENGTH (DEFTIMEL) requires you to rebind all affected application plans.

For more information on writing exit routines, see *IBM DATABASE 2 Data Base Planning and Administration Guide*. For more information on installing a user exit, see *IBM DATABASE 2 Install Guide*.

IRLM Panel 1: DSNTIPI

The information you specify on this panel pertains to the main parameters used to install the IMS/VS Resource Lock Manager (IRLM). The defaults for the parameters on this panel are the same for MVS/370 and MVS/XA.

Although IMS/VS and DB2 do not force the use of a single IRLM, we recommend using one set of IRLM code to simplify service activity. (For more information, see "DB2 and the IRLM" on page 20.) If you use the IRLM for IMS/VS data sharing, the following considerations apply:

- In IMS/VS Release 2.1 and later, if the IRLM is used for data sharing, it will also be used as the IMS/VS local locking mechanism. (If the IRLM is not used, IMS/VS will use its own locking mechanism.)
- Prior to loading DB2 libraries, you must decide if you want separate SMP control data sets for DB2 and IMS/VS.

On this panel you must select two names that relate to the IRLM. The first is the name of the IRLM *subsystem*; you specify this name on the SUBSYSTEM NAME parameter (IRLMSSID). This is the name by which MVS knows the IRLM. The second is the IRLM *start procedure*; you specify this name on the PROC NAME parameter (IRLMPROC). This is the name of the procedure MVS invokes to start the IRLM. You CANNOT specify the same name for both of these parameters.

Though we recommend the defaults (IRLM and IRLMPROC), you can select any name you want for these parameters. If, however, you have installed the IRLM for IMS/VS block-level sharing, the name you specify for SUBSYSTEM NAME must be different from the IRLM name for IMS/VS.

```
DSNTIPI               INSTALL DB2 - IRLM PANEL 1
===>

Enter data below:

 1  INSTALL IRLM       ===> YES       IRLM is required for DB2. Should the
                                      IRLM distributed with DB2 be installed?
 2  SUBSYSTEM NAME     ===> IRLM      IRLM MVS subsystem name
 3  WAIT TIME (SECONDS)===> 60        Wait for unavailable resource
 4  TIME INTERVAL      ===> 500       Milliseconds between lock processing
 5  AUTO START         ===> YES       Start IRLM if not up. YES or NO
 6  PROC NAME          ===> IRLMPROC  Name of start procedure for IRLM
 7  START TIMEOUT      ===> 300       Time to wait for IRLM autostart









PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 52. IRLM Panel 1: DSNTIPI

1. INSTALL IRLM (IRLMINST)

Specify whether to install the IRLM distributed with DB2: YES or NO. The default is YES.

DB2 Release 3 requires Release 4 of the IRLM (this is the version of the IRLM that is distributed with IMS/VS Version 2 Release 2). Some sites may already have this IRLM installed. They may specify NO for this parameter, and for SUBSYSTEM NAME (IRLMSSID) specify the MVS subsystem name for their previously installed IRLM.

2. SUBSYSTEM NAME (IRLMSSID)

Specify the name of the IRLM subsystem. The default is IRLM.

This is the name by which the IRLM is known to MVS. It is used for communication between DB2 and the IRLM. This name will be included in the MVS subsystem table IEFSSN*xx*.

If the IRLM for IMS/VS has been installed for block-level sharing, the name you give to the IRLM for DB2 must be different from the name of the installed subsystem. Otherwise, we recommend that you accept the default value IRLM.

This parameter also sets the IRLMSID parameter of macro DSN6SPRM.

3. WAIT TIME—SECONDS (IRLMWAIT)

Specify the maximum IRLM timeout in seconds. This is the time that the IRLM is to wait for an unavailable resource. This time is used to resolve undetected deadlocks. DB2 will wait only for approximately this length of time for any lock. For certain IMS/VS applications (BMPs), it will wait somewhat longer in case another timed-out application will free locks needed by this application.

This value is referred to as the timeout exit time for the IRLM. When this time elapses, the IRLM checks if users are waiting for resources. In this way, deadlocks are detected.

This minimum value for this parameter is 1; the maximum is 3600. The default is 60.

This parameter also sets the IRLMRWT parameter of macro DSN6SPRM.

4. TIME INTERVAL (IRLMCYCL)

Specify the time, in milliseconds, that the IRLM is to delay between each lock processing.

This is the time that the IRLM delays before processing inter-IRLM requests.

Timeout detection should be monitored carefully. When you set up the TIME INTERVAL parameter, be aware of false deadlocks. This occurs when multiple users request a lock, and a timeout occurs without deadlock because the time interval specified was too short.

The minimum value for this parameter is 20; the maximum is 9999. The default is 500. You can probably accept the default for this parameter. It is used by the IRLM, not DB2.

5. AUTO START (IRLMAUTO)

Specify whether the IRLM is to be started and stopped automatically by DB2: YES or NO. The default is YES.

If you specify YES for this parameter, DB2 will try to start the IRLM if it is not already started when DB2 is started. In addition, when DB2 is stopped, it will automatically stop the IRLM. If you specify NO, DB2 will terminate if the IRLM is not started when DB2 comes up.

If you are using the IRLM only for a single DB2 subsystem, we recommend you accept the default value YES for this parameter. If you are using the IRLM for DB2 and IMS/VS or for multiple DB2 subsystems, you must specify NO.

This parameter also sets the IRLMAUT parameter of macro DSN6SPRM.

6. PROC NAME (IRLMPROC)

This is the name of the IRLM procedure that DB2 will start if you specify YES for the AUTO START parameter described above. The default procedure name is IRLMPROC.

This startup procedure is created during install or migration by job DSNTIJMV and is placed in SYS1.PROCLIB. You can review the IRLM startup procedure supplied with DB2 by examining DSNTIJMV. If the IRLM is not already started when DB2 is started, MVS invokes this procedure.

This parameter also sets the IRLMPRC parameter of macro DSN6SPRM.

7. START TIMEOUT (IRLMSTTO)

Specify the IRLM wait time in seconds. This is the time that DB2 will wait for the IRLM to start during autostart.

When DB2 starts, it will wait for the IRLM for the specified time. If the time expires, DB2 abnormally terminates. The minimum value for this parameter is 1; the maximum is 3600. The default is 300.

This parameter also sets the IRLMSWT parameter of macro DSN6SPRM.

IRLM Panel 2: DSNTIPJ

This panel shows lock parameters used in addition to those on Panel DSNTIPI. Several parameters on Panel DSNTIPJ control the characteristics of IMS/VS time sharing fields. The default values will be valid for most sites under ordinary conditions.

Some of the defaults for the parameters on this panel are different for MVS/370 and MVS/XA.

```
DSNTIPJ                INSTALL DB2 - IRLM PANEL 2
===>

Enter data below:

 1  CROSS MEMORY          ===> YES       Local storage and cross memory usage
 2  MAXIMUM CSA OR ECSA   ===> 5242880   Control block storage (1M-99M)
 3  LOCKS PER TABLE SPACE ===> 1000      Maximum before lock escalation
 4  LOCKS PER USER        ===> 10000     Maximum before resource unavailable
 5  PARTNER               ===>           VTAM name of other IRLM.  Required
                                         for intersystem communication
 6  IDENTIFIER            ===> 1         Number for this IRLM (1-8)
 7  DEADLOCK TIME         ===> 15        Detection interval in seconds
 8  DEADLOCK CYCLE        ===> 4         Number of LOCAL cycles before GLOBAL
 9  INTERNAL TRACE        ===> YES       Automatic start of tracing




PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 53. IRLM Panel 2: DSNTIPJ

1. CROSS MEMORY (IRLMPC)

Specify whether the IRLM is to use the cross address space program call: YES or NO. The default is YES.

If you accept the default value YES, the IRLM control block structure resides in the IRLM private address space, and the program call instruction is used to obtain addressability to the structure. If you are using MVS/370, you may need to accept this default to relieve DB2 storage constraints.

If you specify NO, the IRLM control block structure resides in the common storage area (CSA). For MVS/XA, the extended common storage area (ECSA) will be used if space is available. Using CSA or ECSA requires less processor time, but it can also reduce the size of private address spaces.

2. MAXIMUM CSA OR ECSA (IRLMMCSA)

Specify the maximum amount of control block storage (CSA) that the IRLM is to use for its control block structure. Consider whether you are using CSA in MVS/370, or ECSA in MVS/XA.

You can enter this value in bytes (such as 307200), in kilobytes (such as 300K); or in megabytes (such as 6M); the DSNTINST CLIST will convert kilobyte and megabyte values into byte values.

For MVS/370, the minimum value for this parameter is 100K; the maximum is 9900K. The default is 307200 (300K).

For MVS/XA, the minimum value for this parameter is 1M; the maximum is 99M. The default is 5242880 (5M).

3. LOCKS PER TABLE SPACE (IRLMLKTS)

Specify the maximum number of page locks that may be held concurrently by a single application against a single table space for which LOCKSIZE ANY has been specified. This maximum includes locks against any data pages and index pages (or subpages) that are acquired on behalf of the application when it accesses the table space.

To understand how this parameter functions, you should understand how DB2 selects locks when LOCKSIZE ANY is specified on a CREATE TABLESPACE statement.

DB2 makes the decision at bind time as to what locking level will be used at run time. (For dynamic SQL, bind is done at run time as the first step of execution.)

When a BIND is issued for a table space for which you have specified LOCKSIZE ANY, DB2 usually selects page locking. However, DB2 selects table space locking if page locking is inefficient to fulfill the requested isolation level for the data base operation to be performed.

As soon as DB2 selects page locking for a table space in response to a specification of LOCKSIZE ANY, it establishes and maintains a count of the number of page locks concurrently held by the application against the table space. When and if that count reaches the value you specified for LOCKS PER TABLE SPACE, DB2 escalates the page lock to a table space lock. It stops acquiring page locks, and it releases all page locks that are currently held by the application. Any remaining operations for the application will be performed under the table space lock.

The escalation only pertains to the application (and table space) that reached the specified limit.

The minimum value you can specify for LOCKS PER TABLE SPACE is 0. The maximum for MVS/370 is 25000; the maximum for MVS/XA is 50000.

If you specify 0 for this parameter, you disable lock escalation. In this case, when a BIND is issued against a table space for which LOCKSIZE ANY has been specified, DB2 will almost always select a table space lock.

The default value for this parameter is 1000 for MVS/XA and 200 for MVS/370. If, however, you specified NO for CROSS MEMORY (IRLMPC) on this panel, we recommend that you change this value to consider the available lock space. For MVS/370, you may need to reduce this value to 50. For MVS/XA, you may not need to reduce this value if adequate ECSA space is available.

This parameter also sets the NUMLKTS parameter of macro DSN6SPRM.

4. LOCKS PER USER (IRLMLKUS)

Specify the maximum number of page locks that may be held concurrently by a single application against all table spaces in the system. This maximum includes locks on data pages and index pages (or subpages) that are acquired when the application accesses table spaces. This limit applies to all table spaces for which page locking is in effect. That is, LOCKSIZE PAGE or LOCKSIZE ANY was specified.

As soon as execution begins for an application, DB2 establishes and maintains a count of the total number of page locks concurrently held by the application across all table spaces. When and if that count reaches the value you specify for LOCKS PER USER, DB2 issues a "resource unavailable" condition ("00C90096") and an SQL return code -904. DB2 does not process the SQL request that encountered the limit.

If the application is in a must-complete or must-abort state, requests for page locks are granted, even though they exceed the specified limit.

The minimum value you can specify for LOCKS PER USER is 0. The maximum for MVS/370 is 25000; the maximum for MVS/XA is 100000. DB2 assumes that 200 bytes of storage are required for each lock.

If you specify 0 for this parameter, you disable the user lock limit function.

The default value for this parameter is 10000 for MVS/XA and 2000 for MVS/370. If you specified NO for CROSS MEMORY (IRLMPC) on this panel, we recommend that you change the value to consider the available lock space. For MVS/370, you may need to reduce this value to 500. For MVS/XA, you may not need to reduce this value if adequate ECSA space is available.

This parameter also sets the NUMLKUS parameter of macro DSN6SPRM.

5. PARTNER (IRLMPART)

Specify the VTAM application name of the other IRLM which is sharing data. The default is null. We recommend that the VTAM application name be the same as the other IRLM's procedure name. Use parameters 5 to 9 to specify values for IMS/VS data sharing. If you are not using IMS/VS data sharing, accept the defaults for these parameters.

6. IDENTIFIER (IRLMIDEN)

Specify a decimal number, from 1 to 8, that will be used internally to distinguish this IRLM from the other IRLM when data sharing is in effect. The default for both MVS/370 and MVS/XA is 1.

7. DEADLOCK TIME (IRLMDEDT)

Specify the time, in seconds, of the local deadlock detection cycle.

A deadlock is a situation where two (or more) requesters are waiting for resources held by the other. Deadlock detection is the procedure by which a deadlock and its participants are identified.

Specify a value from 1 to 99. The default is 15.

8. DEADLOCK CYCLE (IRLMDEDC)

Specify the number of local deadlock cycles that must expire before the IRLM performs global deadlock detection processing. Specify a value from 1 to 99. The default is 4.

9. INTERNAL TRACE (IRLMTRAC)

Specify whether the IRLM internal trace is to be automatically started when the IRLM is started: YES or NO. The default is YES.

Protection Panel: DSNTIPP

The information you specify on this panel is used for security matters. For the password fields:

- If your site has assigned a password for any of the data sets identified on this panel, enter the password for that entry.
- If you are defining a data set for the first time, you may enter a password for it in this panel.
- If your site has not assigned a password for any of the data sets, make the field blank.
- To use RACF protection and not passwords, make the password fields blank.

The defaults for the parameters on this panel are the same for both MVS/370 and MVS/XA. If you are migrating, DB2 Release 3 adopts your Release 2 DB2 catalog, directory, temporary data bases, BSDS, and active and archive logs. Consequently, during migration, you cannot change any of the parameters that affect the characteristics of these objects. The parameters on Panel DSNTIPP which you cannot change during migration are listed below; see "Migrating DB2" on page 30 for a complete list.

- ICF CATALOG (PROTVCAT) is the password for the ICF catalog
- BSDS PASSWORD (PROTBSDS) is the password for the BSDS
- LOG PASSWORD (PROTLOGS) is the password for active log data sets
- ARCHIVE LOG PW (PROTARCH) is the password for the archive log data sets
- ARCHIVE LOG RACF (PROTARAC) specifies whether to RACF-protect archive log data sets

You can alter these parameters by using an update process after migration. See Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

```
DSNTIPP         INSTALL DB2 - PROTECTION
===>

Enter data below:

 1  ICF CATALOG        ===> DSNDEFPW   ICF catalog control password
 2  BSDS PASSWORD      ===> DSNOPER1   Bootstrap data set(s) password
 3  LOG PASSWORD       ===> DBADMIN    Active log data sets password
 4  ARCHIVE LOG PW     ===> DBADMIN    Archive log data sets password
 5  ARCHIVE LOG RACF   ===> NO         RACF protect archive log data sets
 6  DIRECTORY/CATALOG  ===> DBADMIN    DB2 directory and catalog password
 7  USE PROTECTION     ===> YES        DB2 authorization enabled.  YES or NO
 8  SYSTEM ADMIN 1     ===> SYSADM     Authid of system administrator
 9  SYSTEM ADMIN 2     ===> SYSADM     Authid of system administrator
10  SYSTEM OPERATOR 1  ===> SYSOPR     Authid of system operator
11  SYSTEM OPERATOR 2  ===> SYSOPR     Authid of system operator
12  UNKNOWN AUTHID     ===> IBMUSER    Authid of default (unknown) user



PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 54. Protection Panel: DSNTIPP

1. ICF CATALOG (PROTVCAT)

Specify the password required to add new entries to the DB2 ICF catalog. If you are installing Release 3, the default is DSNDEFPW.

If you specify YES for the DEFINE CATALOG (VCATSTAT) parameter on the Data Parameters Panel (DSNTIPA2), this password is also used in the catalog definition.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

2. BSDS PASSWORD (PROTBSDS)

Specify the password used to define the bootstrap data sets. The default is DSNOPER1. This password must be entered by the operator when starting DB2. The operator is prompted when DB2 issues OPEN.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

3. LOG PASSWORD (PROTLOGS)

Specify the password used to define and access the active log data sets. The default is DBADMIN. This password is used during definition of the active log VSAM clusters during DB2 initialization.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

4. ARCHIVE LOG PW (PROTARCH)

Specify the password used to define and access the archive log data sets. The default is DBADMIN. This password is used when active log data sets are off-loaded to the archive log. The CHANGE LOG INVENTORY utility must be used to remove this password protection. See *IBM DATABASE 2 Command and Utility Reference* for information on the CHANGE LOG INVENTORY utility.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

5. ARCHIVE LOG RACF (PROTARAC)

Specify whether archive log data sets are to be protected with the Resource Access Control Facility (RACF) when they are created: NO or YES. The default is NO.

If you specify YES for this parameter, RACF protection must be active for DB2, and RACF class TAPEVOL must be active if your archive log goes to tape. If class TAPEVOL is not active and you choose YES for this parameter, your archive will fail.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

If you use RACF, you must define the DB2 subsystem name to RACF. Information about how to do this appears in Appendix B, "RACF Examples for DB2" on page 243.

This parameter also sets the PROTECT parameter of macro DSN6ARVP.

6. DIRECTORY/CATALOG (PROTDIRC)

Specify the password for the DB2 catalog, directory, and temporary data sets. The default is DBADMIN. This password is used during definition of VSAM clusters. It is kept in the BSDS and is used when the OPEN macro is issued.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

7. USE PROTECTION (PROTAUTH)

Specify whether the SQL authorization control statements GRANT and REVOKE are to be effective: YES or NO. The default is YES. If you accept the default YES, the DB2 authorization mechanism will be activated when the subsystem is started. If you specify NO, you disable authorization checking in DB2.

This parameter also sets the AUTH parameter of macro DSN6SPRM.

8. SYSTEM ADMIN 1 (PROTADMN)

Specify the first of two authorization IDs with Install SYSADM authority. The two users with Install SYSADM authority are permitted access to DB2 in all cases. The value you specify for this parameter must be alphanumeric and contain from 1 to 8 characters; the first character must be alphabetic. The default is SYSADM.

The user owning this ID can use the SQL GRANT statement to authorize new users. Consider using a functional userid for each of the two Install SYSADMs, such that the userid is tied to the role of Install SYSADM, rather than to the person filling that role. Management approval might be required to change the password before the functional userid is passed on to the new person. In this way you avoid having to revoke Install SYSADM authority; this can be a problem in that the revoke cascades, and all authorities granted by this Install SYSADM will also be revoked. You should assign this ID to an existing TSO, RACF, IMS/VS, or CICS userid. You may want to specify a TSO userid, because many ease-of-use facilities for DB2 are provided through ISPF.

This parameter also sets the SYSADM parameter of macro DSN6SPRM.

9. SYSTEM ADMIN 2 (PROTADM2)

Specify the second of two authorization IDs with Install SYSADM authority. The two users with Install SYSADM authority are permitted access to DB2 in all cases. The default is SYSADM. See SYSTEM ADMIN 1 above.

This parameter also sets the SYSADM2 parameter of macro DSN6SPRM.

10. SYSTEM OPERATOR 1 (PROTOPR1)

Specify the first of two authorization IDs with Install SYSOPR authority. The default is SYSOPR.

The two users with Install SYSOPR authority are permitted access to DB2 even if the DB2 catalog is unavailable. To avoid serious system problems, you must accept the default value SYSOPR for either SYSTEM OPERATOR 1 (PROTOPR1) or SYSTEM OPERATOR 2 (PROTOPR2). You should assign this ID to an existing TSO, RACF, IMS/VS, or CICS user. You may want to specify a TSO user ID, because many ease-of-use facilities for DB2 are provided through ISPF.

This parameter also sets the SYSOPR1 parameter of macro DSN6SPRM.

11. SYSTEM OPERATOR 2 (PROTOPR2)

Specify the second of two system operators permitted access to DB2 even if the DB2 catalog is unavailable. The default is SYSOPR. See SYSTEM OPERATOR 1 above.

This parameter also sets the SYSOPR2 parameter of macro DSN6SPRM.

12. UNKNOWN AUTHID (PROTUNKN)

Specify the authorization ID used if RACF is not available for batch access and if the USER= parameter is not specified in the job statement. The default is IBMUSER. Null is not a valid value.

This parameter also sets the DEFLTID parameter of macro DSN6SPRM.

MVS PARMLIB Updates Panel: DSNTIPM

The information you specify on this panel is used to produce the DSNTIJMV job that defines DB2 to MVS.

With the information you provide, DB2 prepares updates of the following PARMLIB members:

- IEFSSN*xx*, to define DB2 and IRLM as formal MVS subsystems
- IEAAPF*xx*, to authorize the DSN130.DSNLOAD and DSN130.DSNLINK libraries
- LNKLST*xx*, to include the DSN130.DSNLINK library

To perform these updates, the CLIST edits job DSNTIJMV. This job is used whether you are installing or migrating DB2.

Different sites have different requirements for identifying DB2 to MVS and as a result, the updates that DSNTIJMV makes to MVS PARMLIB members may be incomplete. To ensure that the updates are complete, we recommend that you make the updates directly, with an editor, to the MVS PARMLIB members when you install or migrate DB2. This will be substantially easier for you than editing DSNTIJMV.

The defaults for the parameters on this panel are the same for both MVS/370 and MVS/XA.

```
DSNTIPM         INSTALL DB2 - MVS PARMLIB UPDATES
===>

Check data and reenter to change:

 1  SUBSYSTEM NAME      ===> DSN        Name for connecting to DB2
 2  SUBSYSTEM PREFIX    ===> -          One character to denote subsystem
 3  SUBSYSTEM MEMBER    ===> 00         xx in IEFSSNxx
 4  SUBSYSTEM SEQUENCE  ===> 88888888   Sequence number for insertion
 5  AUTH MEMBER         ===> 00         xx in IEAAPFxx APF member name
 6  AUTH SEQUENCE       ===> 88888888   Sequence number for insertion
 7  LINK LIST ENTRY     ===> 00         xx in LNKLSTxx for DSNLOAD
 8  LINK LIST SEQUENCE  ===> 88888888   Sequence number for insertion









PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 55. MVS PARMLIB Updates Panel: DSNTIPM

1. SUBSYSTEM NAME (MVSSNAME)

Specify the MVS subsystem name for DB2. It is used in the IEFSSNxx member of SYS1.PARMLIB.

The value you specify for this parameter must be alphanumeric and contain from 1 to 4 characters; the first character must be alphabetic. The default is DSN.

2. SUBSYSTEM PREFIX (MVSSPREF)

Specify the DB2 subsystem recognition character (SRC). The default is the hyphen (-).

MVS uses the SRC to identify DB2 commands. When the SRC appears at the beginning of a command entered at an MVS operator's console, the command is passed to DB2 for processing.

The default SRC is the hyphen (-), but you can use any special character. If you want to use the ampersand (&), however, you must edit job DSNTIJMV; you cannot enter the ampersand from the panel. Do not use an SRC already in use by any other subsystem (including JES). If you use JES2, do not use the JES2 backspace character.

The SRC is used in the DB2 entry of the IEFSSNxx member of SYS1.PARMLIB.

3. SUBSYSTEM MEMBER (MVSSMEMB)

Specify the last two characters (*xx*) of the subsystem name member of SYS1.PARMLIB, IEFSSN*xx*. The subsystem name member indicates the available MVS subsystems, including DB2 and the IRLM.

The value you specify for this parameter must be alphanumeric and contain 2 characters. The first character must be alphabetic. The default is 00.

4. SUBSYSTEM SEQUENCE (MVSSSEQN)

Specify the sequence number of the last insertion in member IEFSSN*xx* of SYS1.PARMLIB. This number should not already be present in the member.

Specify a value from 1 to 99999999. The default is 88888888.

5. AUTH MEMBER (MVSAMEMB)

Specify the *xx* suffix of member IEAAPF*xx* in SYS1.PARMLIB. The default is 00. This member is used for authorized program facility (APF) authorization of the DSN130.DSNLOAD library. This data set must be APF-authorized. The member name must currently exist for the MVS update job DSNTIJMV to function correctly.

6. AUTH SEQUENCE (MVSASEQN)

Specify the last sequence number used by IEAAPF*xx*: Specify a value from 1 to 99999999. The default is 88888888.

7. LINK LIST ENTRY (MVSLMEMB)

Specify the *xx* suffix of LNKLST*xx* needed to include the DSN130.DSNLINK library. The value you specify for this parameter must be alphanumeric and contain 2 characters. The default is 00.

8. LINK LIST SEQUENCE (MVSLSEQN)

Specify the last sequence number used by LNKLST*xx*. Specify a value from 1 to 99999999. The default is 88888888. This sequence number should not already be present in the member.

Log Data Sets Panel: DSNTIPL

DB2 uses the information you specify on this panel to define active and archive log data set characteristics.

The defaults for the parameters on this panel are different for MVS/370 and MVS/XA.

Dual logging improves reliability of recovery. We strongly recommend that you specify dual logging for both active and archive logs. If you do this and an error occurs during archiving, DB2 will restart the archive process using the second copy of the active log.

If you specify dual active logging, you should place the two active logs on different DASD volumes and, ideally, on different channels and control units. To do this, specify different volume serial numbers on the VOLUME SERIAL 5 and VOLUME SERIAL 6 parameters (VOLSDAT5 and VOLSDAT6) on the Data Parameters Panel (DSNTIPA2).

The capacity you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might, during rollback and restart, need to access data in the archive log. One parameter on Panel DSNTIPL affects the capacity of the active log:

- NUMBER OF LOGS (LOGSNUM) controls the number of active log data sets you create

Increasing the value you specify for this parameter increases the capacity of the active log. See "Determining the Size of Active Logs" on page 199 for a complete list of factors affecting the active and archive logs. **If you are migrating,** DB2 Release 3 adopts your Release 2 BSDS and active and archive logs. Consequently, during migration, you cannot change any of the parameters that affect the characteristics of these objects. The parameters on Panel DSNTIPL which you cannot change during migration are listed below; see "Migrating DB2" on page 30 for a complete list.

- NUMBER OF COPIES (LOGSTWO) specifies dual or single active logs
- DATA SET PREFIX 1 (LOGSPRE1) is the prefix for the first copy of the active log
- DATA SET PREFIX 2 (LOGSPRE2) is the prefix for the second copy of the active log
- NUMBER OF LOGS (LOGSNUM) is the number of active log data sets
- BOOTSTRAP NAME 1 (BSDSNAM1) is the name of the first BSDS copy
- BOOTSTRAP NAME 2 (BSDSNAM2) is the name of the second BSDS copy

You can, however, alter these parameters by using an update process after migration. See Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

```
DSNTIPL         INSTALL DB2 - LOG DATA SETS
===>

Enter data below:

 1  NUMBER OF COPIES  ===> 2        2 or 1.  Number of active log copies
 2  DATA SET PREFIX 1 ===> DSNC130.LOGCOPY1
 3  DATA SET PREFIX 2 ===> DSNC130.LOGCOPY2
 4  NUMBER OF LOGS    ===> 3        Number data sets per active log copy (2-53)
 5  INPUT BUFFER      ===> 28K      Size in bytes (28K-60K)
 6  OUTPUT BUFFER     ===> 400K     Size in bytes (32K-4000K)
 7  WRITE THRESHOLD   ===> 20       Buffers filled before write (1-256)
 8  BOOTSTRAP NAME 1 ===> DSNC130.BSDS01
 9  BOOTSTRAP NAME 2 ===> DSNC130.BSDS02




PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 56. Log Data Sets Panel: DSNTIPL

1. NUMBER OF COPIES (LOGSTWO)

Specify the number of copies of the active log that will be maintained: 2 (dual logging) or 1 (single logging). The default is 2. Dual logging increases reliability of recovery.

You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

This parameter also sets the TWOACTV parameter of macro DSN6LOGP.

2. DATA SET PREFIX 1 (LOGSPRE1)

See the description of DATA SET PREFIX 2.

3. DATA SET PREFIX 2 (LOGSPRE2)

Specify the prefix for the first and second copies of the active log data sets. If you are installing Release 3, the default prefix is DSNC130.LOGCOPY*x*. The resulting name would be:

DSNC130.LOGCOPY*x*.DS*nn*

where:

DSNC130 is the value you specified for the ICF catalog alias on the Data Parameters Panel (DSNTIPA2). You cannot change this portion of the data set prefix on panel DSNTIPL.

LOGCOPY represents the portion of the data set prefix that you can change on Panel DSNTIPL.

x is **1** for the first copy of the logs and **2** for the second copy.

nn is the data set number.

If you are using single logging, accept the default value. Do not specify nulls.

You cannot change these parameters during a migration to DB2 Release 3 because the data sets from the previous release are still used. You must use the same value you specified for Release 2. After completing the migration process, however, you can update them using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

4. NUMBER OF LOGS (LOGSNUM)

Specify the number of active log data sets for each copy of the log. If you accept the default, which is 3, and you specified **2** for the NUMBER OF COPIES (LOGSTWO) parameter on this panel, 6 active log data sets are created. The minimum value for this parameter is 2; the maximum is 53.

You can change this parameter if you are installing Release 3. You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing migration, however, you can update this parameter. See Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

5. INPUT BUFFER (LOGINPUT)

Specify the size, in bytes, of the input buffer used for reading active and archive log data sets. This buffer is used during recovery operations. Each time a read operation is started, a buffer of the specified size is allocated. Specify a value from 28K to 60K. The default is 28K.

This parameter also sets the INBUFF parameter of macro DSN6LOGP.

6. OUTPUT BUFFER (LOGOUTPT)

Specify the size, in bytes, of the output buffer used for writing active and archive log data sets. This is the amount of storage used for buffering log operations.

Specify a value from 32K to 4000K. The default is 200K for MVS/370 and 400K for MVS/XA.

This parameter also sets the OUTBUFF parameter of macro DSN6LOGP.

7. WRITE THRESHOLD (LOGTHRSH)

Specify the number of buffers to be filled before starting a write. Specify a value from 1 to 256. The default is 10 for MVS/370 and 20 for MVS/XA.

This parameter also sets the WRTHRSH parameter of DSN6LOGP.

8. BOOTSTRAP NAME 1 (BSDSNAM1)

Specify the name of the first copy of the bootstrap data set. If you are installing Release 3, the default is DSNC130.BSDS01.

You can change this value if you are installing Release 3. You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

9. BOOTSTRAP NAME 2 (BSDSNAM2)

Specify the name of the second copy of the bootstrap data set. If you are installing Release 3, the default is DSNC130.BSDS02.

You can change this value if you are installing Release 3. You cannot change this parameter during a migration to DB2 Release 3. You must use the same value you specified for Release 2. After completing the migration process, however, you can update this parameter using the process described in Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

Archive Log Data Sets Panel: DSNTIPA

The information you specify on this panel is used to define the characteristics of archive log data sets. The defaults for the parameters on this panel are the same for MVS/370 and MVS/XA.

Dual logging improves reliability of recovery. We recommend that you specify dual logging for both active and archive logs. If you do this and an error occurs during archiving, DB2 will restart the archive process using the second copy of the active log.

```
DSNTIPA         INSTALL DB2 - ARCHIVE LOG DATA SETS
===>

Enter data below:

 1  NUMBER OF COPIES ===> 2         2 or 1.  Number of archive log copies
 2  ARCHIVE PREFIX   ===> DSNC130.ARCHLOG1
 3   COPY 2 PREFIX   ===> DSNC130.ARCHLOG2
 4  CATALOG DATA     ===> NO        YES or NO to catalog archive data sets
 5  DEVICE TYPE      ===> TAPE      Unit name for archive logs
 6  BLOCK SIZE       ===> 8192      Rounded up to 4096 multiple
 7  MAX CONCURRENT   ===> 3         Number of input volumes allocated
 8  RECORDING MAX    ===> 500       Number of data sets recorded in BSDS
 9  MSVGP 1          ===>           Mass storage volume group (MSVGP)
10  MSVGP 2          ===>           MSVGP for second archive copy
11  WRITE TO OPER    ===> NO        Issue WTOR before mount for archive
12  WTOR ROUTE CODE  ===> 1,3,4
                                    Routing codes for archive WTORs
13  RETENTION PERIOD ===> 9999      Days to retain archive log data sets


PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 57. Archive Log Data Sets Panel: DSNTIPA

1. NUMBER OF COPIES (ARCHTWO)

Specify the number of copies of the archive log to be produced during off-loading: 2 (dual logging) or 1 (single logging). The default is 2. Dual logging increases reliability of recovery.

This parameter also sets the TWOARCH parameter of macro DSN6LOGP.

2. ARCHIVE PREFIX (ARCHPRE1)

See the description of COPY 2 PREFIX (ARCHPRE2).

3. COPY 2 PREFIX (ARCHPRE2)

Specify the prefix of the first and second copy of archive log data sets. If you are installing Release 3, the default prefix is DSNC130.ARCHLOG*x*. The resulting name would be:

DSNC130.ARCHLOG*x*.A*nnnnn*

where:

DSNC130 is the value you specified for the ICF catalog alias on the Data Parameters Panel (DSNTIPA2). You cannot change this portion of the data set prefix on panel DSNTIPA.

ARCHLOG represents the portion of the data set prefix that you can change on Panel DSNTIPA.

x is **1** for the first copy of the logs and **2** for the second copy.

A*nnnnn* is generated internally.

ARCHIVE PREFIX (ARCHPRE1) also sets the ARCPFX1 parameter of macro DSN6ARVP. COPY 2 PREFIX (ARCHPRE2) also sets the ARCPFX2 parameter of macro DSN6ARVP.

If you are using single logging, accept the default value. Do not specify nulls.

4. CATALOG DATA (ARCHCTLG)

Specify whether archive log data sets are to be cataloged in the primary ICF catalog: YES or NO. The default is NO.

This parameter also sets the CATALOG parameter of macro DSN6ARVP.

5. DEVICE TYPE (ARCHDEVT)

Specify the device type or unit name on which archive log data sets will be stored.

The value you specify can be any alphanumeric string; the first character must be alphabetic. The default is TAPE. If you choose to archive to DASD, you may wish to specify a generic device type with a limited volume range.

This parameter also sets the UNIT parameter of macro DSN6ARVP.

6. BLOCK SIZE (ARCHSIZE)

Specify the block size of the archive log data set. Ensure that the block size is compatible with the device type. For instance, if the device type is changed from tape to DASD, ensure that the DASD specified can support the block size.

Specify a value from 8192 to 28672. The default is 8192. The value you specify is rounded up to the next multiple of 4096 bytes.

This parameter also sets the BLKSIZE parameter of macro DSN6ARVP.

7. MAX CONCURRENT (ARCHINPT)

Specify the maximum number of archive log volumes that can be allocated for input mode at the same time.

Archive log data sets are used in input mode when a read of the log is required in a system restart or dynamic backout and when required for data base recovery.

Specify a value from 1 to 99. The default is 3.

This parameter also sets the MAXALLC parameter of macro DSN6LOGP.

8. RECORDING MAX (ARCHMAXV)

Specify the maximum number of archive log volumes to be recorded in the BSDS. When this number is exceeded, recording resumes at the beginning of the BSDS.

Specify a value from 10 to 1000. The default is 500.

This parameter also sets the MAXARCH parameter of macro DSN6LOGP.

9. MSVGP 1 (ARCHMSGP)

Specify the mass storage volume group (MSVGP) to be used for the first copy of archive log data sets. On the MSS, the parameter entered in this field is used to group volumes for specific purposes. The default is null.

This parameter also sets the MSVGP parameter of macro DSN6ARVP.

10. MSVGP 2 (ARCHMSG2)

Specify the mass storage volume group (MSVGP) to be used for the second copy of archive log data sets. On the MSS, the parameter entered in this field is used to group volumes for specific purposes. The default is null.

This parameter also sets the MSVGP2 parameter of macro DSN6ARVP.

11. WRITE TO OPER (ARCHWTOR)

Specify whether to send a message to the operator and wait for an answer before attempting to mount an archive log data set: NO or YES. The default is NO. Other DB2 users may be forced to wait while the mount is pending. They will not be affected while DB2 is waiting for a response to the message. Use:

NO (default) to omit the message. This value is usually selected when archive devices, such as DASD and MSS, can satisfy a mount request without a long delay.

YES to send the message and wait for an answer. This value is usually selected when archive devices, such as tape drives, have a long delay before satisfying a request.

This parameter also sets the ARCWTOR parameter of macro DSN6ARVP.

12. WTOR ROUTE CODE (ARCHWRTC)

Specify the list of route codes from messages from the archive log data sets to the operator.

You must specify at least one number. You can specify up to 14 numbers. Separate numbers in the list by commas only, not by blanks. The default is:

```
1,3,4
```

For descriptions of the routing codes, see *OS/VS Message Library: VS2 Routing and Descriptor Codes* and *MVS/Extended Architecture Message Library: Routing and Descriptor Codes.*

These routing codes are also discussed in the description of the WTO macro in *OS/VS2 MVS Supervisor Services and Macros* and *MVS/Extended Architecture System Programming Library: Supervisor Services and Macro Instructions.*

This parameter also sets the ARCWRTC parameter of macro DSN6ARVP.

13. RETENTION PERIOD (ARCRETN)

Specify the number of days that DB2 is to retain archive log data sets.

Specify a value from 0 to 9999. The default is 9999.

This retention period is often used in MSS and tape management systems to control the reuse and scratching of data sets and tapes. DB2 uses the value you specify for ARCRETN as the value for the JCL parameter RETPD when archive log data sets are created. The retention period is added to the current date to calculate the expiration date. If a value for the retention period causes the expiration date to exceed 99365, the expiration date is set to 99365.

You must create image copies of all DB2 objects, probably several times, before the archive log data sets are discarded. If you fail to retain an adequate number of archive log data sets for all the image copies, you might have to cold start or reinstall DB2. In both cases, data is lost.

For information about managing the log and determining how long to keep the logs, refer to *IBM DATABASE 2 Operation and Recovery Guide*.

This parameter also sets the ARCRETN parameter of macro DSN6ARVP.

Data Bases and Spaces to Start Automatically Panel: DSNTIPS

Use this panel to specify the data bases, table spaces, and/or index spaces to restart automatically when you start DB2.

The defaults for the parameters on this panel are the same for MVS/370 and MVS/XA.

```
DSNTIPS         INSTALL DB2 - DATA BASES AND SPACES TO START AUTOMATICALLY
===>

Enter data below:

 1  ===> RESTART       RESTART or DEFER the objects named below.
    The objects to restart or defer may be ALL in item 2, a data base
    name, or data base name.space name.

 2  ==> ALL              14  ==>                  26  ==>
 3  ==>                  15  ==>                  27  ==>
 4  ==>                  16  ==>                  28  ==>
 5  ==>                  17  ==>                  29  ==>
 6  ==>                  18  ==>                  30  ==>
 7  ==>                  19  ==>                  31  ==>
 8  ==>                  20  ==>                  32  ==>
 9  ==>                  21  ==>                  33  ==>
10  ==>                  22  ==>                  34  ==>
11  ==>                  23  ==>                  35  ==>
12  ==>                  24  ==>                  36  ==>
13  ==>                  25  ==>                  37  ==>

PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 58. Data Bases and Spaces to Start Automatically Panel: DSNTIPS

1. START OR DEFER (DBSTADFR)

Specify whether DB2 will perform restart processing for the objects listed in items 2 through 37 when DB2 is started: RESTART or DEFER.

If you accept the default RESTART, DB2 performs restart processing for the items listed. If you specify DEFER, DB2 does not perform restart processing for the items listed.

DEFER ALL includes the DB2 system data bases DSNDB01, DSNDB04, DSNDB06 and DSNDB07. For information about the restart process and the implications of deferring objects at restart time, see *IBM DATABASE 2 Operation and Recovery Guide*.

2 - 37. START NAMES (DBSTARTx)

Use these fields to enter the names of the data bases and spaces for which you want to control restart processing. Enter one of the following values for these fields:

- **ALL** on field 2 (leaving fields 3 - 37 blank) to restart or defer all DB2 data bases and spaces. This is the default.

  DEFER ALL includes the DB2 system data bases DSNDB01, DSNDB04, DSNDB06 and DSNDB07. For information about the restart process and the implications of deferring objects at restart time, see *IBM DATABASE 2 Operation and Recovery Guide.*

- Data base name to restart or defer all spaces in that data base.
- Table space or index space name in the format "database-name.space-name" to restart or defer the individual table or index space.

You can specify up to 36 object names on this panel. If you want to control restart processing for more than 36 objects, edit job DSNTIJUZ after you run the CLIST, and add the object names as ending positional parameters to macro DSN6SPRM. You can add up to 2500 object names in DSNTIJUZ.

Section 2: System Administration

This section discusses system administration— managing the resources that affect the entire DB2 subsystem. This involves tasks that you perform after you complete an install or migration. This section consists of the following chapters:

After you install or migrate DB2, system administration tasks replace system planning tasks. The two tasks, however, are closely related. System administration consists of providing security for DB2, attaching DB2 to other subsystems, monitoring DB2, tuning DB2, and updating the options you selected during system planning. You will perform many of the tasks discussed in this section on an ongoing basis, as the need arises.

- **Providing Security**

  Authorization is the key to DB2 subsystem security. Within DB2, specific degrees of authorization can be given to different users. For example, a user designated as a system administrator can grant data base administration authority to another user. Similarly, a user designated as a data base administrator can grant varying degrees of authority to other users. These users may, in turn, grant authority to still other users.

- **DB2 Attachment Facilities**

  Applications that access DB2 resources can run in one of four environments: TSO, batch, IMS/VS, or CICS. To support a DB2 application, you must have the attachment facility provided for the subsystem (TSO, IMS/VS, or CICS) under which the application runs. The TSO and call attachment facilities support batch as well as TSO access.

  Use of the TSO attachment facility is required. It allows you to use the DB2 online functions that are provided with DB2 and to bind application plans. Use of call attachment facility and the IMS/VS and CICS attachment facilities is optional.

- **Monitoring DB2**

  The DB2 trace facility allows you to monitor significant events within DB2. You can use the trace facility to collect statistical, accounting, performance, and serviceability data. DB2 provides commands that allow you to start, stop, and display trace activity. You can use the information the trace facility generates to help tune DB2.

- **Tuning DB2**

  Tuning DB2 is an iterative process. You should tune DB2 after you complete the install or migration process, and thereafter as the need arises.

  DB2 is a complex subsystem, and many factors affect its performance. Overemphasizing one factor can easily create a bottleneck in another area. Therefore, it is important to understand DB2 monitoring tools and tuning guidelines. This will help you configure DB2 to establish a balance of all factors.

- **Updating DB2 Parameters**

  You can modify all but two of over a hundred parameter values you specify when you first install DB2. This capability allows you to adjust DB2 characteristics to your needs without requiring you to reinstall or remigrate. The update process is similar to install and migration. If you have ISPF, you can use ISPF panels to perform many of the updates.

Chapter 5. Providing Security

This chapter describes the security mechanisms DB2 provides to control its use. It gives an overview of DB2 security and shows how DB2 security interacts with the security facilities of MVS, IMS/VS, CICS, TSO, and batch.

Some general things to keep in mind when you read this chapter are:

- What degree of security will you need? To what degree do you need to restrict the use of your DB2 data?
- How do you want to structure user authority to access DB2 data?
- How do you want DB2 security to relate to your system security?

This chapter contains the following major sections:

- The DB2 authorization approach
- Granting and revoking authorization
- Distributing authority: central or distributed
- Controlling access to the DB2 subsystem
- Controlling access to DB2 data
- Using the catalog to help administer authority

The DB2 Authorization Approach

DB2 security requires that all users be authorized to do whatever they need to do. Every DB2 capability that a user can perform must be authorized, either explicitly or implicitly. Each DB2 user should have a unique authorization ID.

An authorization ID is a short identifier that designates a user. See *IBM DATABASE 2 SQL Reference* for more information regarding short identifiers.

To create a table, a user must be authorized to create tables; to use a table space, a user must be authorized to use that table space; and so on. Creators of tables have certain implicit authorizations for their objects; all explicit authorizations for other users are granted through the GRANT statement. The REVOKE statement takes these privileges away. The five levels of administrative authorities, described next, can use the GRANT and REVOKE statements to effectively distribute authority among DB2 users.

Special Administrative Authorities

DB2 provides *administrative* and *individual* authorities. Administrative authorities are arranged in a hierarchy. At the head of this hierarchy is a system administrator. Individual authorities relate to tasks, such as the ability to run STOSPACE utility and the ability to use the CREATE TABLESPACE command.

Those who are granted administrative authorities automatically possess a certain set of individual authorities. Individual authorities, however, can also be granted on a case-by-case basis.

The hierarchy of the administrative authorities is shown in Figure 59 on page 120.

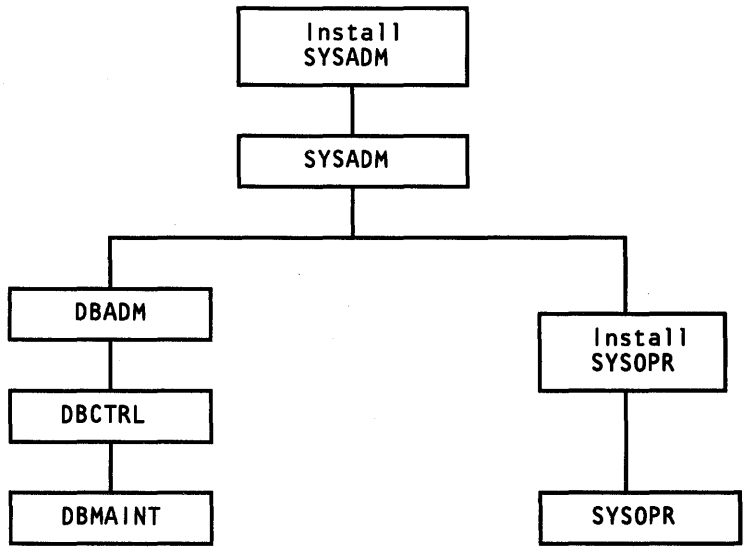


Figure 59. Administrative Authorities

A brief description of each administrative authority follows:

- **Install SYSADM** (System administration): The one or two users designated as system administrator during the install or migration process are at the head of the DB2 authorization structure. These users have *Install SYSADM* authority. The Install SYSADM is permitted access to DB2 in all cases and has control over all DB2 resources.

  There are a few differences between the Install SYSADM and other users with SYSADM authority.

  - The Install SYSADM is not recorded in the DB2 catalog. Thus, the catalog need not be available for authorization checks involving the Install SYSADM.
  - No other user can revoke SYSADM authority from the Install SYSADM.
  - Only the Install SYSADM can repair or recover certain critical subsystem table spaces and indexes.

  The Protection Panel (DSNTIPP) allows you to define two IDs with Install SYSADM authority. The two Install SYSADMs have nearly identical privileges.

- **SYSADM** (System administration): Except for certain critical subsystem table spaces and indexes that are recoverable only by the Install SYSADM, persons having SYSADM authority have control over all DB2 resources. They can grant and revoke any of the other levels of administrative authority, and they can grant and revoke any individual authority. They can even revoke an authority granted by another user. Therefore, use the authorization ID that is granted SYSADM authority only for those occasions when you need special authority—not for normal work.

  Users with Install SYSADM or SYSADM authority are the only users who can retrieve information from the DB2 catalog (using the SELECT statement) without being explicitly granted that capability. They can drop any DB2 object except the DB2 catalog and certain other system data bases. They cannot prohibit other users from accessing their own tables or revoke the authority other users have to grant privileges on those tables.

- **DBADM** (Data base administration): These users control a particular DB2 data base and have all capabilities on objects within the data base. A user can have DBADM authority over more than one DB2 data base, but having DBADM authority over one data base does not automatically give a user authority to create other data bases. To create a data base, a user would have to be granted the CREATEDBA privilege by a system administrator.
- **DBCTRL** (Data base control): Users having DBCTRL have all the capabilities of DBADM except that they cannot automatically access the data in the data base.
- **DBMAINT** (Data base maintenance): Users with DBMAINT authority have read-only privileges of DBCTRL. This level of authorization is meant for users who perform such tasks as running utilities to make image copies and obtain statistics.
- **Install SYSOPR** (System operation): One or two users are designated as the system operator(s) during install or migration. These users can issue system operation commands, as can the SYSOPRs. However, there are a few differences between the Install SYSOPR and the SYSOPR:
  - The Install SYSOPR is not recorded in the DB2 catalog. Thus, the catalog need not be available for authorization checks involving the Install SYSOPR.
  - No other user can revoke the SYSOPR authority from the Install SYSOPR.
  - The Install SYSOPR can perform the IMAGCOPY utility on the critical DB2 system data bases (DSNDB01 and DSNDB06). However, the Install SYSOPR does not have authority to access these data bases.

  The Protection Panel (DSNTIPP) allows you to define two IDs with Install SYSOPR authority.
- **SYSOPR** (System operation): Users with SYSOPR authority can issue system operation commands, but they have no access to DB2 data bases.

The individual authorities making up the special administrative authorities are shown in Figure 60 on page 122.

Each administrative authority has the individual authorities shown in its box, as well as the individual authorities for all levels of authority beneath it. For example, DBADM has ALTER, DELETE, INDEX, INSERT, SELECT, and UPDATE authorities as well as those listed for DBCTRL and DBMAINT.

These authority levels allow system administrators to separate the task of data base administration by data base. Different groups can administer their own data without affecting any other data base.

Application programmers don't need any authority to code and precompile programs that access data. However, specific privileges might need to be granted to programmers to create tables, bind and execute programs, and access DB2 data. Your site might grant these privileges to each programmer, or might grant such privileges to a project authorization ID which could be used by all programmers working on the application.

The user who binds an application plan must have the authority to access the data used by that plan. However, a user who executes a program that uses the plan does *not* need authority to access the data; the user needs only the EXECUTE privilege on the plan.

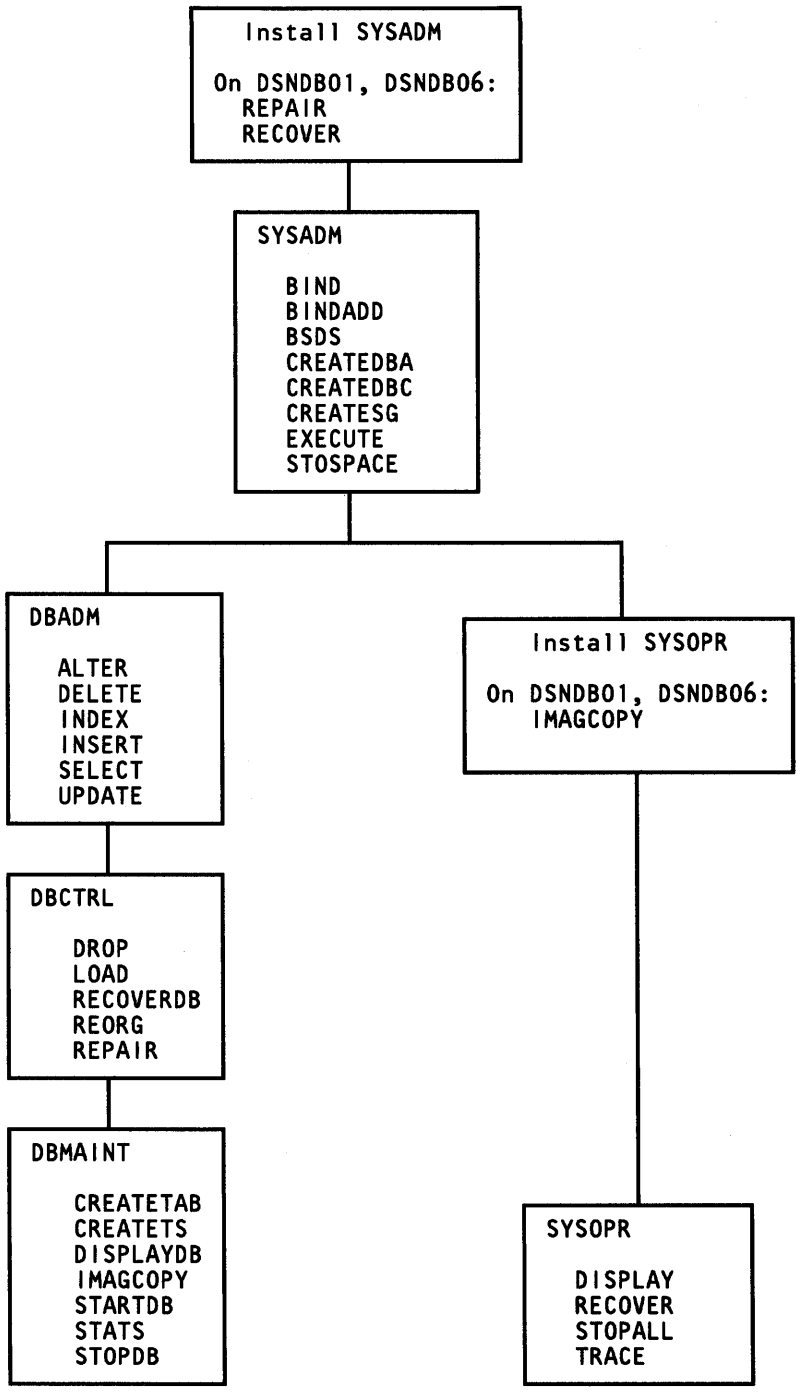


Figure 60. Individual Authorities of Administrative Authorities

Authorities Held by the Creators of DB2 Objects

Users without a special administrative authority can create an object if they have been granted the authority to create that type of object. Regardless of how DB2 users gain the authority to create objects (tables, table spaces, indexes, and so forth), they have access to the objects they create. For example, the creator of a table can alter and drop the table; create an index or a view for it; and insert, update, select, and delete rows in it. However, users' access to their objects is not necessarily total. For

example, users without special administrative authority cannot -DISPLAY, -START, -STOP, RECOVER, REORG, REPAIR, issue an IMAGCOPY, or run statistics against their objects without having been granted these privileges explicitly. If you have been granted DBMAINT privileges, you have implicitly been given the authority to -START, -STOP, and -DISPLAY.

Users who do not have special administrative authority to create data bases can be granted that privilege on two levels. For more information, see "Grant or Revoke Subsystem Privileges" on page 125, or refer to *IBM DATABASE 2 SQL Reference* for GRANT statement syntax.

Granting and Revoking Authorization

All users of DB2 are identified by authorization IDs. A single user can have more than one authorization ID. The appropriate privilege must be granted to a particular user's authorization ID for that user to use any DB2 capability.

DB2 defines and enforces the manner in which a data base can be used. Some of the ways users can affect a DB2 data base are as follows:

- By actions performed on tables (retrieving, inserting, updating, and deleting information) using SQL statements.
- By using data definition statements to create, alter and drop DB2 objects.
- By using utilities that reorganize, recover, repair, or load tables in the data base.
- By using DB2 commands that start, stop, and otherwise control the data base.

The two SQL authorization statements GRANT and REVOKE provide a means to control the ways that a user can interact with a DB2 data base.

In DB2, anyone who creates an object is implicitly given full authorization to access that object. The authorization ID entitles the user to ownership of the data as well as certain privileges.

In addition, any privileges on the object can be granted to another user. Also, at the creator's option, privileges can be granted such that the recipient may in turn grant privileges to other users.

Using the GRANT and REVOKE Statements

The GRANT statement enables you to give users certain privileges in DB2. The REVOKE statement enables you to take them away. Only a privilege that has been specifically granted can be revoked.

Implicit privileges cannot be revoked. For example, suppose SYSADM grants user "A" the privilege to create a table within a data base. User "A" creates table "TBL1" and inserts information into the table. SYSADM then attempts to revoke from user "A" the privilege to insert into "TBL1." Such a revoke will fail, generating a -556 SQL code. This is because the privilege to insert information into a table you have created is implicit and cannot be revoked.

You can grant and revoke privileges to and from a single authorization ID, or you can specify several authorization IDs on one statement. Instead of granting privileges to authorization IDs, you can also grant privileges to PUBLIC, which means that all DB2 users are granted those privileges. PUBLIC is a special authorization ID used by DB2. No user of DB2 should have PUBLIC as an authorization ID. When a privilege

is revoked from PUBLIC, authorization IDs that were specifically granted that privilege will still have that privilege.

The general form of GRANT is:

```
GRANT some_privilege ON some_resource TO some_authorization_ID
```

There are five forms of the SQL GRANT statement, and five forms of the corresponding REVOKE statements. Each form controls a separate set of privileges: table, plan, subsystem, data base, and use privileges. The syntax of each statement is described in *IBM DATABASE 2 SQL Reference*. The following sections describe the capabilities of each form.

Grant or Revoke Table Privileges

This form includes these capabilities:

- Retrieve data from a table or view (SELECT)
- Insert new rows in a table or view (INSERT)
- Delete rows from a table or view (DELETE)
- Update columns of a table or view (UPDATE)
- Create indexes for columns of a table (INDEX)
- Alter a table (ALTER)

To grant someone all these privileges, use the ALL parameter.

The following examples show how the GRANT and REVOKE statements are used to grant and revoke privileges on the DB2 sample table DSN8130.TEMPL.

- Authorize all users to retrieve data from the DSN8130.TEMPL table.

  ```
  GRANT SELECT
    ON DSN8130.TEMPL
    TO PUBLIC;
  ```

- Give update privileges (on the JOBCODE and SALARY columns of the table) to users KWAN, GEYER, and STERN. You cannot grant column UPDATE privileges WITH GRANT OPTION.

  ```
  GRANT UPDATE (JOBCODE,SALARY)
    ON DSN8130.TEMPL
    TO KWAN,GEYER,STERN;
  ```

- Authorize WALKER to add new columns to the table and also to insert and delete rows. Allow WALKER to pass these privileges to other users.

  ```
  GRANT ALTER,INSERT,DELETE
    ON DSN8130.TEMPL
    TO WALKER
    WITH GRANT OPTION;
  ```

- Because WALKER is allowed to pass the above privileges to other users, suppose ALTER privileges are granted to LUTZ:

  ```
  GRANT ALTER
    ON DSN8130.TEMPL
    TO LUTZ;
  ```

  Because the WITH GRANT OPTION was not specified, LUTZ can't pass the ALTER capability to anyone else.

- Authorize all table privileges that you have with grant option to NICHOLLS. Don't allow the privileges to be passed to others.

```
GRANT ALL
  ON DSN8130.TEMPL
  TO NICHOLLS;
```

- REVOKE Stern's UPDATE privileges:

```
REVOKE UPDATE
  ON DSN8130.TEMPL
  FROM STERN;
```

  Note that UPDATE privileges to all columns are revoked. You can't revoke UPDATE privileges for specific columns.

- REVOKE the table privileges given to WALKER:

```
REVOKE ALL
  ON DSN8130.TEMPL
  FROM WALKER;
```

  Even though WALKER doesn't possess ALL privileges on the DSN8130.TEMPL table, REVOKE ALL will revoke the privileges that WALKER *does* have which you granted. You can use REVOKE ALL, provided the authorization ID in question has at least one table privilege.

  The above REVOKE example illustrates the cascading effect of the REVOKE statement: Not only will WALKER lose the privileges WALKER has, but, because WALKER passed the ALTER privilege to LUTZ, LUTZ will lose that privilege also (provided that LUTZ didn't also obtain that privilege from another source).

Grant or Revoke Plan Privileges

This form includes the following capabilities:

1. Use of the BIND, REBIND, or FREE subcommands of DSN to replace, rebind, free, or change ownership of the specified plans (BIND)
2. Run programs associated with the specified application plans (EXECUTE)

Examples:

- Allow SMITH to replace, rebind, or free application plan DSN8BC13:

```
GRANT BIND
  ON PLAN DSN8BC13
  TO SMITH;
```

  Note: If SMITH now issues a BIND (REPLACE) statement, ownership of the plan will transfer to SMITH.

- Allow SETRIGHT to run the application program associated with the above plan.

```
GRANT EXECUTE
  ON PLAN DSN8BC13
  TO SETRIGHT;
```

Grant or Revoke Subsystem Privileges

This form includes the following capabilities:

- Grant or revoke SYSADM and SYSOPR administrative authorities
- Issue these operator commands:

  -DISPLAY DATABASE

-DISPLAY THREAD
-RECOVER INDOUBT
-RECOVER BSDS
-START TRACE
-STOP DB2
-STOP TRACE

- Create data bases and have DBADM authority over the data bases created (CREATEDBA)
- Create data bases and have DBCTRL authority over the data bases created (CREATEDBC)
- Create new application plans (issue the BIND statement with ADD parameter) (BINDADD)
- Create storage groups (CREATESG)
- Use the STOSPACE utility (STOSPACE)

Examples:

- Allow THOMPSON to create data bases and have data base administrative (DBADM) authority over the data bases created:

```
GRANT CREATEDBA
  TO THOMPSON;
```

- Allow MEHTA to create new application plans; that is, be able to issue the BIND command.

```
GRANT BINDADD
  TO MEHTA;
```

- REVOKE THOMPSON's authority to create data bases:

```
REVOKE CREATEDBA
  FROM THOMPSON;
```

  The above revoke statement will prevent THOMPSON from creating any more data bases, but THOMPSON will retain DBADM authority over any data bases he or she may have already created.

Grant or Revoke Data Base Privileges

This form includes the following capabilities:

- DBADM, DBCTRL, and DBMAINT administrative authorities
- Operator commands to start, stop, and display a data base (STARTDB, STOPDB, DISPLAYDB)
- Create a table (CREATETAB)
- Create a table space (CREATETS)
- Drop a data base (see note below) (DROP)
- Use DB2 utilities to:
  - Produce image copies and merged image copies (IMAGCOPY)
  - Load table spaces and indexes (LOAD)
  - Reorganize table spaces and indexes (REORG)
  - Recover table space and indexes (RECOVERDB)
  - Repair table spaces and indexes (REPAIR)
  - Run the statistics utility (STATS)

This form of the GRANT statement does not include all the privileges that may be needed for data base administration. For some tasks you also need the privileges to use storage groups and buffer pools; for those, see "Grant or Revoke Use Privileges" on page 128.

When you grant the DROP privilege it means that the authorization ID to whom you grant it can drop the specified data base. When a data base is dropped, all the objects in the data base are also dropped. However, having this authority doesn't mean that the grantee can drop individual objects in the data base. For example, suppose a DB2 data base XYZ contains two table spaces named A and B. A contains tables C and D, and table space B contains a table E. Now suppose you grant Jones the authority to drop the XYZ data base:

```
GRANT DROP ON DATABASE XYZ TO JONES;
```

Jones now has the authority to issue the following statement:

```
DROP DATABASE XYZ;
```

The above DROP statement will cause XYZ and the objects A, B, C, D, and E to be dropped. However, the above GRANT statement does not give Jones the authority to drop A, B, C, D, or E individually. There is no implicit authorization to issue, for example, the following:

```
DROP TABLE D;
```

Objects in a data base can be dropped by their creator, someone with SYSADM authority. Objects other than views and storage groups can be dropped by someone with DBADM authority for the data base.

Here are some examples of using this form of GRANT and REVOKE:

- GRANT data base administrative authority (DBADM) on the DSN8D13A data base to JEFFERSON:

  ```
  GRANT DBADM
    ON DATABASE DSN8D13A
    TO JEFFERSON;
  ```

- Permit PEREZ to create tables and table spaces in the DSN8D13A data base:

  ```
  GRANT CREATETAB,CREATETS
    ON DATABASE DSN8D13A
    TO PEREZ;
  ```

- Authorize several users to use the REORG and LOAD utilities:

  ```
  GRANT REORG,LOAD
    ON DATABASE DSN8D13A
    TO JOHNSON,PEREZ,SCHNEIDR,MARINO,PIANKA;
  ```

- REVOKE JEFFERSON's DBADM authority:

  ```
  REVOKE DBADM
    ON DATABASE DSN8D13A
    FROM JEFFERSON;
  ```

  While JEFFERSON had DBADM authority over the DSN8D13A data base, JEFFERSON could have (for example) created one or more tables. This is because that privilege is implied in DBADM authority. However, the following REVOKE statement would be rejected:

  ```
  REVOKE CREATETAB
    ON DATABASE DSN8D13A
    FROM JEFFERSON;
  ```

The statement will be rejected because there is no record that JEFFERSON was ever granted CREATETAB authority.

Grant or Revoke Use Privileges

This form includes the following capabilities:

1. Authority to use a buffer pool
2. Authority to use a storage group
3. Authority to use a table space

Examples:

- Allow STERN to use the DSN8S13E table space and pass the privilege on to others:

```
GRANT USE OF TABLESPACE DSN8D13A.DSN8S13E
  TO STERN
  WITH GRANT OPTION;
```

- Allow all users to use the DSN8G130 storage group:

```
GRANT USE OF STOGROUP DSN8G130
  TO PUBLIC;
```

Implicit Privileges

Users can perform certain actions without receiving authority through the GRANT statement. The appropriate authorization IDs possess these privileges implicitly. For example, some IDs have implicit authority to:

- Drop a table. A table can be dropped only by its creator, someone who has SYSADM authority, or by someone who has DBADM authority over the data base that contains the table.
- Drop or alter a table space or index. Again, only the creator, someone with SYSADM authority, or someone with DBADM authority over the data base can drop or alter a table space or index.
- Authority to terminate the execution of a DB2 utility. Only the original submitter of the utility job or someone with SYSADM or SYSOPR authority may issue a -TERMINATE command to terminate the execution of a utility.
- Drop a storage group. Only the creator or SYSADM can drop it.
- Create a synonym. (A synonym is an alternative name for a table or view.) Any DB2 user may create a synonym.
- Drop a synonym. Only the creator can drop it.
- Create a comment on a table, view, or column. Only the creator of a table or view, or someone with SYSADM authority (for tables or views) or DBADM authority (for tables) over the containing data base may issue a COMMENT ON statement.
- Use the LOAD utility to load a data base. This means that the user can load any table in the data base. However, when a table is created, the person creating the table is implicitly granted the capability to use the LOAD utility to load it, whether or not the creator has LOAD authority for the data base containing the table.

- Use EXECUTE with the GRANT option for to bind an existing plan. Because plans can be bound only by using the TSO attachment facility, programs running in the IMS/VS and CICS environments might be run in that environment under a different authorization ID. If that is true, the programmer who binds a plan containing IMS/VS or CICS programs in TSO will have to obtain explicit EXECUTE authority under the second authorization ID he or she is using in the IMS/VS or CICS environment.

For more information about binding and executing programs, see the *IBM DATABASE 2 Application Programming Guide*.

The Cascading Effect of the REVOKE Statement

The WITH GRANT OPTION clause of the GRANT statement allows a user obtaining a capability to pass the same capability to other users. This property of the GRANT statement can cause the REVOKE statement to have side effects that are not immediately discernible. It's possible for one REVOKE statement to trigger many other revokes. This is particularly true with regard to revoking capabilities that have passed through several levels of GRANT statements.

Only an authorization ID that has granted a capability or an authorization ID having the appropriate administrative authority may revoke a previously granted privilege. When a privilege is revoked from authorization ID X, that privilege is also automatically revoked from all authorization IDs to whom X granted it, unless they have some source for the capability other than X. In the previous section, we saw how WALKER passed ALTER privileges on the DSN8130.TEMPL table to LUTZ. When WALKER's ALTER capability was revoked, LUTZ's ALTER capability was also revoked. Suppose, however, that LUTZ had also obtained ALTER privileges on DSN8130.TEMPL from JEFFERSON. (JEFFERSON can do this because he or she has DBADM authority.) Now, when WALKER's ALTER capability is revoked, LUTZ will not lose that capability because the ALTER privilege was also granted by another source.

Whenever a privilege is revoked from X by Y, DB2 checks to see if X should retain that privilege. (See Figure 61 on page 130 for an illustration of the following scenario.) X will retain it if X was granted the same privilege by another source, for example, authorization ID B. If X had the privilege from only the one source, Y, then X will lose it.

Y's action may affect other IDs, C and D, to whom X passed on the same privilege. If X did not retain the privilege, IDs C and D will also lose it, unless they had received it from another source. If X retained the privilege, the situation is more complicated. DB2 will check to see if X could have granted the privilege to C and D relying only on B's authorization, as if Y had never granted X authority. C and D will keep their authorization if Y's authority was not needed. Otherwise, they will lose the privilege from X, unless they had received it through an alternative source.

The way DB2 determines whether or not X could have granted the privilege is by comparing the time at which X received the privilege with the time at which X granted the privilege to someone else.

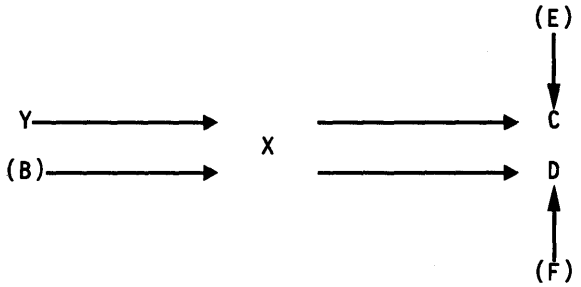


Figure 61. Authority of X, C, and D Affected by Y, B, E, and F

As another example, suppose JONES grants a privilege to BROWN at a certain time and date (TIME1). BROWN then grants the capability to SMITH at a later time (TIME2). At a third time (TIME3), BROWN is granted the same capability by PARKER. See Figure 62.

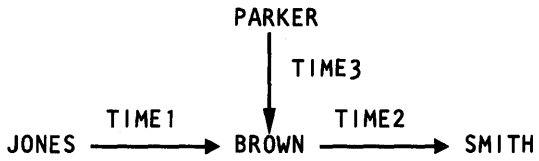


Figure 62. Revoking Authorization (Example 1)

If PARKER subsequently revokes the capability from BROWN, SMITH doesn't lose the capability. The reason is that BROWN granted the capability to SMITH *before* PARKER granted it to BROWN. Thus BROWN *could* have granted the capability to SMITH without ever receiving it from PARKER.

If, on the other hand, JONES, instead of PARKER had revoked the capability from BROWN, SMITH would lose it as well. This is because BROWN granted the capability to SMITH *after* receiving it from JONES but *before* receiving it from PARKER. Therefore, BROWN couldn't have granted it to SMITH if JONES had never granted it to BROWN.

Take another example: Suppose WALKER and LUTZ grant the same privilege to PEREZ at one time (TIME1) and another (TIME2) respectively. PEREZ then passes the capability on to LEE at a third time (TIME3). See Figure 63.

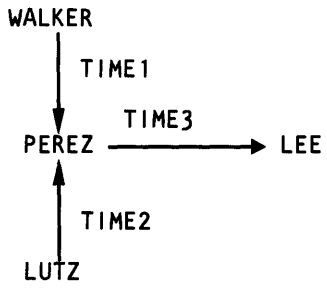


Figure 63. Revoking Authorization (Example 2)

When PEREZ grants the capability to LEE, PEREZ doesn't have any way of specifying the *original* source of the capability being granted. Therefore, if either WALKER or LUTZ (but not both) revoked the capability from PEREZ, LEE would still retain it because DB2 would check in the catalog and find that PEREZ received the capability before granting it to LEE.

Another example will give you more insight into the way REVOKE works: Suppose a user with authorization ID X wants A to perform some function. In order to do so, A must be granted capabilities 1, 2, 3, 4, and 5. Y also wants A to perform some function. In order to do so, A must be granted capabilities 4, 5, 6, 7, 8, and 9. Note that X and Y both granted capabilities 4 and 5 as shown in Figure 64. When A finishes X's assignment, X revokes from A all the capabilities granted previously (1, 2, 3, 4, and 5). But, because Y also granted capabilities 4 and 5, A can continue to do Y's assignment because he or she retains capabilities 4 through 9.

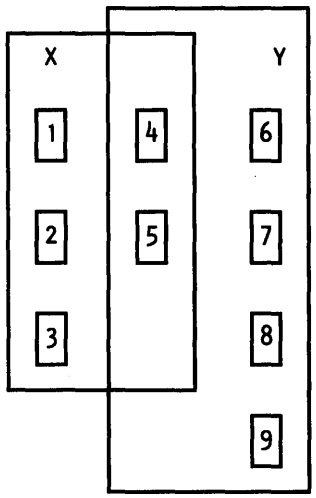


Figure 64. Revoking Authorization (Example 3)

Other Implications of the REVOKE Statement

View Deletion

When a view is created, DB2 inserts information in the DB catalog table, SYSIBM.SYSTABAUTH. When a DB2 user's authorization to use a table changes, DB2 checks the SYSIBM.SYSTABAUTH table to see whether the user created a view based on that table. If a privilege is lost from the table, it will also be lost from the view. If, however, a privilege is gained on the table, it will not also be gained for the view. If, as a result of this update, the user no longer has SELECT privileges on the view, DB2 deletes the view. (This only occurs when the SELECT privilege that was used to create the view is revoked.)

For example, suppose SCHNEIDR has SELECT and INSERT privilege on the DSN8130.TEMPL table and creates a view of it. If SCHNEIDR loses INSERT privilege on the table, SCHNEIDR also loses that privilege on the view. If SCHNEIDR loses SELECT privilege on the DSN8130.TEMPL table, DB2 will delete the view.

Application Plan Invalidation

When an application plan refers to a table or a view, DB2 inserts information in the SYSIBM.SYSTABAUTH catalog table. The information includes the plan name, the table (or view) name, and the action performed on the table or view (SELECT, INSERT, UPDATE, or DELETE). Now suppose a user who has SELECT authority on a table creates an application plan that contains a SELECT operation on the table. If the user loses SELECT authority on the table, DB2 will invalidate the application plan. If that user loses INSERT authority on the table, DB2 won't invalidate the application plan, because the plan doesn't involve an insert operation on the table.

For more information about invalidated plans, see the *IBM DATABASE 2 Application Programming Guide.*

Authorizing the Use of Default Objects

Chapter 3 of *IBM DATABASE 2 System Planning and Administration Guide* discusses the DB2 default objects (for example, default data base DSNDB04, storage group SYSDEFLT, and buffer pool BP0). DB2 will use a default object if you don't specify a particular one. These defaults are established during DB2 install and migration, and all DB2 users are granted access to them automatically.

Also established during install and migration is the authorization for all users to create table spaces and tables in the default data base. This enables a user to create a table without first having to create a table space. DB2 will automatically generate a table space in the default data base.

If you decide that certain users should not be granted access to these default objects, you can revoke the authority from PUBLIC and then grant the authority to whomever should have it. This revocation can only be done by someone who has system administration (SYSADM) authority.

Authorities Needed for Application Development

Programmers writing and compiling programs that access DB2 data do not need any special privileges. But they do need to be granted the authority to:

- Test SQL statements with SPUFI
- Bind and execute plans
- Test with existing tables
- Create tables
- Run utility jobs

Distributing Authority: Centralized vs. Distributed Authorization

Data processing sites differ in the way users are allowed access data. DB2 allows you to distribute authority to best suit your security objectives. Some sites maintain tight control over the data base, with authority residing with a central system administrator. Other sites distribute authority down to the departmental level. Many sites fall somewhere in between, placing authority in the control of a data base administrator.

Three security scenarios are described below. The first scenario describes how authority is distributed during the development of a new DB2 application. The first scenario describes a site in which all authority is centralized. The second scenario describes the modification of an existing application in a site where authority is not as centralized as in the first scenario. In this scenario, authority is delegated to a data base administrator who controls access to the data base while the application program is being modified. The third scenario describes a site in which authority resides at the departmental level.

Scenario 1. Tightly Controlled New Application

All authority is centralized and controlled by the authorization ID that has SYSADM administrative authority. A new application is to be developed. SYSADM creates the following:

- A storage group (TIGHTSG)
- A data base (TIGHTDB) with TIGHTSG as its default storage group.
- A table space (TIGHTTS) in the above data base with the necessary amount of DASD space.

SYSADM next grants certain capabilities to an authorization ID (TIGHTDBA) who performs the data base administration function for the TIGHTDB data base. The capabilities granted are:

- Authority to create tables in the TIGHTDB data base (CREATETAB privilege)
- Authority to create new application plans (BINDADD privilege)
- Authority to use the TIGHTTS table space (USE OF TABLESPACE privilege)

On being granted the above capabilities, TIGHTDBA does the following to facilitate the development of the application:

- Creates tables (with the CREATE TABLE statement), placing them in the TIGHTTS table space. In this case, assume that only one table (TIGHTTABLE) is necessary.
- Grants the application programmer (APPLPROG) the following privileges on TIGHTTABLE:

  SELECT
  INSERT
  UPDATE
  DELETE

APPLPROG develops the application and precompiles it. This scenario assumes that the program uses qualified table names.

TIGHTDBA now performs the initial BIND operation, creating a new application plan (TIGHTAPP). After this, TIGHTDBA authorizes APPLPROG to test the application by granting BIND and EXECUTE on TIGHTAPP.

APPLPROG tests the application program using test data. Note that BIND REPLACE, REBIND, and FREE commands can be used against TIGHTAPP.

When testing is complete, TIGHTDBA takes control again and does the following:

- Revokes all table privileges from APPLPROG
- Revokes BIND and EXECUTE privileges from APPLPROG
- Rebinds the plan to authorize access to production data
- Loads TIGHTTABLE with production data
- Grants EXECUTE authority on plan TIGHTAPP to one or more operators

Scenario 2. Delegated Authority When Modifying an Existing Application

Authorization in this environment is not as tightly controlled as in Scenario 1. This scenario features a modification to an existing application. It assumes unqualified table names are used.

The central authorization body grants more control to the data base administration function, which in turn grants more control to the application programmer.

SYSADM grants the following capabilities to an authorization ID (EXISTDBA), who performs the data base administration function for an existing data base (EXISTDB):

- DBADM administrative authority on EXISTDB with the ability to pass it to others (WITH GRANT OPTION).
- Authorize EXISTDBA to create new applications plans with the ability to pass this capability on to other users.

EXISTDBA next grants the application programmer (APPLPROG) the following privileges:

- Authority to create new application plans
- Authority to create table spaces and tables in the EXISTDB data base
- Authority to use buffer pools

APPLPROG performs the following activities in connection with developing the application:

- Creates test tables
- Loads test data
- Develops the application
- Precompiles the application program
- Binds the application, creating a new (test) application plan
- Tests the application program
- Rebinds the plan
- Drops the test tables
- Frees the application plan

EXISTDBA now takes over control of the application and does the following:

- Revokes BINDADD capability from APPLPROG
- Revokes APPLPROG's capability to create table spaces and tables in the EXISTDB data base
- Binds the application, creating a new application plan (MODAPPL)
- Tests MODAPPL against production data
- Grants EXECUTE authority on MODAPPL to one or more operators

Scenario 3. Departmental Distribution

In this scenario, control is distributed. DB2 resources are made available to different departments in the enterprise, and each department operates autonomously. Data base housekeeping chores are done for each department by an authorization ID with DBCTRL administrative authority.

The authorization ID with SYSADM administrative authority makes DB2 resources available for each department. SYSADM does the following for a particular department (DEPTX):

- Creates a storage group (DEPTXSG)
- Creates a data base (DEPTXDB) with storage group DEPTXSG and a particular buffer pool (say, BP1) as the default.
- Grants DBCTRL administrative authority to an authorization ID (DEPTXDBC), a member of DEPTX.
- Grants DEPTXDBC the authority to use storage group DEPTXSG and buffer pool BP1
- Grants DEPTXDBC the capability to create new application plans

SYSADM grants all the above capabilities *WITH GRANT OPTION*, so that DEPTXDBC can pass these capabilities on to other members of the department.

DEPTXDBC now grants the capability to create tables in the DEPTXDB data base to each member of the department. DEPTXDBC also authorizes each member of the department to use storage group DEPTXSG and buffer pool BP1. Finally, each member of the department is granted the capability of creating new application plans (BINDADD privilege). All these capabilities are granted without grant option.

Each department member can now develop DB2 applications using the DEPTXDB data base. Tables are created using the IN DATABASE DEPTXDB clause. This allows department members to develop their applications without having to be concerned with table spaces, storage groups, or buffer pools. Note, however, that because the default for space allocation is minimal, only small tables can be created.

Because DEPTXDBC has DBCTRL authority, functions like loading, repairing, recovering, reorganizing, and copying the data base are allowed, but the data in tables created by department members cannot be accessed unless these accesses are authorized. Note further, that members of the department have the power only to create tables in the DEPTXDB data base.

Creating Objects for Another User

If you have SYSADM, DBADM, or DBCTRL administrative authority, you can create a DB2 table on behalf of another user. As far as DB2 is concerned, the other user is the creator of the table and has all capabilities. For example, suppose you have DBADM authority, and you create a table for a user whose authorization ID is BROWN. BROWN now becomes the "owner" of the table and can load data into it, retrieve data from it, create views of it, drop it, or perform insert, delete, and update operations against it.

When creating a table for someone else, you qualify the table name with the authorization ID of the intended user. The following is an example of someone with DBADM or DBCTRL authority creating a table for BROWN:

```
CREATE TABLE BROWN.TEMPL
  (EMPNO CHAR(6)          NOT NULL,
   LASTNAME VARCHAR(15)   NOT NULL,
   DEPT CHAR(3)           NOT NULL,
   HIREDATE DECIMAL(6)    NOT NULL,
   JOBCODE DECIMAL(3)             ,
   EDUC SMALLINT);
```

If you created the above table and you had DBADM authority, you would have complete access to its contents. If, however, you had only DBCRTL authority and you created the table, you *wouldn't* be able to manipulate the data in it.

Only a user with SYSADM authority can create views for other users.

Anyone who creates a table can create a view of it. Also, a user can create views based on tables created by others, provided the user has the SELECT privilege. The creator of a view is granted only the capabilities on the view that he or she holds on the table on which it is based. Also, the creator of the view can use the view mechanism to hide sensitive data from users. This can be accomplished by creating (1) a subset of the rows of a table, (2) a subset of the columns of a table, or (3) a subset of a combination of rows and columns of a table.

A few examples will illustrate how you can use views to control access to the data base. In the first example, the manager of Department B01 (Michael Thompson) needs to see the data from the DSN8130.TEMPL table that gives him SALARY, NAME, and EMPNO information about people *in his department only.* Here's how you could create the view and grant him the authority to use it:

```
CREATE VIEW DBADM.VM80EMPSAL AS
  SELECT LASTNAME, SALARY, EMPNO
  FROM DSN8130.TEMPL
  WHERE WORKDEPT = 'B01';

GRANT SELECT ON DBADM.VM80EMPSAL TO THOMPSON;
```

The next example shows how SYSADM could create a view for a user whose job involves updating the DSN8130.TEMPL table to reflect changes to employee work departments. The user needs to see only two columns, employee number and work department; the rest of the table should be hidden. Here's the view, assuming the user's authorization ID is Jones:

```
CREATE VIEW JONES.NAMELOC AS
  SELECT EMPNO, WORKDEPT
  FROM DSN8130.TEMPL;
```

The above two views are for the use of specific users. The next is an example of creating a view for general usage. You want to create a view of the DSN8130.TEMPL table, and you want to make it available to anyone in your organization. However, because the salary (SALARY), job code (JOBCODE), and date of birth (BRTHDATE) columns contain sensitive information, these three columns should be omitted from the view:

```
CREATE VIEW DBADM.VEMPL AS
  SELECT EMPNO,FIRSTNME,LASTNAME,WORKDEPT,
  HIREDATE,EDUCLVL,SEX
  FROM DSN8130.TEMPL;

GRANT SELECT ON DBADM.VEMPL TO PUBLIC;
```

Deleting a View and Its Authorization

You can delete views using the DROP statement. To delete the view created in the previous example, you can use the following statement:

```
DROP VIEW JONES.NAMELOC;
```

Dropping a view doesn't affect the table (or tables) on which the view is based. In other words, if NAMELOC were dropped, nothing would happen to DSN8130.TEMPL. However, when a view or table is dropped, DB2 will also automatically drop all views that are dependent on the view or table being dropped. For example, suppose a view named XYZ is based on the DEPT table and the NAMELOC view. If either DEPT or NAMELOC is dropped, XYZ will be dropped also. Because of this, check the SYSIBM.SYSVIEWDEP table in the DB2 catalog before you drop a table or view. This table will tell you if any views are dependent on the object you intend to drop.

(For dependent plans, check SYSIBM.SYSPLANDEP.) For example, if you wanted to drop the DSN8130.TEMPL table, the following query would give you the names of the views that are dependent on DSN8130.TEMPL:

```
SELECT BNAME
  FROM SYSIBM.SYSVIEWDEP
  WHERE BNAME = 'TEMPL' AND BTYPE = 'V'
  AND BCREATOR = 'DSN8130';
```

If you also wanted to know who created each dependent view, this query would tell you:

```
SELECT BNAME,BCREATOR
  FROM SYSIBM.SYSVIEWDEP
  WHERE BNAME = 'TEMPL' AND BTYPE = 'V'
  AND BCREATOR = 'DSN8130';
```

Controlling Access to the DB2 Subsystem

This section explains how DB2 controls access to the subsystem. When a connection request is received, DB2 obtains an authorization ID for the user and verifies that the user can access the subsystem.

The method used to establish and verify the authorization ID varies according to whether the requester's environment is IMS/VS, CICS, TSO, or batch. This section discusses the relationship between DB2's access security mechanism and the access security provided by all of these environments. It also explains how to use security mechanisms of other subsystems to help ensure that the authorization ID that DB2 receives represents a verified user. Most of this discussion centers around the use of the Resource Access Control Facility (RACF).

RACF provides access control to MVS and any of its subsystems by:

- Identifying and verifying system and subsystem users
- Authorizing access to system and subsystem resources
- Logging and reporting of unauthorized attempts to access the system, subsystem, and protected resources

To control access to DB2 using RACF, your site identifies a DB2 subsystem as a RACF-protected resource. DB2 invokes RACF to verify that subsystems and users connecting to DB2 are authorized to do so. DB2's authorization mechanisms then ensure proper individual user access of those resources. In this way, RACF participates indirectly in protecting individual DB2 resources such as tables, table spaces, and indexes.

For more information about RACF capabilities, see *Resource Access Control Facility (RACF) General Information Manual.*

Connection Verification

DB2 allows a site to control which applications can connect to a DB2 subsystem. DB2 does not itself provide a user signon with password verification. It assumes that the site has existing security mechanisms to verify user identifications and associate the verified authorization ID with the address space.

When a subsystem application or user requests a connection, DB2 invokes RACF to verify that the requester is authorized to a resource type recognized by RACF (RACF refers to the resource type as the "CLASS"). For DB2, the resource type, or CLASS is DSNR.

RACF also checks if the DB2 subsystem name and connection type are authorized. (RACF refers to the connection type as the "ENTITY.") The DB2 subsystem name is a one- to four-character name established by the site. The formats for the subsystem name and connection type for each execution environment are:

- *ssnm*.BATCH—used by TSO and call attachment facility
- *ssnm*.MASS (multiple address space subsystem)—used by IMS/VS
- *ssnm*.SASS (single address space subsystem)—used by CICS

where *ssnm* represents the name assigned by the site to the DB2 subsystem.

The source of an authorization ID associated with a connection request depends upon the address space type.

Figure 65 lists the authorization ID origin for each case.

Address Space Type	Authorization ID Origin
TSO (single or multiple thread connection)	TSO Logon ID
BATCH (single or multiple thread connection)	USER parameter on JOB statement
IMS/VS Control Region	USER parameter on JOB statement or entries from the RACF table for started procedures (ICHRIN03)
CICS	USER parameter on JOB statement or entries from the RACF table for started procedures (ICHRIN03)

Figure 65. Sources of Authorization Identifiers

Figure 66 on page 139 describes the process of establishing an authorization ID and, if RACF is active, verifying that the authorization ID may connect to DB2.

If RACF is operational, the RACF user ID associated with the address space of the requester must be authorized to the "CLASS" and "ENTITY" shown. If RACF accepts the request, the authorization ID is verified. If RACF rejects the request, the connection is rejected. See *IBM DATABASE 2 Install Guide* for information on how to establish RACF protection for DB2 subsystems.

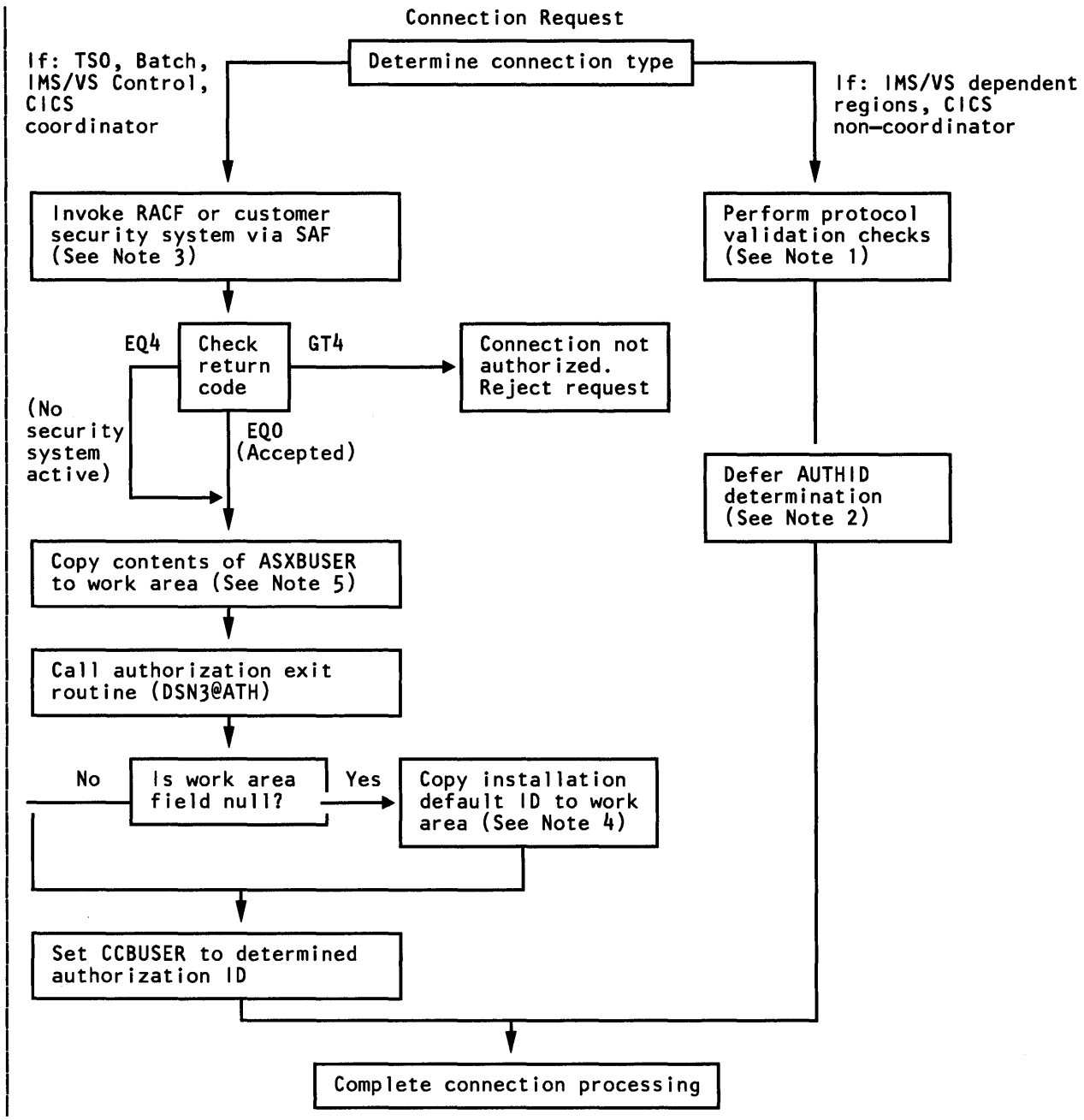


Figure 66. DB2 Connection Authorization Processing

Notes to Figure 66:

1. For IMS/VS dependent region and CICS non-coordinator connection requests, the following connection requirements are validated :
   - The recovery coordinator task must already be connected.
   - The program status word (PSW) key of the requestor must match the recovery coordinator's PSW key.
   - If this is a CICS connection, the requesting task control block (TCB) must be a child (or descendant) of the coordinator TCB.

2. The establishment and verification of authorization IDs for IMS/VS and CICS non-coordinator connections is deferred until just prior to allocation of a plan for a transaction. This authorization ID is provided by the requesting subsystem and has been authenticated by it.
3. RACF is invoked indirectly through SAF (System Authorization Facility). The coding of the invocation process is:

```
RACROUTE REQUEST=AUTH,
     REQSTOR=IDENTIFY,
     SUBSYS=ssnm,
     CLASS=DSNR,
     ENTITY=ssnm.zzzz
```

   where *ssnm* is the name of the invoking DB2 subsystem, and *zzzz* is one of the connection types, MASS, SASS, or BATCH.
4. If, upon return from the authorization exit, the authorization ID is null (that is, the first character is X'40' or less), DB2 uses the default authorization ID specified during the ISPF tailoring session.
5. If the authorization ID was verified by RACF, the field ASXBUSER in the MVS address space extension block will contain the authorization ID, padded on the right with blanks (X'40'). Otherwise, ASXBUSER will contain blanks.

The DB2-Supplied Authorization Exit Routine

DB2 invokes an authorization exit (module DSN3@ATH) to allow you to inspect and/or change the authorization ID of the connecting application. An IBM-supplied authorization exit is distributed with DB2.

The authorization exit routine is invoked if

1. RACF verifies the authorization ID, or
2. RACF is not installed or activated for this DB2 subsystem.

In either case, the authorization ID is passed to the exit routine, described in Figure 67 on page 141. You may replace the distributed routine with an authorization exit of your own, which may change the authorization ID if needed. The request to connect to DB2 will still be accepted.

See "Replacing the Authorization Exit" on page 141 for instructions on how to code your own authorization exit.

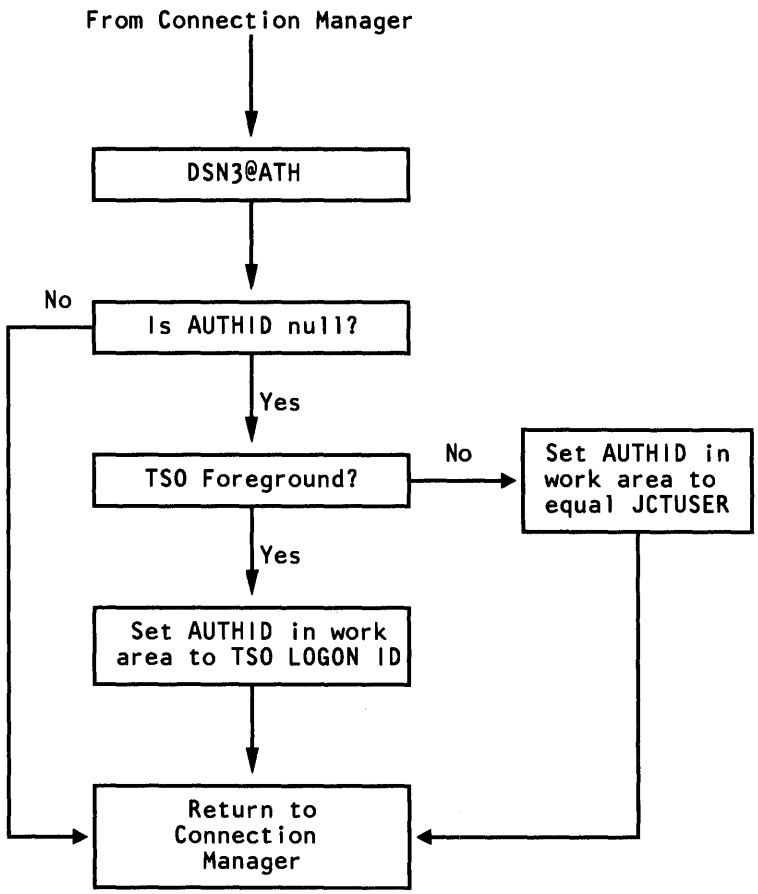


Figure 67. DB2-Supplied Authorization Exit Processing

The DB2-supplied authorization exit routine DSN3@ATH examines the passed authorization ID. If the first character of the authorization ID is greater than X'40', no action is taken. Otherwise, the authorization ID field in the work area is changed as follows:

- If the application is a TSO foreground user, the authorization ID is set to the TSO LOGON ID obtained from the field whose address is in ASCBJBNS.
- If the application is a batch job or started task, the authorization ID is set to the contents of the job control table (JCT) field "JCTUSER".

The authorization ID obtained from the JCT is the value specified on the USER parameter of the JCL JOB statement. This authorization ID is not verified because RACF could not be active. (This can be determined because ASXBUSER, the location where RACF places the authorization ID during verification, was null; therefore, RACF could not be active.) If the USER parameter was not supplied on the JOB statement, JCTUSER will be null. Thus the work area field also remains null.

Replacing the Authorization Exit

To replace the IBM-supplied authorization exit routine, users apply an SMP USERMOD. If an SMP USERMOD is undesirable, the replacement module may be linked into a user library either included in the MVS LINKLIST or concatenated to the STEPLIB in the started task JCL for ssnmMSTR ("ssnm" is the subsystem name, and DSN is the default). In any case, the library must have APF authorization. Conditions for installing the exit by this method are:

- The replacement module must be re-entrant.
- The module must have a CSECT name of DSN3@ATH.
- The module is loaded into commonly addressable, store protected key 7 storage during DB2 initialization.
- If the module is to be executed under MVS/XA, it must have the addressing and residency attributes "AMODE (31)" and "RMODE (ANY)".

DSN3@ATH is invoked by a standard CALL. Its execution environment is:

- Supervisor state
- Enabled for interrupts
- PSW key 7
- Non-cross memory mode
- No MVS locks held
- Under the TCB of the application requesting a DB2 connection.

The authorization ID passed to the exit is a copy of the ASXBUSER field (padded to the right with blanks). If RACF (or its equivalent) is installed and operational, this authorization ID has been verified and is non-null; otherwise, it is equal to X'40'.

The exit may change the authorization ID. Upon return, if the authorization ID is null (that is, the first character is X'40' or less), DB2 will use the installation-chosen default authorization ID. In all cases, the connection is accepted.

Figure 68 on page 143 shows the contents of the general purpose registers on entry to the user exit. All areas referenced are in key 7, fetch protected storage. Figure 69 on page 143 shows the expected contents of the general purpose registers on return from the exit.

Register	Data in Register
GPR 0	Undefined.
GPR 1	The address of a two-word parameter list in standard parameter list format (that is, a list of addresses, the last of which has the high-order bit on). There are two parameters: • The first contains the address of an exit parameter list (EXPL). • The second is a pointer to an 8-byte field containing the authorization ID extracted from ASXBUSER. The authorization ID field may be changed by the exit. If the changed authorization ID is less than 8 characters, left-justify the string in the field and pad unused characters with blanks (X'40'). The exit parameter list (EXPL) is mapped by an assembler language DSECT named DSNDEXPL and is included in the DB2-distributed MACLIB. To use the DSECT, include the macro DSNDEXPL in the exit routine, and establish addressability with a USING statement on the label EXPL. The fields in EXPL are: • EXPLWA is the address of a 512-byte double-word aligned work area that may be used by the exit routine. • EXPWL contains the length of the work area (512). • EXPLRC1 (return code) and EXPLRC2 (reason code) are ignored upon return from the routine.
GPR 2-12	Undefined
GPR 13	The address of a standard 18-word register save area
GPR 14	Return address
GPR 15	Entry point address of exit routine

Figure 68. General Registers Upon Entry to Authorization Exit

Registers	Data in Register
GPR 2-14	Must have been saved and restored by the exit.

Figure 69. General Registers Upon Return from Authorization Exit

Resource Authorization

Before an application plan can be allocated to a user, an authorization check is performed to verify that the user is authorized to execute the specified plan. An authorization check is also required for DB2 commands received from a connection or MVS console.

DB2 itself does not provide a user sign-on with password verification. It assumes that the site has existing security mechanisms to verify user identifications and to associate the verified authorization ID with the address space.

The DB2 Authorization ID

This section describes the choices a site can make to decide on an acceptable authorization ID.

TSO and Batch Connections

The authorization ID is determined by the DB2 connection manager, not the attachment facility. It is the same authorization ID established during connection processing.

IMS/VS Subsystem Connections

For message-driven regions, the authorization ID consists of the following elements (in order):

- Sign-on ID of the terminal user
- LTERM name
- ASXBUSER field (if RACF is present), or
- PSB name (if the ASXBUSER field is not valid).

For nonmessage-driven regions or message-driven regions where the sign-on ID or LTERM information is not available:

- ASXBUSER field (if RACF is present), or
- PSB name (if the ASXBUSER field is not valid).

Functions of IMS/VS security you can use to prevent unauthorized access to DB2 are:

- **Sign-on Security:** With sign-on security, individuals must be defined to IMS/VS before they are allowed to access it. Sign-on security is available through the Resource Access Control Facility (RACF) or through a user-written security exit. When a person signs on to IMS/VS, IMS/VS verifies that the person is authorized to use IMS/VS before the person is allowed access to the system. IMS/VS then passes the validated sign-on name to DB2 as the authorization ID.
- **Terminal Security:** Terminal security enables you to limit the entry of a transaction code to a particular logical terminal (LTERM) or group of LTERMs in the system. To protect a particular program, you can authorize a transaction code to be entered only from any terminal on a list of LTERMs. Alternatively, you can associate each LTERM with a list of the transaction codes that a user can enter from that LTERM. IMS/VS then passes the validated LTERM name to DB2 as the authorization ID.

CICS Subsystem Connections

To prevent a CICS user from accessing DB2 resources that have been granted to PUBLIC, you can use CICS transaction code security: only users who are authorized to access the DB2 transaction code can use DB2 resources.

In CICS, the authorization ID passed to DB2 is specified by authorization directives in the RCT. Any one of the following may be used:

- The operator ID from the SNT as related to the signed-on user (3 characters padded with blanks to 8 characters)
- The terminal ID (4 characters padded with blanks to 8 characters)
- The transaction ID (4 characters padded with blanks to 8 characters)
- The CICS user ID (8 characters)
- The CICS authorization name
- Character string up to eight characters in length

Controlling Access to DB2 Data

When DB2 is installed or migrated, the system administrator can specify several security options:

- Enable authorization: if authorization is not enabled, there is no security checking for DB2 data (except that users cannot use GRANT and REVOKE statements to change data in the DB2 system catalog tables).
- Use RACF protection for VSAM or MVS data sets used by DB2 or specify passwords for them. The specified passwords will be used as the VSAM or MVS passwords for DB2 system data sets: BSDS, active log, archive log, directory, catalog, and scratch data sets.
- Provide a user-tailored ID authorization exit. For information on how to replace the IBM-supplied exit, see "Replacing the Authorization Exit" on page 141.

Use of RACF or another security subsystem can play a major role in protecting DB2 data sets. Passwords can be an effective way of supplementing this security subsystem.

Passwords

Because DB2 table spaces and index spaces are stored in VSAM data sets, VSAM passwords can be used to protect the data stored in them. Someone defining a table space can specify a password for the VSAM data set(s) that support the table space.

Passwords for the VSAM data sets that contain DB2 table spaces and index spaces are stored in the ICF catalog used to catalog the data sets, and also in the DB2 system catalog.

DB2 uses VSAM passwords to open its system data sets as well as to extend them. To protect the DB2 system data sets, the system administrator must assign VSAM passwords to them when DB2 is installed. Passwords for the system data sets and the DB2 system catalog are stored in the BSDS.

In addition to data set passwords, DB2 must be provided with a password for each password-protected ICF catalog associated with a storage group definition. These

passwords will be used by DB2 in access method services DEFINE, ALTER, and DELETE commands. The catalog passwords are supplied in a CREATE STOGROUP statement and are stored in the SYSIBM.SYSSTOGROUP table of the DB2 system catalog.

Passwords also may be used to protect MVS data sets used by DB2, such as archive log data sets and libraries.

VSAM Passwords

You may use VSAM passwords for the bootstrap data sets, active log data sets, directory and catalog data sets, and data base data sets. Other data sets use MVS passwords.

Bootstrap Data Set

The BSDS is allocated to DB2 (when it is executing) and to the Print Log Map and Change Log Inventory utilities. The BSDS can also be allocated to a standalone log read job. Password processing and prompting are performed by MVS and are transparent to DB2.

The BSDS may contain passwords for the active and archive log data sets. Therefore, the BSDS must also be password protected by the procedures established at your site. Because the BSDS is opened by both the Change Log Inventory and Print Log Map utilities, and by the standalone read services, password verification of the BSDS is performed by MVS.

Active Log Data Sets

Information about an active log data set, including its password, is put in the BSDS by the Change Log Inventory Utility.

DB2 provides the password for an active log data set when the data set is opened, bypassing normal MVS operator intervention. When the standalone log read services program is executed, it uses the passwords contained in the BSDS if a BSDS DD statement is provided. Otherwise, it relies on password prompting when the log data set is opened if a DD statement describing the log is provided.

Directory and Catalog Data Sets

You may provide password protection for the DB2 directory and catalog data sets which you define during the ISPF tailoring session. If you do this, you may also specify the read/write password during the ISPF tailoring session (they all must have the same one).

Data Base Data Sets

You can include a password when you create a data base data set to be used by DB2. This must be the VSAM master password (MASTERPW). You may use one of two methods to establish VSAM passwords for data base data sets:

- You may define the data set and establish the VSAM password immediately thereafter. Then, when you issue a CREATE TABLESPACE or CREATE INDEX statement, you must specify the VSAM password again.
- You may issue a CREATE TABLESPACE or CREATE INDEX statement before defining the data set. In that case, DB2 will define the data set automatically, and you can establish a VSAM password at that point. Using this method, you only need to specify the password once.

All data sets associated with the same table space or index must have the same password.

The password may be changed with ALTER TABLESPACE or ALTER INDEX, but is not actually changed until you use access method services to change the data set password in its ICF catalog entry.

ICF Catalog

You can also include a password when you create a storage group. This is the VSAM master password of the ICF catalog in which all dynamically created data base data sets associated with the storage group are cataloged.

MVS Passwords

You may establish MVS passwords for archive log data sets and DB2 libraries.

Archive Log Data Sets

You can specify a password for all archive log data sets when DB2 is started, using the initialization parameters. If specified, dynamically created archive log data sets are given the password protected attribute. Next, the archive log's data set name and password are put into the MVS password data set.

The password for an archive log data set is stored in the BSDS and is used on subsequent reference by DB2.

DB2 Libraries

You may specify MVS passwords for all DB2 target and distribution libraries immediately after they are created. To establish these MVS passwords, refer to *MVS/Extended Architecture Data Administration Guide*. In addition, you may refer to *TSO Extensions Command Language Reference* for information about using the TSO PROTECT command to create passwords.

Using the Catalog to Help Administer Authority

The DB2 catalog authorization tables contain information about all privileges granted to DB2 users. Each row in these tables contains the authorization ID that grants the privilege and the authorization ID to whom it is granted. Since you can select information from these tables, all authorization information concerning a data base is available to you.

For information about the catalog tables, see *IBM DATABASE 2 SQL Reference.*

Creating Views of the DB2 Catalog Tables

SQL cannot be used (1) to create, alter, or drop objects in the DB2 catalog, or (2) to insert, update, or delete rows of catalog tables. The only privilege applicable to catalog tables is SELECT. A user with SYSADM authority can grant the SELECT privilege on all catalog tables. If user JONES does not have SYSADM authority, JONES can select from a catalog table only if JONES or PUBLIC was granted the SELECT privilege on that table.

Most users should be allowed to see some parts of the catalog. For example, if JONES has any privileges on a table, JONES should be allowed to see the description of that table as recorded in catalog tables SYSIBM.SYSTABLES and SYSIBM.SYSCOLUMNS. However, if JONES is granted the SELECT privilege on

SYSIBM.SYSTABLES and SYSIBM.SYSCOLUMNS, JONES can see the description of all tables. Thus, the system administrator should consider the alternative of creating appropriate views of catalog tables and granting the SELECT privilege on the views, rather than the catalog tables.

Using the DB2 Catalog to Get Authorization Information

This section contains several examples showing how you can use the DB2 system catalog to gather security information about your DB2 subsystem. The examples assume you have the SYSADM level of authority. The examples demonstrate how to list:

- All DB2 users with any privilege
- All DBADM authorization IDs
- Users authorized to access a table
- The tables a user is authorized to access

Listing All DB2 Users with Granted Privileges

The DB2 authorization tables are part of the catalog data base. Each authorization table includes a column named GRANTEE and a column named GRANTEETYPE. If GRANTEETYPE is blank, the value of GRANTEE is an authorization ID that has been granted a privilege. There is no single DB2 authorization table that contains all authorization IDs. Therefore, if you want to list all authorization IDs with privileges, you can issue:

```
SELECT GRANTEE FROM SYSIBM.SYSCOLAUTH
  WHERE GRANTEETYPE = ' ' UNION
SELECT GRANTEE FROM SYSIBM.SYSPLANAUTH
  WHERE GRANTEETYPE = ' ' UNION
SELECT GRANTEE FROM SYSIBM.SYSUSERAUTH
  WHERE GRANTEETYPE = ' ' UNION
SELECT GRANTEE FROM SYSIBM.SYSDBAUTH
  WHERE GRANTEETYPE = ' ' UNION
SELECT GRANTEE FROM SYSIBM.SYSTABAUTH
  WHERE GRANTEETYPE = ' ' UNION
SELECT GRANTEE FROM SYSIBM.SYSRESAUTH
  WHERE GRANTEETYPE= ' ';
```

Periodically you should compare the list resulting from above SELECT statements with lists of users from subsystems which connect to DB2, such as:

IMS/VS
CICS and
TSO

If there are users with GRANTed privileges that do not exist in the other subsystems, you should REVOKE their authority.

Listing All DBADM Authorization IDs

To develop a list of all authorization IDs that have DBADM authority, you can issue:

```
SELECT DISTINCT GRANTEE
  FROM SYSIBM.SYSDBAUTH
  WHERE DBADMAUTH ¬=' ' AND GRANTEETYPE = ' ';
```

Listing Users Authorized to Access a Table

To list all users who are authorized to access the DSN8130.TEMPL table, you can issue:

```
SELECT DISTINCT GRANTEE
  FROM SYSIBM.SYSTABAUTH
  WHERE TTNAME = 'TEMPL' AND TCREATOR = 'DSN8130'
AND GRANTEETYPE = ' ';
```

To find out who can modify the DSN8130.TEMPL table, you can issue:

```
SELECT DISTINCT GRANTEE
  FROM SYSIBM.SYSTABAUTH
  WHERE TTNAME = 'TEMPL' AND TCREATOR = 'DSN8130' AND
GRANTEETYPE = ' ' AND
       (ALTERAUTH ¬= ' ' OR
        DELETEAUTH ¬= ' ' OR
        INSERTAUTH ¬= ' ' OR
        UPDATEAUTH ¬= ' ') ;
```

To list the columns of DSN8130.TEMPL for which users have been granted update privilege, issue:

```
SELECT DISTINCT COLNAME, GRANTEE
  FROM SYSIBM.SYSCOLAUTH
  WHERE CREATOR='DSN8130' AND TNAME='TEMPL'
AND GRANTEETYPE = ' '
  ORDER BY COLNAME;
```

To list the users of DSN8130.TEMPL that have the privilege of updating any column.

```
SELECT DISTINCT GRANTEE FROM SYSIBM.SYSTABAUTH
  WHERE TTNAME = 'TEMPL' AND TCREATOR='DSN8130' AND GRANTEETYPE=' '
  AND UPDATEAUTH ¬= ' ' AND UPDATECOLS = ' ' ;
```

Listing the Tables a User is Authorized to Access

When you want to know which tables and views a user (for example, PGMR001) is authorized to access, you can issue:

```
SELECT DISTINCT TCREATOR, TTNAME
  FROM SYSIBM.SYSTABAUTH
  WHERE GRANTEE = 'PGMR001' AND GRANTEETYPE =' ';
```

If you'd rather list the tables and views PGMR001 has created, you examine the SYSIBM.SYSTABLES table:

```
SELECT NAME FROM SYSIBM.SYSTABLES
  WHERE CREATOR = 'PGMR001';
```

Chapter 6. DB2 Attachment Facilities

DB2 provides attachment facilities for TSO and batch, IMS/VS, and CICS.

DB2 is connected to address spaces (TSO, batch, IMS/VS, or CICS) by *connection threads*. Threads are bidirectional paths between an application, command, or transaction processing program and DB2 resources. DB2 uses plans to determine what resources to allocate to the related threads.

The IMS/VS and CICS attachment facilities provide multiple-thread connections to IMS/VS and CICS address spaces. Most TSO and batch applications use the DSN command processor, which in turn uses the TSO terminal monitor program (TMP). These applications are limited to one thread for each address space. As an alternative, the call attachment facility provides multiple thread connections for TSO and batch applications.

Limiting Access to DB2

If many concurrent users are connected to DB2, applications may fail or perform badly because the demand for system resources exceeds the amount available. As an example, each thread requires space in the EDM pool for storage of an application plan and approximately 4K bytes of working storage.

To limit the number of concurrent connections to DB2 when you installed or migrated DB2, you specified three parameters on the Storage Sizes Panel (DSNTIPE):

- **MAX USERS (NUMCONCR)** limits the maximum number of threads allowed to access DB2, including TSO foreground users, batch jobs (in DSN command, or running a utility), utility jobs, IMS/VS regions, and CICS threads. This parameter also sets the CTHREAD parameter of macro DSN6SYSP.
- **MAX TSO CONNECT (NUMCONTS)** limits the number of concurrent users allowed to access DB2 from TSO, including those using DSN call authorization or in QMF, whether running a DB2 request or not. This parameter also sets the IDFORE parameter of macro DSN6SYSP. (This parameter does not limit the number of concurrent users allowed to access DB2 from CICS or IMS/VS; those subsystems may, however, set limits.)
- **MAX BATCH CONNECT (NUMCONBT)** limits the number of concurrent users allowed to access DB2 from batch, including those using DSN call authorization and utilities. This parameter also sets the IDBACK parameter of macro DSN6SYSP.

A connection may or may not be using a thread, depending on the activity in progress. A QMF user, for instance, uses a thread only when an SQL statement is being processed but remains identified to DB2 during the entire session.

If the maximum number of concurrent users is reached, additional attempts to connect to DB2 will be queued (first-in, first-out) or rejected, depending on the type of connection function performed (attempts to create a thread are queued; attempts to identify are rejected).

Connection Identification

DB2 associates a name with each address space connection. This 8-character name is called a *connection name*. The connection name identifies the connection of applications to DB2.

- For DSN users running in TSO foreground, the connection name "TSO" is used.
- For DSN users running in TSO batch, the connection name "BATCH" is used.
- For call attachment facility users, the connection name "DB2CALL" is used.
- For IMS/VS and CICS, the system identification name is used.

DB2 uses the connection name to distinguish the connection to which a message or recovery information applies. Use the -DISPLAY THREAD command to display the connection name associated with the thread.

When a system or subsystem failure occurs, DB2 can determine (for each connection name) all threads left in an unresolved (commit-indoubt) state. When connection is requested with a connection name that is in an unresolved state, DB2 will inform the connector of this status. A connector need not resolve all indoubt status before requesting access to DB2 resources. However, if resolutions are outstanding at the time of a resource access request, a system message is written identifying the connection name. Some DB2 messages identify the connection name to which the message applies.

TSO Attachment Facility

The TSO attachment facility allows users to access DB2 in the foreground through a TSO terminal, or in batch mode by invoking the TSO TMP (terminal monitor program) from an MVS batch job. Every TSO foreground user and batch job attaches separately to DB2.

This section discusses the DB2 and DB2I functions available under TSO. Information about connecting TSO to DB2 appears in *IBM DATABASE 2 Install Guide.*

DB2 Functions Available under TSO

You can access DB2 using the TSO command processor command, DSN. DSN also allows authorized users to create, modify, and maintain data bases and to control DB2.

DSN supports many subcommands, such as BIND, RUN, SPUFI, and DCLGEN. The DSN command processor will pass operator commands to DB2. The -START DB2 command cannot be issued from the TSO environment. If DB2 is not running, DSN cannot establish a connection to it; a connection is required for DSN to transfer commands to DB2 for processing.

If you want your application to invoke the TSO attachment facility from an MVS batch partition, you must include a JCL EXEC statement in the job to invoke the TSO TMP. Appropriate DD statements are required for TSO input and output. In this case, the DSN command would be in the data set you pass to TSO for input; the TSO responses would go into the data set you pass to TSO for output.

The name of the TSO terminal monitor program is "IKJEFT01"; the DDNAME that the TMP uses for its input is "SYSTSIN," and the DDNAME that it uses for its output is "SYSTSPRT."

DB2I Functions under TSO

Under TSO and DB2I, you can access the DSN command processor, the DSNH CLIST, and the DSNU CLIST.

TSO users can invoke DSN and use it to perform most DB2 functions, such as running applications and monitoring DB2 performance.

Figure 70 illustrates the basic structure of the TSO attachment facility.

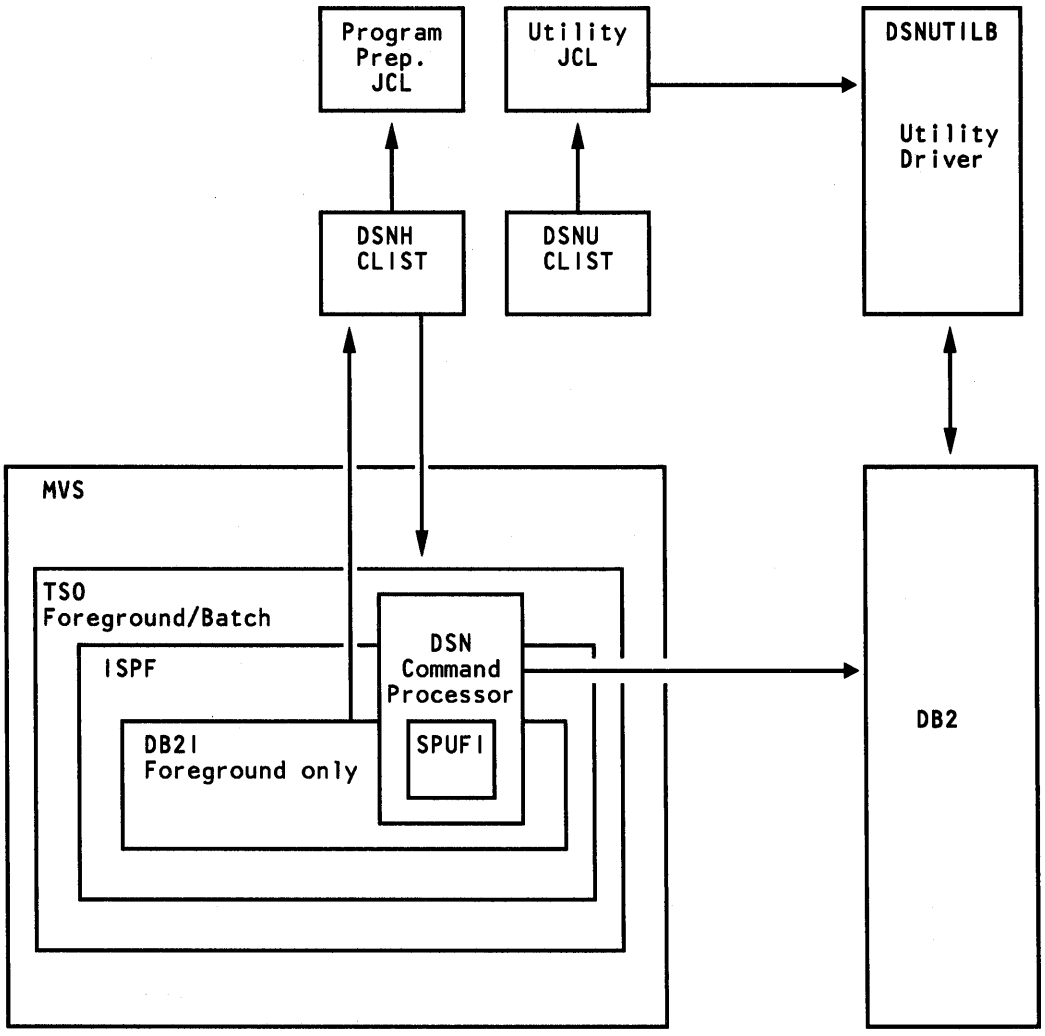


Figure 70. How TSO and DB2 are Connected

The following sections discuss how the TSO attachment facility uses the DSN command processor, the DSNH CLIST, and the DSNU CLIST.

DSN Command Processor

The DSN command processor handles subcommands in this way:

- DSN compares the submitted subcommand with its list of known subcommands, which include:

ABEND	BIND	DCLGEN
-DISPLAY	END	FREE
REBIND	-RECOVER	RUN
SPUFI	-START	-STOP
* comment statement	-TERM	

- If DSN recognizes the submitted subcommand, it processes it through to DB2.
- If DSN does not recognize the subcommand, it assumes it must be a TSO command and tries to attach it. If the attach is successful, the subcommand runs; if not, DSN treats it as a typographical error and displays the message `NOT A VALID COMMAND.`

DSNH CLIST (Program Preparation)

The DSNH CLIST allows you to precompile, compile, link-edit, bind, and execute an application by issuing a single command. CICS users can invoke the CICS command translator.

You can invoke the DSNH CLIST through DB2I using the DB2I Program Preparation panels. You can also invoke it directly from TSO or another CLIST. For a complete description of the DSNH CLIST, see *IBM DATABASE 2 Command and Utility Reference.*

DSNU CLIST (Utilities)

The DSNU CLIST generates the JCL needed to invoke the DSNUPROC procedure, which executes DB2 utilities as MVS jobs. The DSNU CLIST generates JCL for these utilities:

CHECK	COPY	LOAD
MERGECOPY	MODIFY	RECOVER
REORG	REPAIR	RUNSTATS
STOSPACE		

For a more information on the DSNU CLIST, see *IBM DATABASE 2 Command and Utility Reference.*

Call Attachment Facility

Most TSO applications should use the TSO attachment facility described above, which invokes the DSN command processor, which in turn requires the TSO Terminal Monitor Program (TMP). Using the TSO TMP, it is easy to run DSN applications interactively or as batch jobs. DSN provides services such as automatic connection to DB2, attention key support, and translation of return codes into error messages. However, when using DSN services, your application must run under the control of DSN. You must depend on DSN to manage your connection to DB2; you cannot change plan names without allowing your application to terminate. For some applications, these limitations present a problem.

The call attachment facility provides an alternate choice for TSO and batch applications needing tight control over the session environment. Applications using the call attachment facility can explicitly control the state of their connections to DB2 by using connection functions supplied by the call attachment facility:

CONNECT
DISCONNECT
OPEN
CLOSE
TRANSLATE

For more information on the call attachment facility, see *IBM DATABASE 2 Advanced Application Programming Guide.*

IMS/VS Attachment Facility

This section describes the IMS/VS attachment facility. It contains information about:

- Defining a connection between DB2 and IMS/VS
- The relationship between IMS/VS applications and DB2
- The IMS/VS attachment facility options

Defining a Connection

A connection to DB2 is established from the IMS/VS control region and every dependent region that accesses DB2 resources. DB2 resources are identified by an application plan. Each plan is given a plan ID when it is created.

The responsibility for making the initial contact with DB2 resides in the IMS/VS control region. DB2 must be defined to IMS/VS as an external subsystem. As dependent regions are started, they are connected to DB2 if required.

Figure 71 shows the connection between DB2 and IMS/VS.

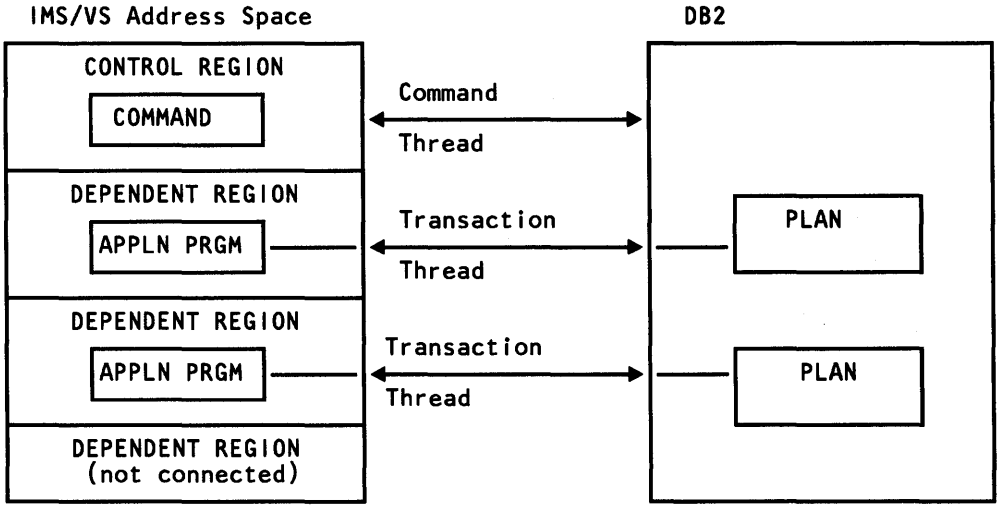


Figure 71. IMS/VS Connected to DB2

You can use RACF to validate the connection between IMS/VS and DB2. When the connection is established, the IMS/VS subsystem identification (IMSID) becomes the

connection name. DB2 checks that a dependent region belongs to an authorized control region.

Whenever you enter a DB2 command, the control region establishes a thread to DB2 (unless a thread remains available from the processing of a previous command).

In the case of a dependent region, a thread allows an IMS/VS application to use a DB2 plan. Each time a dependent region schedules a different IMS/VS transaction, a new thread may be established. The connection to DB2 from IMS/VS regions is based on user-specified values in a IMSVS.PROCLIB member called the subsystem member (SSM). This permits application requests to travel directly to a specific DB2 subsystem, thereby reducing path length by avoiding control region overhead. See Figure 72 for an illustration.

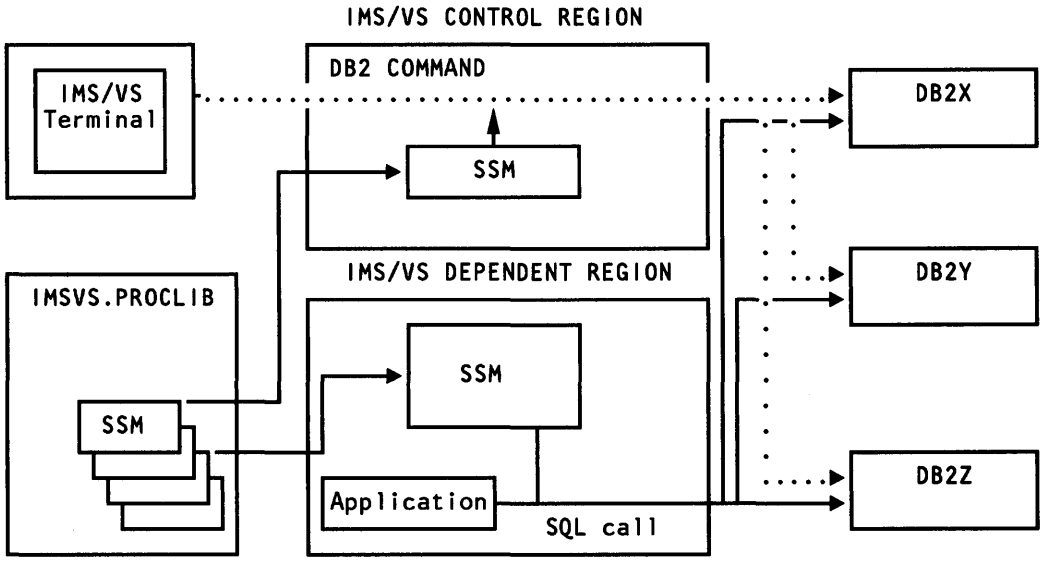


Figure 72. SSM and DB2 Connect

Each SSM member in IMSVS.PROCLIB can have multiple entries defining connections to multiple DB2s.

You can also group transactions into classes and assign different classes to different regions. This permits you to set up specific regions for different application types. By doing so, you can specify some regions to be devoted to process transactions that access only DL/I resources (see MPP #2 in Figure 73 on page 156). Other regions can be reserved for applications that access only DB2 resources; other regions can be allowed to access both DL/I and DB2 data (see MPP #1 in Figure 73 on page 156). See *IBM DATABASE 2 Application Programming Guide* for details about the way an external subsystem is selected.

After control and dependent regions have been initialized and contact has been made with the DB2 subsystem, applications can access DB2 data. Two important factors affect DB2 performance:

- The creation and termination of threads that allow access to DB2 data
- The way in which application units of work are committed or aborted

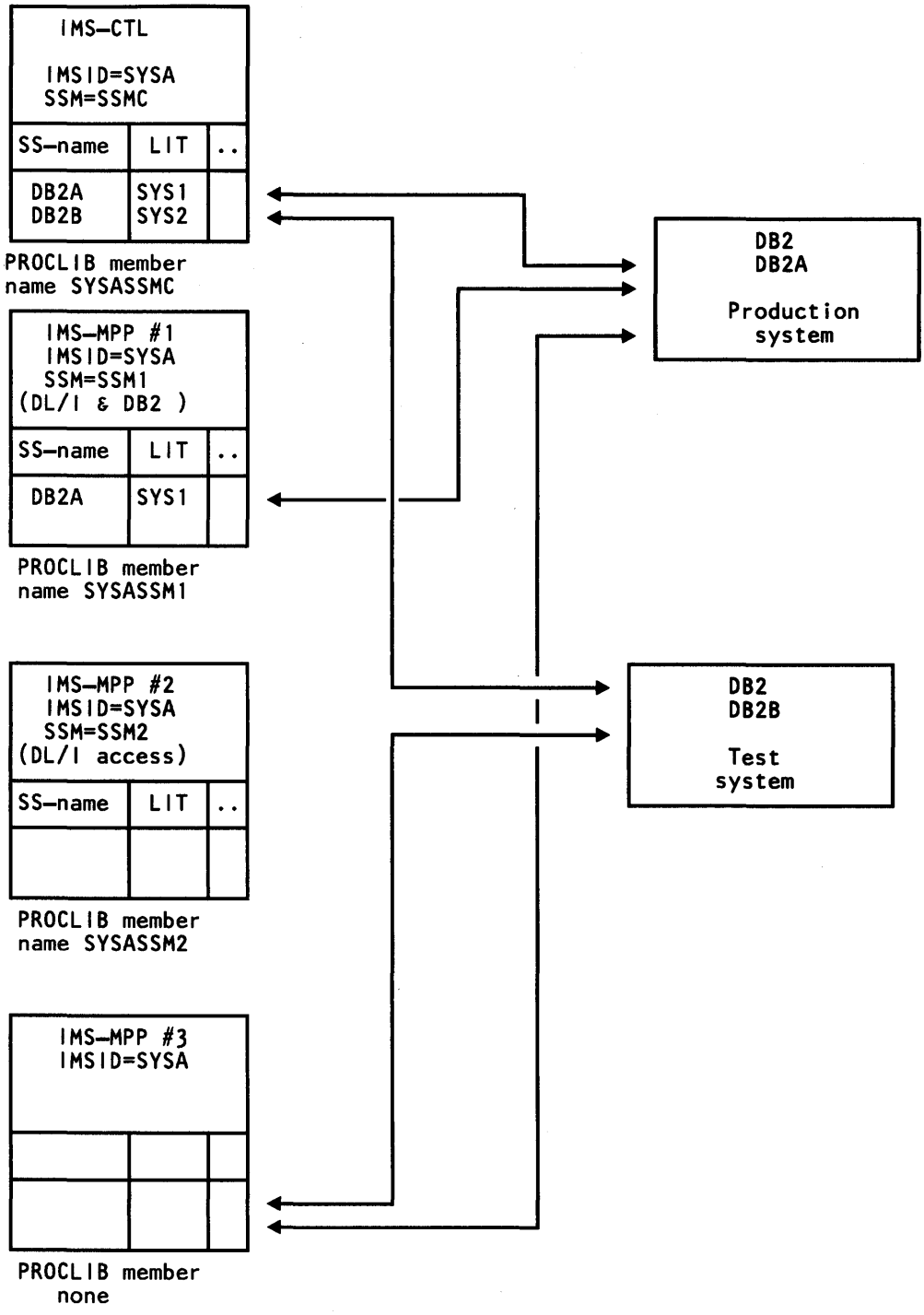


Figure 73. How IMS/VS and DB2 are Connected

Relationship between IMS/VS Applications and DB2

Figure 74 is an example of the flow of an application as it relates to IMS/VS and DB2.

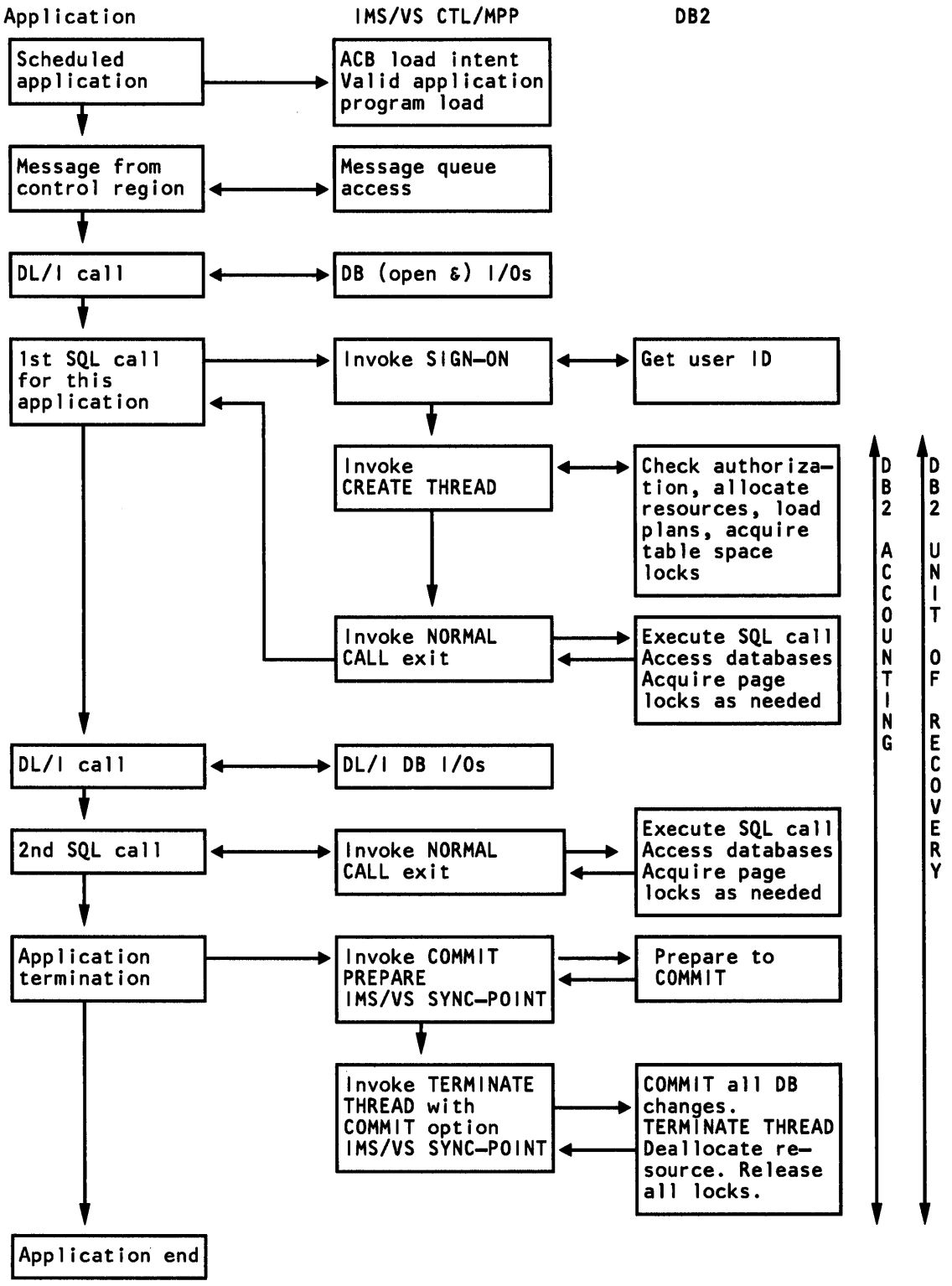


Figure 74. Transaction Flow and Events for a MODE=SNGL Transaction

The first column represents an application transaction from scheduling to termination. The center column represents the IMS/VS response to the path of the transaction. The last column represents the actions DB2 takes in response to this transaction.

A thread, as defined in the beginning of this chapter, is created when an application issues its first SQL call. The thread usually is terminated when an application terminates or an abend occurs. If the thread is not terminated, the same thread will handle all of the SQL calls issued during each application schedule. However, when a new transaction is processed, user ID authorization checking will occur again. DB2 will do validity checking during the processing of SQL statements for those statements that were not checked during the bind or rebind process and for all dynamic SQL statements.

IMS/VS uses the two-phase commit process when committing data base changes across subsystems. The commit process is triggered by these application sync points:

- I/O PCB GU call for an application defined as MODE=SNGL
- Application CHKP or SYNC call
- Application termination

Each unit of work is associated with one specific thread and is made up of one or more units of recovery. A unit of recovery is the work done by an application between two consecutive points of consistency (that is, commit or abort points).

You can specify the DB2 plan ID to be used for an IMS/VS application. If you do not, the DB2 plan ID name defaults to the IMS/VS application load module name.

The attachment facility modules provide most of the connection functions. These modules are a DB2 component, but they are shipped on their own distribution tape. This tape also contains the IMS/VS attachment macros. After installing the tape, the IMS/VS attachment modules are in DSN130.DSNLOAD and the macros are in DSN130.DSNMACS.

IMS/VS Attachment Facility Options

The IMS/VS attachment facility options are described in the *IBM DATABASE 2 Install Guide.*

Threads: No parameter controls the maximum number of DB2 connections that the IMS/VS attachment facility will use. The number is limited by the number of active dependent regions enabled to access DB2 resources as specified in SSMs.

Considerations:

- Consider separating SQL applications from DL/I into different IMS/VS dependent regions. This makes it easier to monitor performance, isolate problems, and select scheduling algorithms and class assignments.
- Avoid defining more external subsystems than needed in the SSM. Each definition will add storage and processor requirements.
- Assess the time and resources involved in creating and terminating threads. The process is closely related to IMS/VS scheduling. Consider the following:
  - When appropriate, use IMS/VS options and parameters, such as WFI (wait for input) and PROCLIM, so the system can process more than one transaction per schedule.

However, where concurrent access to a table space is desired, you must be aware of the locks acquired. This is especially true when assigning the WFI attribute to transaction types and table space locks that are not released until application termination. See *IBM DATABASE 2 Data Base Planning and Administration Guide* for a more detailed discussion of locking.

  - Require applications to issue a GU call against the I/O PCB before termination if further input messages can be processed.

- Keep the use of ROLL/ROLB to a minimum.

DB2 can be used in an Extended Recovery Facility (XRF) environment to facilitate recovery from an IMS/VS failure. To accomplish this, you must place all DB2 data sets on DASD shared between the primary and alternate XRF processors. This will enable DB2 to be stopped manually on the primary processor and started on the alternate. To preserve data integrity in case of an operator error, global resource serialization must be active, and the primary and alternate XRF processors must be included in the global resource serialization ring. For more information about XRF, see *IBM DATABASE 2 Operation and Recovery Guide* and *IMS/VS Version 2 Operations and Recovery Guide.*

CICS Attachment Facility

The CICS attachment facility allows CICS command-level applications issuing SQL statements to access DB2 resources. This section contains information about:

- The connection between CICS and DB2
- The thread process
- CICS attachment facility options

The connection between CICS and DB2 is defined through the resource control table (RCT). The CICS user requests the connection of CICS to DB2 through a CICS attachment facility STARTUP TRANSACTION provided with DB2. The task initializes the CICS attachment by:

- Loading the site-defined resource control table (RCT)
- Attaching supporting CICS transaction and MVS tasks
- Establishing the subsystem connection between CICS and DB2

After the attachment facility environment and subsystem connections have been established, CICS transactions can access DB2 resources.

When a CICS transaction makes a DB2 SQL request, the responsibility for servicing the request is assigned to an attachment-managed thread subtask. The relationship between the CICS transaction and its assigned thread subtask is maintained for the duration of the logical unit of work. The thread subtask may then be assigned the responsibility of servicing another CICS transaction.

This mechanism allows DB2 resources allocated to a given thread subtask to be used serially by different CICS transactions, reducing deallocation and reallocation overhead.

Figure 75 on page 160 shows how CICS connects to DB2.

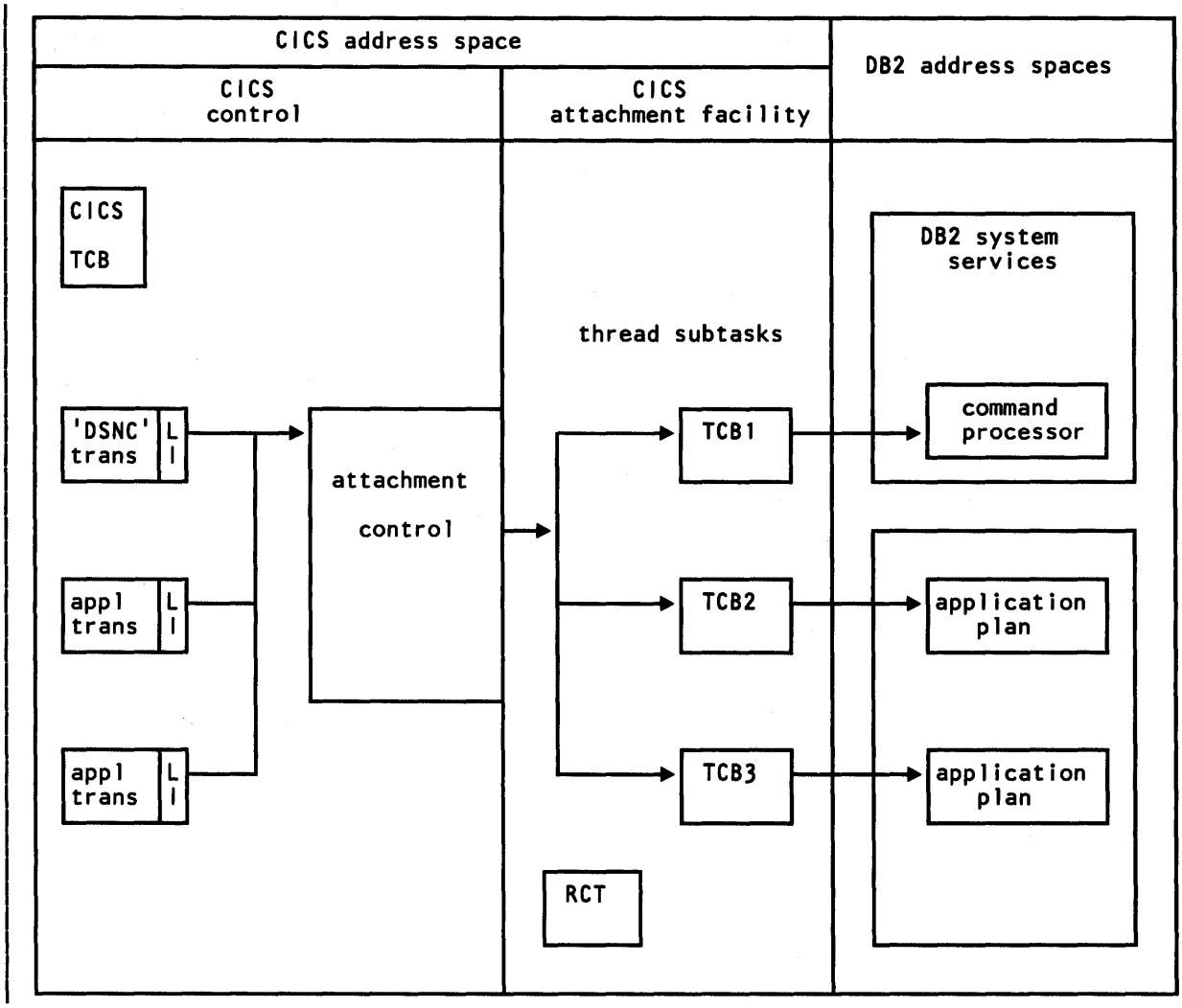


Figure 75. How CICS and DB2 Are Connected

Thread Process

When a transaction issues its first SQL call, a CICS attachment facility thread-element is assigned to it as specified in the Resource Control Table (RCT). A thread-element consists of the control and resources needed to communicate to DB2. Normally, a thread-element has associated with it an MVS subtask, which is managed by the CICS attachment facility. This MVS subtask will perform the actual SQL request.

An RCT entry defines one or more thread-elements. A thread-element can be in one of the following states:

- Unavailable for reassignment (in use with resources allocated);
- Available for reassignment with resources allocated to it;
- Available for reassignment without any resources allocated; or
- Idle (no subtask assigned to it).

If an RCT entry associated with the transaction ID is found and the entry allows for multiple thread-elements, a search for an available thread-element proceeds in the following order:

1. An available element allocated to the required plan;
2. An available element allocated to a different plan; then
3. An idle element.

If a thread-element is found, it is associated with the transaction. If not, the request will create a new thread-element if the maximum (THRDA) has not been reached. If the maximum has been reached, the request will be made to wait, diverted to the POOL, or abended, depending on the RCT entry specification (TWAIT).

In the POOL, thread-elements are maintained that can be assigned to any transaction which has been diverted or defaulted to the POOL entry. For terminal-driven transactions within the POOL, thread-elements are deallocated at the end of each logical unit of work (SYNCPOINT). For nonterminal-driven transactions, the thread-element is deallocated only at the physical end of the task.

The search for an available thread-element in the POOL begins with an available element allocated to the required plan. If an available element is not found, an idle element is used.

Figure 76 shows the different actions the CICS attachment facility will take, based on the status of the thread-element associated with the transaction.

THREAD STATUS		CICS ATTACHMENT FACILITY ACTIONS
Available	Yes Same Plan	1. Signon (if USER ID or TRAN ID changes) 2. Process the call
	Yes Different Plan	1. Terminate thread 2. Signon 3. Create thread 4. Process the call
	No	1. Signon 2. Create thread 3. Process the call
Idle		1. Acquire a subtask 2. Identify to DB2 3. Signon 4. Create thread 5. Process the call

Figure 76. CICS Attachment Facility Actions and Thread Status

If no RCT entry associated with the transaction ID is found, the POOL entry and its thread-elements are used to process the request.

A thread-element may process one or more CICS transactions before being terminated. When a new transaction is processed, signon is invoked if the TRAN ID or USER ID changes. If signon is invoked, DB2 authorization checking will again take place.

When a synchronization point or a transaction termination occurs, CICS invokes the two-phase commit process. At this point, the thread-element assigned to the transaction is examined. If the thread-element is from the POOL, the thread-element may

be reassigned only if another transaction is waiting for the plan in use. Otherwise, the thread-element is terminated.

If the thread-element is from a dedicated entry (TYPE=ENTRY), the CICS attachment facility will assign the thread-element to a transaction that is queued for this entry. If no transactions are queued, the thread-element is examined to see if it is a protected thread-element. This is done by checking the THRDS parameter in the RCT entry specified for this thread. If it is not protected, the thread-element is terminated.

In addition, all entries are checked every 30 seconds for activity. Thread-elements that have no activity for two consecutive 30-second periods are terminated.

Figure 77 on page 163 shows the major events that are typical when a transaction is active. The time period may be divided into one or more units of work or units of recovery, each representing the work done between two commit points. A commit point is determined explicitly by the application program issuing a sync-point command, or implicitly by the application's termination.

When a thread-element is terminated, the associated MVS subtask is not necessarily detached. Subtasks are detached only when the number of active TCBs (sum of all THRDAs) is within two of THRDMAX.

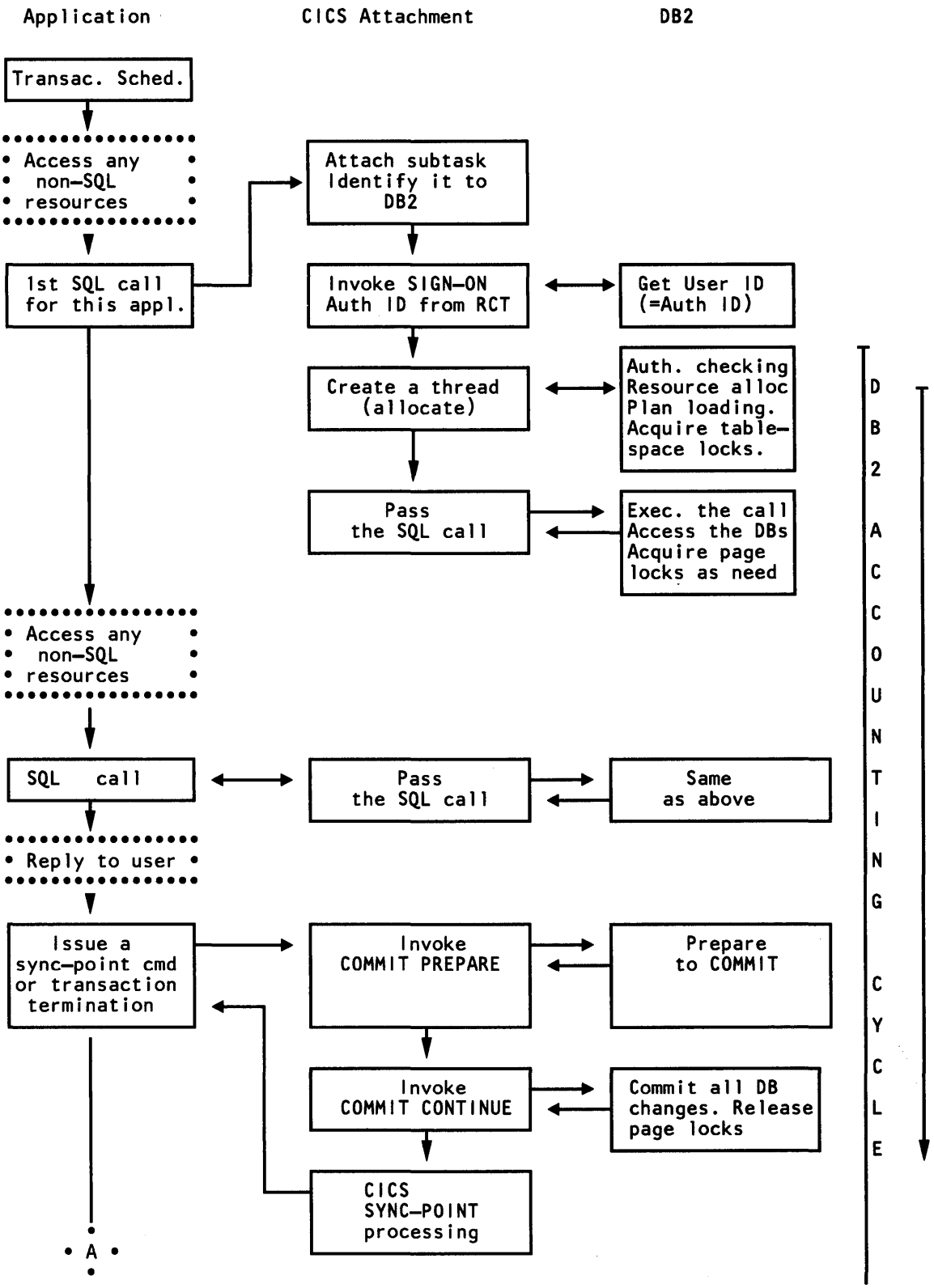


Figure 77 (Part 1 of 2). CICS Transaction Flow and Events

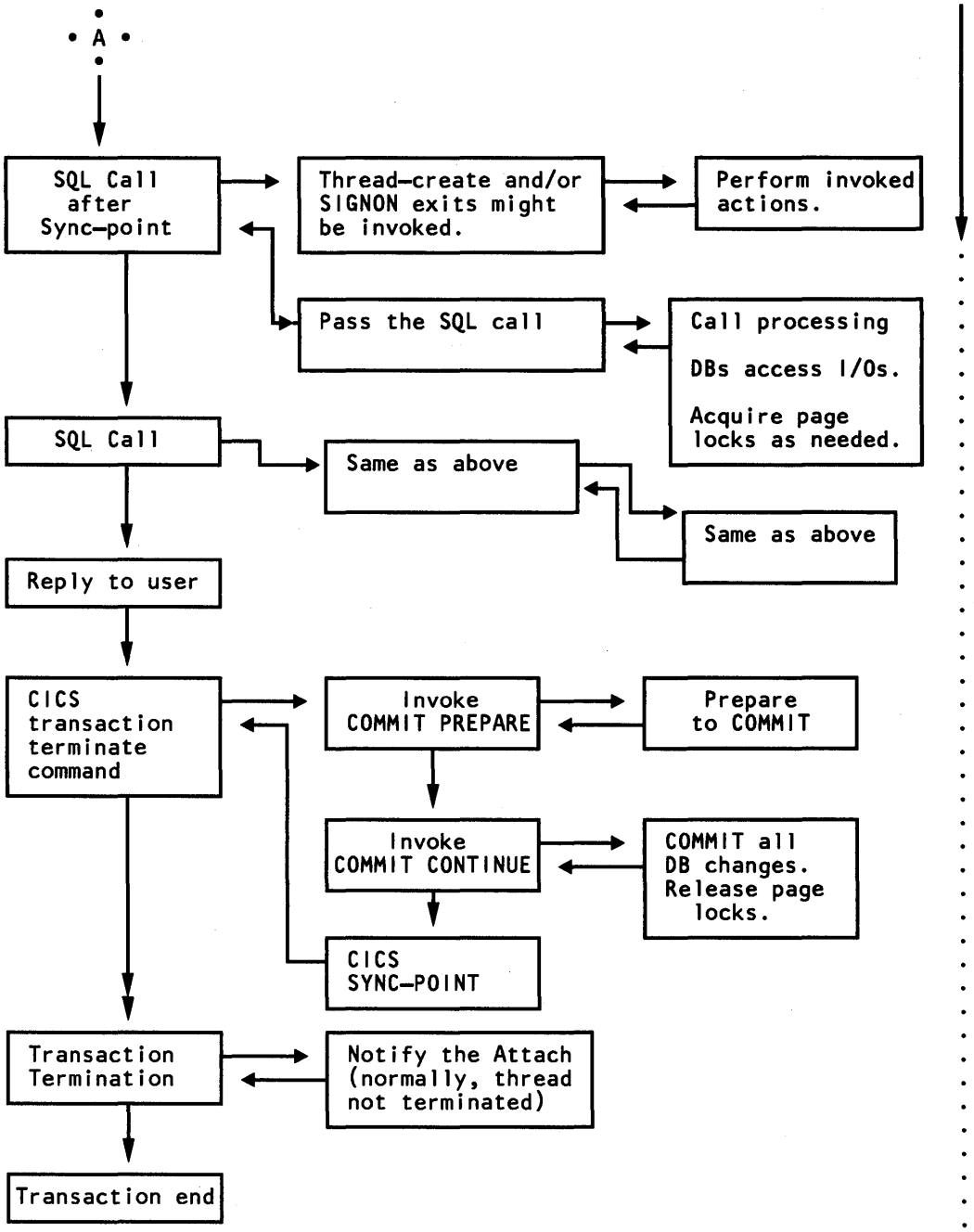


Figure 77 (Part 2 of 2). CICS Transaction Flow and Events

CICS Attachment Facility Options

Three types of options relate to performance:

- Thread-specific parameters, such as dispatching priorities
- Maximum thread-element number and how many threads can be allocated to the different transactions
- Where thread-elements are allocated (POOL or ENTRY)

Figure 78 shows the most important parameters used when defining RCT entries and how they might relate to some CICS facilities. These facilities are defined via parameters in the CICS System Initialization Table (SIT).

Parameters		Meaning - usage hints	SIT correlation
INIT	ENTRY/POOL		
THRDMAX	-	Maximum number of threads that can be activated. If reached and demand for more occurs, the attachment will try to terminate all threads and associated subtasks not currently active with a transaction (available or reusable status). Should be less than or equal to MAX USERS, specified at install on Storage Sizes Panel (DSNTIPE).	MXT/ AMXT
-	THRDA	Maximum number of threads for this ENTRY. If exceeded, action specified by TWAIT will apply to the request.	CMXT
-	THRDS	Number of threads to be started when attachment is started if TYPE=POOL or TYPE=ENTRY. If TYPE=ENTRY, THRDS is also the number of protected threads.	-
TWAITI	TWAIT	Specify action to be taken against a transaction that requires a thread, if all are busy. Valid specifications are YES, NO, POOL: • YES: transaction is made to wait until a thread-element becomes available. • NO: transaction is aborted. • POOL: transaction is diverted to use the POOL.	-
DPMODI	DPMODE	Specify HIGH, EQ, or LOW for dispatching priority relative to CICS. HIGH may be useful for: • High priority inquiry transactions • Short duration update transactions	ICV

Figure 78. CICS Attachment Facility Options

These RCT parameters can help you control thread reuse, serial transaction execution, workload, and unexpected thread termination.

Thread reuse can eliminate overhead and improve speed, two important considerations for high-volume, performance-oriented transactions. However, thread reuse should be specified only for transactions that will not create concurrency problems by contending for DB2 table space locks, CICS, MVS, VSAM, or other resources. To increase the probability of thread reuse, the THRDS parameter and TWAIT=YES should be specified when possible for TYPE=ENTRY thread elements. In addition, give THRDA and THRDS values large enough to handle the normal arrival rate of transactions with minimal queueing.

Controlling the maximum number of CICS transactions concurrently executing a plan can help you control workload. To do this, THRDA must be greater than 0 and TWAIT must be YES.

For transactions that create resource contentions, serial execution can help improve performance. Specifying THRDA=1 and TWAIT=YES will achieve serialization.

To help prevent early or unexpected thread termination, coordinate THRDMAX with individual THRDAs. Any thread will be terminated when the number of CICS attachment subtasks is within two of THRDMAX.

The RCT parameters listed in Figure 78 apply to both POOL and ENTRY thread-elements; however, you may decide whether to have only one or both of the above thread types. Advantages and disadvantages include:

- POOL Advantages:
  - Thread-elements can be shared among more transactions.
  - The same transaction rate might be handled by a smaller number of thread-elements (if the transaction rate is low).
  - Less virtual storage is needed.
  - The number of thread-elements is easier to optimize when all of the transactions go to the POOL.
- POOL Disadvantages:
  - Thread-elements are always terminated at transaction termination or at commit.
    - More processor time is needed for each single thread.
    - More catalog I/Os will be required.
  - There is no way to isolate heavy or long-running transactions (unless CICS facilities are used).
- ENTRY Advantages:
  - Transactions can be isolated into groups.
  - Thread-elements can be reserved for specific transactions.
  - Each thread-element can handle several units of work and even several transaction occurrences before being terminated.
    - Less processor time will be used for each single thread.
    - Fewer catalog I/Os will be required.
  - THRDS can be used to designate a certain number of thread-elements to retain their resources for reassignment for the higher transaction rates. These thread-elements are terminated by task purge, which occurs every 30 seconds. Only thread-elements that have not been used for the task purge interval are terminated.
- ENTRY Disadvantages:
  - More virtual storage is needed.
  - The same transaction rate will probably require a larger number of thread-elements (if the transaction rate is low).

Chapter 7. Monitoring DB2

This chapter describes the various facilities for monitoring DB2 activity and performance. It includes information on

Using the RUNSTATS Utility

The RUNSTATS utility scans table spaces or indexes to gather data about space utilization and index efficiency. It uses that data to update statistical information in DB2 catalog tables.

This utility has three uses:

1. *By updating the DB2 catalog, RUNSTATS provides DB2 with current information from which to perform path selection.* To ensure that information in the catalog is current, you should invoke RUNSTATS:
   - After loading a table space and before binding application plans that will access the table space.
   - After reorganizing a table space or an index. Then, you should rebind plans where performance remains a concern.
   - Periodically. By comparing the output of one execution with previous executions, you can detect a performance problem early.
   - Against the DB2 catalog. Original statistics on the DB2 catalog tables represent a typical catalog; in Release 3 you can run the RUNSTATS utility against your DB2 catalog to provide the SQL optimizer with more accurate information for access path selection.
2. *RUNSTATS helps you determine performance trends.* When used routinely, RUNSTATS provides data about table spaces and indexes over a period of time. You can use the utility as a performance tracking tool to help you decide when to tune and reorganize your data base.
3. *RUNSTATS helps you assess the effect your changes* have on performance. After you've determined the data base space utilization characteristics you consider normal for your site, you can develop a profile against which to compare the effects of your changes. For instance, your objective may be to reduce the number of I/Os required to perform a particular operation. The RUNSTATS utility can help you determine whether you've met your objective.

Figure 79 on page 168 shows the statistical information RUNSTATS collects.

Catalog Table	Column Name	Column Description
SYSTABLES	CARD	Total number of rows in the table
	NPAGES	Total number of pages on which rows of this table appear
	PCTPAGES	Percentage of total pages of the table space which contain rows of the table
SYSCOLUMNS	HIGH2KEY	The second highest value of the column if the column is the first column of an index key
	LOW2KEY	The second lowest value of the column if the column is the first column of an index key
	COLCARD	Number of distinct values for the column if the column is the first column of an index key
SYSTABLESPACE	NACTIVE	Number of active pages in the table space; shows the number of pages that would be touched if a record cursor were used to scan the entire file
SYSINDEXES	CLUSTERED	Indicates whether the table is actually clustered (determined by RUNSTATS)
	FIRSTKEYCARD	Number of distinct values of the first key column
	FULLKEY	Number of distinct values of the key
	NLEAF	Number of active leaf pages in the index
	NLEVELS	Number of levels in the index tree
SYSTABLEPART	CARD	Total number of rows in the table space or partition
	NEARINDREF	Number of rows relocated near their original page
	FARINDREF	Number of rows relocated far from their original page
	PERCACTIVE	Percentage of space occupied by active rows, containing actual data from active tables
	PERCDROP	Percentage of space occupied by rows of data from dropped tables
SYSINDEXPART	CARD	Number rows referenced by the index or partition
	NEAROFFPOS	Number of referenced rows near, but not at optimal position, because of an insert into a full page
	FAROFFPOS	Number of referenced rows far from optimal position because of an insert into a full page (value is lowest just after a table space reorganization)
	LEAFDIST	100 times the number of pages between successive leaf pages of the index

Figure 79. DB2 Catalog Data Collected by RUNSTATS

You can use this information to determine whether a table space or an index should be reorganized to improve performance or storage utilization. In SYSTABLEPART, the columns that can help you decide when to reorganize include PERCDROP, NEARINDREF, and FARINDREF. In SYSINDEXPART, the columns that can help you decide when to reorganize include NEAROFFPOS, FAROFFPOS, and LEAFDIST. For examples of queries on these columns, see "Monitoring the Use of Table Spaces" and "Monitoring the Use of Indexes" on page 170.

For information on the path selection function of BIND, see "Using the Bind Process" on page 198. For information about how RUNSTATS affects data base performance, see *IBM DATABASE 2 Data Base Planning and Administration Guide.*

Using the STOSPACE Utility

The STOSPACE utility gathers data about the actual space allocated for storage groups, table spaces, and indexes. It uses that data to update columns in the DB2 catalog. Used periodically, the STOSPACE utility can help you determine if the defined DASD space should be increased or decreased. Figure 80 provides statistical information recorded by the STOSPACE utility that is useful for space allocation decisions.

Catalog Table	Column Name	Column Description
SYSTABLESPACE	SPACE	Number of kilobytes of storage allocated to the table space
SYSINDEXES	SPACE	Number of kilobytes of storage allocated to the index
SYSSTOGROUP	SPACE	Number of kilobytes of storage allocated to the storage group
	SPCDATE	Date when the SPACE column was last updated

Figure 80. DB2 Catalog Data Collected by STOSPACE

When storage groups are used in the creation of table spaces and indexes, DB2 defines the data sets for them. The STOSPACE utility permits a site to monitor the DASD space actually used within the storage group. If the table space or index space is partitioned, and different storage groups have been specified, then the SPACE column (of SYSTABLESPACE or SYSINDEXES) gives the number of kilobytes of storage allocated to a partition. The partition is determined by the last execution of the STOSPACE utility.

For information about how STOSPACE affects data base performance, see *IBM DATABASE 2 Data Base Planning and Administration Guide.*

Using the DB2 Catalog

Information in the DB2 catalog can help you determine when to reorganize table spaces and indexes. You cannot, however, reorganize catalog table spaces.

Monitoring the Use of Table Spaces

Information from the SYSTABLEPART catalog table can tell you how well DASD space is being used.

If you want to find out the number of varying-length rows relocated to other pages because of an update, run RUNSTATS and issue this statement:

```
SELECT CARD, NEARINDREF, FARINDREF
  FROM SYSIBM.SYSTABLEPART
  WHERE DBNAME = 'XXX'
  AND TSNAME = 'YYY';
```

A large number in the FARINDREF indicates high I/O activity on the table space. If that number increases over time, you will probably want to reorganize the table space to improve performance.

Issue the following statement to determine the percentage of "dead" space in table space YYY:

```
SELECT PERCDROP
  FROM SYSIBM.SYSTABLEPART
  WHERE DBNAME = 'XXX'
  AND TSNAME = 'YYY';
```

DB2 reclaims space lost through DROP activity when you reorganize the table space.

Issue the following statement to determine whether the rows of a table are stored in the same order as the entries of its cluster index (XXX.YYY).

```
SELECT NEAROFFPOS, FAROFFPOS
  FROM SYSIBM.SYSINDEXPART
  WHERE IXCREATOR = 'XXX'
  AND IXNAME = 'YYY';
```

A large number for FAROFFPOS indicates clustering is degenerating. Reorganizing the table space would improve performance. However, REORG will not change the positioning of records in a table to reflect a clustering index defined on that table if the table space contains multiple tables.

Monitoring the Use of Indexes

The index statistics give you information about the level of indexes in your data base and the number of *leaf pages* in the index. When DB2 first builds your index, it creates a *root page* 4096 bytes long, as shown in Figure 81 on page 171. This root page points directly to the data in your tables giving the key and the row ID. Index pages that point directly to the data in your tables are called *leaf pages*. If you have only one index page, the root page is also known as the leaf page.

If your index grows beyond the 4096-byte page, a page split occurs and a second-level index page is created. In this case, the index entries in the root page point to the entries in the second-level index page, as shown under LEVEL 1 in Figure 81 on page 171. The root page will then contain the page number and highest key of each page in the second-level index. The root page is no longer called a leaf page; the second-level pages pointing directly to the data are now known as leaf pages.

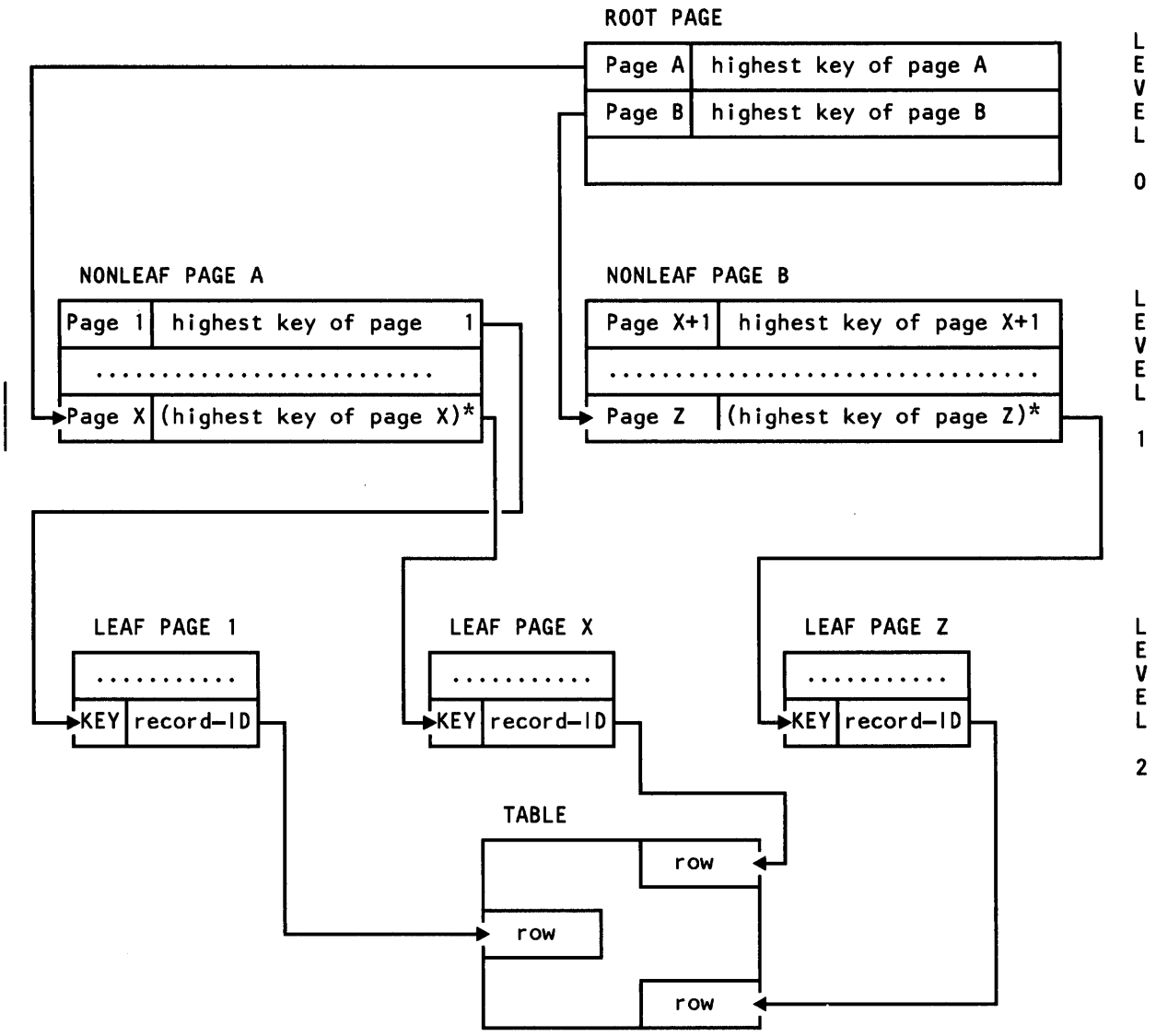
The number of pages in the second-level index will depend on the number of index entries you have. Each leaf page holds 4096 bytes of data, and, if one page is full, an insertion in the index would cause a leaf page split. A leaf page split would cause another page to be added to that level of index as shown under LEVEL 2 in Figure 81 on page 171. If the root page overflows a second time, another level of index would be added.

Remember that each time the root page is split, another level of index is created. You have only one root page per index or partition (if the index is partitioned).

The following statement determines the average distance between successive leaf pages during sequential access of the ZZZ index.

```
SELECT LEAFDIST
  FROM SYSIBM.SYSINDEXPART
  WHERE IXCREATOR = 'AAA'
  AND IXNAME = 'ZZZ';
```

A number that increases over time probably indicates the index should be reorganized.



* NOTE: The highest key of a nonleaf page is not actually stored in the page. It is implicit. The next higher level page gives the value of the highest key of the nonleaf page.

Figure 81. Sample Index Structure (three-level index)

Using the DISPLAY Command

The DB2 -DISPLAY command gives you information about the status of threads, data bases, allied subsystems, and applications. Two forms of the -DISPLAY command are particularly helpful for monitoring DB2: -DISPLAY THREAD and -DISPLAY DATABASE.

The -DISPLAY THREAD command displays

- Active or indoubt threads within the DB2 subsystem
- Active or indoubt threads associated with specific connection names

The -DISPLAY DATABASE command displays

- Status of a data base (for example, active or active/restricted)
- Names of table spaces or index spaces within a data base
- Connection names of applications holding or awaiting locks on a data base
- Connection names of applications currently using a data base
- Information about data bases, table spaces, or index spaces whose use is restricted

For information about the syntax of these commands, see *IBM DATABASE 2 Command and Utility Reference.*

Using MVS, CICS, and IMS/VS Tools

To monitor DB2 and CICS, you can use:

- RMF Monitor II for physical resource utilizations
- GTF for detailed I/O monitoring when needed
- CICSPARS for response time analysis

RMF Monitor II allows you to dynamically monitor system-wide physical resource utilizations. This can show queuing delays in the I/O subsystem.

In addition, the CICS attachment facility DSNC DISPLAY command allows any authorized CICS user to dynamically display statistical information related to thread usage and situations when all threads are busy. For more information about the DSNC DISPLAY command, see *IBM DATABASE 2 Command and Utility Reference.*

If the number of threads reserved for specific transactions or for the POOL is not large enough to handle the actual load and is causing performance problems, you can dynamically modify the value specified in the Resource Control Table (RCT) with the DSNC MODIFY TRANSACTION command.

To monitor DB2 and IMS/VS, you can use:

- RMF Monitor II for physical resource utilizations
- GTF for detailed I/O monitoring when needed
- IMSPARS or its equivalent for response time analysis
- The IMS/VS DC Monitor or its equivalent for tracking all IMS/VS-generated requests to DB2

In addition, the DB2 IMS/VS attachment facility allows you to use the -DISPLAY THREAD command to dynamically observe DB2 performance.

Using DB2 Trace

DB2 provides a trace facility that allows you to record subsystem data and events. Analysis and reporting of the trace records, however, must take place outside of DB2. You can use another licensed product, IBM DATABASE 2 Performance Monitor (DB2PM), to format, print, and interpret DB2 trace output. For more information on DB2PM, see *IBM DATABASE 2 Performance Monitor General Information* and *IBM DATABASE 2 Performance Monitor User's Guide.*

DB2 trace can record four types of data:

- **Statistics** data for DB2 capacity planning and subsystem tuning.
- **Accounting** data for assigning DB2 costs to individual authorization IDs and for tuning individual programs.
- **Performance** data, for monitoring various DB2 events. You can use this data to perform program, resource, user, and subsystem tuning.
- **Global** data, which describes entries to and exits from functions and modules. This data is intended primarily for servicing DB2.

This chapter describes three of these trace types: statistics, accounting, and performance. Global data, used primarily for servicing DB2, is explained in *IBM DATABASE 2 Diagnosis Reference Volume 4: Service Traces*.

Controlling DB2 Trace

Three commands allow you to control DB2 trace:

-START TRACE invokes traces. You may also have invoked traces automatically when you installed or migrated.

-DISPLAY TRACE displays the active trace options.

-STOP TRACE stops tracing activity.

To use these commands, you must have SYSADM or SYSOPR authority or have been granted trace authority (to issue -START TRACE or -STOP TRACE commands), or have been granted display authority (to issue -DISPLAY TRACE commands).

Starting DB2 Trace

The -START TRACE command invokes any of the four DB2 trace types: statistics, accounting, performance, and global.

You can specify several other parameters, each of which qualifies the scope of the trace in some way. This section discusses the various keywords and parameters you can specify when invoking a trace. These keywords are:

- TYPE—type of trace being invoked
- CLASS—class or classes of events the trace will monitor
- DEST—destination or destinations to which trace data is sent
- PLAN—specific DB2 plan or plans the trace will monitor
- AUTHID—specific authorization ID or IDs the trace will monitor
- RMID—specific resource manager or managers the trace will monitor

Not all keywords can be specified for all trace types. For instance, you cannot specify PLAN for a statistics trace. In addition, the parameters you can specify for particular keywords vary depending on the trace type. For example, you cannot specify a destination of RES (resident trace table) for a performance trace. All restrictions are identified in the following sections.

Specifying a Trace Type

You must specify a trace type when invoking DB2 trace. It is a required parameter; the default is null. You can specify only one trace type for each invocation. When you specify the trace type on the -START TRACE command, you can use the abbreviations listed below:

For statistics: STAT or S
For accounting: ACCTG or A
For performance: PERFM or P
For global: GLOBAL or G

Statistics Trace: To distinguish a DB2 statistics trace from other trace types, specify STAT when invoking it, as in -START TRACE(STAT).

The statistics trace provides usage measurements for the *system services* area and the *data base services* area of DB2. For the system services area, DB2 trace collects measurements for these subcomponents:

Agent services manager
Instrumentation facility
Latch manager
Log manager
Storage manager
Subsystem support

For the data base services area, DB2 trace collects usage measurements for the buffer manager and the lock manager. It also collects information about the use of SQL, BIND, and the EDM pool.

Statistics trace collects this data on a subsystem wide basis. You can use the information it produces to perform capacity planning for DB2 and to tune the entire set of active DB2 programs.

You can limit the statistics trace by class and destination using qualifiers on the -START TRACE command. There is, however, only one class of statistics data. Consequently, the keyword that concerns you is DEST. You cannot qualify a statistics trace by PLAN, AUTHID, or RMID.

If you specified YES for the TRACE AUTO START parameter (OPTRCAUT) on the Operator Functions Panel (DSNTIPO), statistics trace starts automatically when you start DB2. From this panel, you can also control the statistics collection interval (STATISTICS TIME parameter (OPSTATIM)), as well as whether DB2 will send statistics data to SMF (SMF STATISTICS parameter (OPSMFSTA)). Before you invoke a statistics trace, you should read "Recording Trace Data in SMF" on page 182.

Accounting Trace: To distinguish the DB2 accounting trace from other trace types, specify ACCTG when invoking it, as in -START TRACE(ACCTG).

The accounting trace provides data related to:

- Start and stop times
- Number of commits and aborts
- Counts of the use of certain SQL statements
- Counts of buffer manager requests
- Counts of certain locking events

- CPU times

DB2 trace begins collecting this data when a thread connects to DB2. DB2 trace writes a completed record when the thread terminates or when the authorization identifier changes. You can use this data to perform program-related tuning and to assess and charge DB2 costs.

You can limit an accounting trace by class, destination, plan, and authorization ID using qualifiers on the -START TRACE command. The keywords that most concern you are CLASS, DEST, PLAN, and AUTHID. You cannot qualify an accounting trace by resource manager identifier (RMID).

Accounting data for class 1 (the default) is accumulated by several DB2 components during normal execution. This data is then collected at the end of the accounting period. It does not involve as much overhead as does individual event tracing.

On the other hand, when you start class 2 or class 3, many additional trace points are activated. Then every occurrence of those events is traced internally by DB2 trace, but these traces are not written to any external destination. Rather, the accounting facility uses these traces to compute the additional total statistics that appear in the accounting record when class 2 or class 3 is activated. Accounting class 1 must be active to externalize the information.

If you specified YES for the TRACE AUTO START parameter (OPTRCAUT) on the Operator Functions Panel (DSNTIPO), accounting trace starts automatically when you start DB2. From this panel, you can also control whether DB2 will send accounting data to SMF (SMF ACCOUNTING parameter (OPSMFACT)). SMF records the accounting data in type 101 records.

Before you invoke an accounting trace, you should read "Recording Trace Data in SMF" on page 182.

Performance Trace: To distinguish DB2 performance trace from other trace types, specify PERFM when invoking it, as in -START TRACE(PERFM).

The performance trace provides data related to a variety of subsystem events. You can use this data to monitor and tune DB2 programs and resources for individual users, as well as for the entire subsystem.

You can limit the performance trace by class, destination, plan, authorization ID, and resource manager ID using qualifiers on the -START TRACE command. These qualifiers are: CLASS, DEST, PLAN, AUTHID, and RMID.

You cannot start collection of performance data when you install or migrate DB2. It can be started only with the -START TRACE command.

Global Trace: For information on the DB2 global trace, see *IBM DATABASE 2 Diagnosis Reference Volume 4: Service Traces.*

Specifying a Class

The CLASS keyword specifies a particular class or type of trace events for each trace. For example, to initiate data collection for all performance events within performance class 2, you can specify:

```
-START TRACE(PERFM) CLASS(2)
```

The classes for the three trace types discusses here appear in Figure 82 on page 176. Each of these classes includes many subsystem events. These events are identified by instrumentation facility component identifiers (IFCIDs). There are about 140 different IFCIDs. They are described in Appendix D, "DB2 Trace Record Descriptions" on page 258.

Trace Type	Class	Description of Class
Accounting	1	Standard accounting data
	2	Elapsed processor time in DB2
	3	Elapsed wait time in DB2
	4	Reserved
Statistics	1	Statistics data
	2	Reserved
	3	Reserved
Performance	1	Background events
	2	Subsystem events
	3	SQL events
	4	Buffer manager I/O
	5	Log Manager I/O
	6	Lock information
	7	Lock detail
	8	Data manager detail
	9	Sort Detail
	10	Optimizer and BIND detail
	11	Dispatching
	12	Storage manager
	13	Edit and validation
	14	RARQ Entry and exit
	15	Reserved

Figure 82. Classes for DB2 Trace Types

The default on accounting and statistics traces is class 1. On performance trace, the default is classes 1, 2, and 3. If you specify an asterisk for the class parameter, all classes within the trace type will be invoked. For instance, `-START TRACE(PERFM) CLASS(*)` starts a trace for classes 1 through 15 of performance data. You probably do not want to do this; it can produce an enormous amount of data.

You may, however, invoke a trace for several classes at once. For instance, `-START TRACE(PERFM) CLASS(4,5)` invokes data collection for classes 4 and 5 of performance data.

Specifying a Destination

The DEST keyword specifies the location to which trace data will be sent. For the three trace classes discussed here, these destinations include:

SMF—System Management Facility
GTF—Generalized Trace Facility
SRV—Serviceability Routine

The default destination for accounting, statistics, and performance data is SMF. You can also send these three types of trace data to GTF and SRV. The following chart illustrates the default and the possible destinations for accounting, statistics, and performance data. If you omit the DEST keyword from the -START TRACE command, DB2 uses the default destination.

Trace	Default Destination	Other Possible Destinations
Accounting	SMF	GTF and SRV
Statistics	SMF	GTF and SRV
Performance	SMF	GTF and SRV

Figure 83. Trace Default and Other Possible Destinations

Trace data can be sent to more than one destination. To send data to both SMF and GTF, specify:

```
-START TRACE(PERFM) DEST(SMF,GTF)
```

When specifying a location with the DEST keyword, use the appropriate abbreviation. Specify `DEST(GTF)` instead of `DEST(Generalized Trace Facility)`.

When determining the destination to which you want to send trace data, consider the volume as well as the type of data you are tracing. If you expect the data collected by your trace to be voluminous, you may want to use GTF. For instance, if you activate performance class 7, which records data about every lock requested in DB2, you may want to send your data to GTF.

For daily monitoring, however, you may want to use SMF. DB2 spans GTF records longer than 256 characters, and this can complicate your analysis or processing of trace data. For instance, if you accept the default classes for performance, accounting, or statistics trace, you may want to use SMF.

The use of SRV is reserved for IBM serviceability personnel. See *IBM DATABASE 2 Diagnosis Reference Volume 4: Service Traces* for further information.

Specifying a Plan

The PLAN keyword limits the trace to a particular DB2 plan or plans. This option cannot be used for a statistics trace. Up to 8 plan names may be specified. The default is an asterisk (*), which invokes the trace for all plans.

If you specify more than one plan, multiple traces are invoked. This example invokes a trace for SAMPLE1 and a trace for SAMPLE2:

```
-START TRACE(PERFM) PLAN(SAMPLE1,SAMPLE2)
```

If you invoke traces for more than one plan, you can specify only one authorization ID.

Specifying an Authorization ID

The AUTHID keyword limits the trace to the activity of a particular DB2 authorization ID. Up to 8 authorization IDs may be specified. The default is an asterisk (*), which invokes the trace for all authorization IDs.

If you specify more than one authorization ID, multiple traces are invoked. This example invokes one trace for USER1 and one trace for USER2.

```
-START TRACE(ACCTG) AUTHID(USER1,USER2)
```

If you invoke traces for more than one authorization ID, you can specify only one plan.

Specifying a Resource Manager

The RMID keyword limits the trace to the activity of a particular DB2 resource manager. You cannot use this keyword for a statistics or accounting trace. Up to 8 resource managers may be specified. The default is an asterisk (*), which invokes the trace for all resource managers. Figure 84 identifies the DB2 resource managers you can specify when invoking a performance trace.

RMID	Resource Manager
1	Initialization procedures
2	Agent services management
3	Recovery management
4	Recovery log management
6	Storage management
7	Subsystem support for allied memories
8	Subsystem support for SSI functions
10	Buffer management
12	System parameter management
14	Data manager
16	Instrumentation commands, trace, and dump services
18	Data Space management (operates in the data base services area)
19	Data Space management (access method services in system services)
20	Service controller
21	Data Base utilities
22	Relational database support
23	General command processing
24	Message generator
26	Instrumentation accounting and statistics

Figure 84. Resource Manager Identifiers (RMIDs)

Unlike the PLAN and AUTHID keywords, multiple traces are not started if you specify more than one RMID. This example invokes a single trace for 5 resource managers:

```
-START TRACE(PERFM) RMID(1,7,14,18,24)
```

Specifying a Comment

The COMMENT keyword enables you to append a descriptive phrase to trace data sent to SMF or GTF. This information can help you distinguish between multiple traces. In SMF or GTF, comments appear within the record to document that a trace was started. (The comment does not appear in the resident trace table.)

You may enter any character string for the comment. You must enclose blanks, commas, and special characters within quotation marks. For instance, you can specify:

```
-START TRACE(PERFM) CLASS(4,6,7,10) COMMENT('Investigate BIND times')
```

This starts a trace for performance classes 4 (buffer manager I/O), 6 (lock information), 7 (lock detail), and 10 (optimizer and bind detail) and sends the data for these classes to the default destination — SMF. The record written to SMF indicates a trace has been started and contains the comment "Investigate BIND times."

Displaying and Stopping Trace Activity

With DB2 you can display active trace information and stop trace activity. The commands performing these two functions use the same keywords, so they are discussed together here.

Displaying Trace Activity

The -DISPLAY TRACE command displays information about the traces invoked, including the options in effect. Because other traces may be started and stopped while you are viewing -DISPLAY TRACE output, the information presented by the -DISPLAY TRACE command may change immediately after the display is complete.

To display all the active traces, enter:

```
-DISPLAY TRACE
```

The output would look similar to this:

```
TNO TYPE     CLASS        AUTHID   PLAN     DEST  RMID
1   GLOBAL   *            *        *        GTF   *
2   GLOBAL   *            SYSADM01 PLAN8888 GTF   07,14
3   ACCTG    *            *        *        SMF   *
4   STAT     *            *        *        SMF   *
5   PERFM    01,02,03,04  SYSADM01 *        SMF   *
```

TNO shows the trace number, explained in "Trace Numbers" on page 181.

An asterisk can appear for keywords CLASS, AUTHID, PLAN and RMID. It cannot appear for TNO, TYPE, or DEST. The asterisk means one of the following:

- The keyword does not apply to the type of trace invoked. For instance, you cannot specify the RMID keyword for an accounting trace. Consequently, an asterisk appears under the RMID column for the accounting trace in the above example, or
- All valid parameters for the keyword are active. This is caused either by specifying an asterisk for the keyword when starting the trace, or by accepting the default for the keyword (which is an asterisk) when starting the trace.

Stopping Trace Activity

The -STOP TRACE command terminates trace activity. To stop a trace, you may not need to specify all the keywords you specified when starting the trace. Just provide sufficient detail to stop the trace. For instance, you may have started a trace with the following command:

```
-START TRACE(PERFM) CLASS(1,2,5,6) DEST(GTF) RMID(10,12,24)
```

This command contains several keywords that limit the trace. However, you can stop the trace by entering `-STOP TRACE(PERFM)`. This provides sufficient detail to stop the trace. Actually, this command stops all performance traces.

If several traces were active and you wanted to stop only one of them, you might need to provide more detail in the -STOP TRACE command. Assume you start three traces as follows:

```
-START TRACE(PERFM) CLASS(1,2,5,6) DEST(GTF) RMID(10,12,24)

-START TRACE(PERFM) CLASS(12) RMID(10,12,24)

-START TRACE(ACCTG) AUTHID(SYSADM01)
```

You can stop the first of these three traces in several ways without disturbing the other two. You could enter `-STOP TRACE DEST(GTF)`. Because the first trace is the only one sending data to GTF, specifying the trace destination provides sufficient detail.

Qualifying a Display or Stop Trace Command

All keywords for the -START TRACE command are valid for the -DISPLAY TRACE and -STOP TRACE commands. The -DISPLAY TRACE and -STOP TRACE commands have one additional keyword, TNO, which allows you to display or stop activity for a particular trace number. This is explained in further detail in "Trace Numbers" on page 181.

Figure 85 summarizes the valid keywords for the -DISPLAY TRACE and -STOP TRACE commands discussed in this book.

Keyword	Default	Other Values	Comments and Restrictions
TRACE	*	ACCTG or A, STAT or S, PERFM or P,	None
CLASS	*	Any valid class number; see Figure 82 on page 176.	You cannot specify a parameter for CLASS if you do not specify a parameter for TRACE. In addition, if multiple classes were specified on a -START TRACE and multiple classes are specified on a -STOP or -DISPLAY TRACE, the values must match.

Figure 85 (Part 1 of 2). Summary of Keywords for Trace Commands

Keyword	Default	Other Values	Comments and Restrictions
DEST	*	GTF, SMF, and SRV	If multiple classes were specified on a -START TRACE and multiple classes are specified on a -STOP or -DISPLAY TRACE, the values must match.
PLAN	*	From 1 - 8 valid PLANs	If you specify multiple PLANs, you can specify only one AUTHID or TNO.
AUTHID	*	From 1 - 8 valid AUTHIDs	If you specify multiple AUTHIDs, you can specify only one PLAN or TNO.
RMID	*	From 1 - 8 valid RMIDs; see Figure 84 on page 178.	If multiple classes were specified on a -START TRACE and multiple classes are specified on a -STOP or -DISPLAY TRACE, the values must match.
TNO	*	From 1 - 8 valid trace numbers	See "Trace Numbers" for further information. If you specify multiple TNOs, you can specify only one PLAN and AUTHID.
COMMENT	Blanks	Any character string	The comment appears within the display trace record that is sent to SMF, GTF, or SRV.

Figure 85 (Part 2 of 2). Summary of Keywords for Trace Commands

Trace Numbers: When you invoke a trace, DB2 assigns it a trace number (TNO) from 1 to 32. This appears in the standard trace record header DB2 sends to SMF and GTF as a bit mask value. No more than 32 traces may be active at one time.

You can use this number to stop or to display trace activity. For instance, you may have started two traces with these commands:

```
-START TRACE(STAT) DEST(GTF)

-START TRACE(ACCTG) AUTHID(USER1)
```

DB2 assigns a number to each of these traces. If these are the only active traces, their TNOs are 1 and 2, respectively.

You can stop the first trace by entering:

```
-STOP TRACE TNO(1)
```

Similarly, you can display information about the second trace by entering:

```
-DISPLAY TRACE TNO(2)
```

This is the output the display command produces:

```
TNO TYPE     CLASS        AUTHID   PLAN     DEST  RMID
2   ACCTG    *            USER1    *        SMF   *
```

When you specify a trace number, you do not need to specify any other keywords or parameters. The trace number provides a sufficient level of detail for DB2 to stop or display trace activity.

Multiple Traces: One -START TRACE command can invoke multiple traces. Only certain keywords, however, invoke multiple traces.

- Multiple parameters specified for PLAN or AUTHID invoke multiple traces. This command invokes three traces:

  ```
  -START TRACE(ACCTG) AUTHID(USER1,USER2,USER3)
  ```

  You can display and stop these traces individually or collectively.

- Multiple parameters specified for CLASS, DEST, or RMID invoke only one trace. This command invokes one trace:

  ```
  -START TRACE(PERFM) CLASS(1,2,3) DEST(SMF,GTF)
  ```

  You cannot display or stop individual classes or destinations started by this command.

Use DB2 Performance Monitor (DB2PM) to format, print, and interpret DB2 trace output. If you don't have DB2PM or wish to do your own analysis of the DB2 trace output, refer to Appendix C, "Interpreting DB2 Trace Output" on page 252.

Affect on DB2 Performance

The volume of data DB2 trace collects can be quite large. Consequently, the number of performance trace records you request will affect system performance. When you activate a performance trace, you should qualify the -START TRACE command with the particular CLASS(es), PLAN(s), AUTHID(s), PLAN(s), and RMID(s) you want to trace.

Recording Trace Data in SMF

Each site is responsible for processing the SMF records produced by DB2 trace. You can use the SMF program IFASMFDP to dump these records to a sequential data set. You may want to develop an application to process these records. For more information about SMF, refer to *MVS/370 System Programming Library: System Macros and Facilities Volume 2*.

Activating SMF: SMF must be running before you can send data to it. To make it operational, update member SMFPRMxx of SYS1.PARMLIB. This member indicates whether SMF is active and which types of records SMF will accept. To update it, specify the ACTIVE parameter and the proper TYPE subparameter for SYS and SUBSYS.

During DB2 execution, you can use the SMF SET or SS command to alter SMF parameters you specified previously. For example, the following command records statistics (record type 100), accounting (record type 101), and performance (record type 102) data to SMF.

```
SYS(TYPE(100:102))
```

You can also code an IEFU84 SMF exit to process the records that are produced.

Allocating Additional SMF Buffers: The volume of data that DB2 can collect when you specify a trace type of performance can be quite large. If you are sending this data to SMF, you must allocate adequate SMF buffers. The default buffer settings will probably be insufficient.

If an SMF buffer shortage occurs, SMF will reject any trace records sent to it. DB2 sends a message (DSNW133I) to the MVS operator when this occurs. DB2 treats the error as temporary and remains active even though data may be lost. DB2 sends a message (DSNW123I) to the MVS operator when the shortage has been alleviated and trace recording has resumed.

You can determine if trace data has been lost by examining the DB2 statistics records with an IFCID of 0001. These records show:

- The number of trace records successfully written
- The number of trace records that could not be written
- The reason for the failure

If your site uses SMF for performance data or serviceability data, ensure that:

- Your SMF data sets are large enough to hold the data.
- SMF is set up to accept record type 102 (specify member SMFPRM*xx*).
- Your SMF buffers are large enough.

  For MVS/XA, specify SMF buffering on the VSAM BUFSP parameter of the access method services DEFINE CLUSTER statement. Do not use the default settings if DB2 performance or serviceability data is sent to SMF. Specify `CISZ(4096)` and `BUFSP(81920)` on the DEFINE CLUSTER statement for each SMF VSAM data set.

  DB2 runs in a cross memory environment and above the 16Mb line of MVS/XA virtual storage. SMF has the following restrictions:
  - SMF does not accept records when the MVS-dispatched home address of the caller does not equal the primary address of the caller. This means DB2 must buffer information until the MVS environment is correct for SMF to write the records. Special buffering is required to handle storage manager and agent services data in a cross memory environment. DB2 schedules a service request block to write SMF buffers if the number of bytes buffered for SMF exceeds a predetermined maximum. If an unexpected abend occurs in the middle of the buffering routines (such as TSO attention), DB2 SMF data could be lost.
  - SMF does not accept buffers above the 16Mb line of MVS/XA virtual storage. Thus, DB2 uses CSA buffers below the line when forced to queue buffers because of the SMF cross memory restriction.

  For MVS/370, you control SMF buffering with the BUFNUM keyword on the SMF parmlib member. To ensure adequate buffers exist, specify `BUFNUM(4,15)` on the SMFPRM*xx* parameter.

Recording Trace Data in GTF

The MVS operator must start GTF before you can send data to it. (For more information on GTF, see *MVS/Extended Architecture Service Aids Logic.*) GTF should be started specifying YES for the TIME option and specifying that user records are to be recorded. To start GTF with these options, enter:

```
S GTF,,,(TIME=YES)
```

The system will respond:

```
AHL100A  specify trace options
```

Then enter:

```
TRACE=USR
```

You can record statistics, accounting, performance, and global trace data in GTF using a GTF EID of X'0FB9'. Trace records longer than the GTF limit of 256 bytes are spanned by DB2.

You can use the following logic to process GTF records:

1. Is the GTF EID of the record equal to the DB2 ID (that is, does QWGTEID = X'0FB9')?

   If it is *not* equal, get another record.

   If it is equal, continue processing.

2. Is the record spanned?

   If it is *not* spanned, process the record.

   If it is spanned (that is, QWGTDSCC ¬= QWGTDS00), test to determine whether it is the first, a middle, or the last statement.

   a. If it is the *first* segment (that is, QWGTDSCC = QWGTDS01), save the entire record including the sequence number (QWGTWSEQ) and the subsystem ID (QWGTSSID).

   b. If it is a *middle* segment (that is, QWGTDSCC = QWGTDS03), find the first segment by matching on the sequence number (QWGTSEQ) and on the subsystem ID (QWTGSSID). Then move the data portion immediately after the GTF header to the end of the previous segment.

   c. If it is the *last* segment (that is, QWGTDSCC = QWGTDS02), find the first segment by matching on the sequence number (QWGTSEQ) and on the subsystem ID (QWTGSSID). Then move the data portion immediately after the GTF header to the end of the previous record.

      Now process the completed record.

Chapter 8. Tuning DB2

This chapter presents an overview of the tuning process and suggests several ways you can tune DB2 to upgrade performance. Tuning the DB2 subsystem is an iterative process. That is, you conduct tuning on an ongoing basis, as the need arises.

Many factors affect the performance of DB2. Overemphasizing one factor can create a problem in another area. You should configure DB2 to establish a balance of all factors.

The factors discussed in this section relate to overall subsystem performance, rather than to application design or data base design. Performance guidelines for application design are presented in *IBM DATABASE 2 Application Programming Guide* that applies to your particular application environment. Performance guidelines for data base design are presented in *IBM DATABASE 2 Data Base Planning and Administration Guide*.

Tuning Strategy

Tuning your DB2 subsystem involves these steps:

1. **Assess DB2 performance**

   In most cases, this step means monitoring your system over time, making note of any degradation in performance. You may notice that subsystem response is becoming more sluggish, or more applications are beginning to fail from lack of resources (virtual storage constraints or locked tables). With regular use of a performance monitoring tool over time, you may notice at one point an increase in the amount of processor time DB2 is using, even though subsystem responses appear normal. If, however the subsystem continues to perform with acceptable speed and you are not having any problems, DB2 may not need further tuning.

2. **Monitor DB2**

   If you have a performance problem, you should first verify that the problem is not being caused by improper application or data base design. If you suspect the problem is caused by improper application design, consult *IBM DATABASE 2 Application Programming Guide*. If you suspect the problem is caused by improper data base design, consult *IBM DATABASE 2 Data Base Planning and Administration Guide*.

   If you suspect that the problem is caused by the selection of install parameters, I/O device assignments, or other factors, begin monitoring DB2 to collect data describing its internal activity. The facilities that allow you to monitor these things are described in Chapter 7, "Monitoring DB2" on page 167.

3. **Identify bottlenecks**

   After you have collected data about DB2 internal activity, you should look for unusual statistics or counts. You may find several of these unusual values, and you will be able to use them to determine how to modify the configuration of your subsystem.

4. **Modify DB2 configuration**

   After you have determined the probable source of your performance problem, you may need to modify the configuration of DB2. This can involve reassignment of data sets to different (faster) I/O devices, running the RUNSTATS utility, creating indexes, rebinding applications, or modifying some of your install parameters. For a description of how to modify install parameters, see Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.

   After modifying the configuration, monitor DB2 for changes in performance. The modifications may improve the performance of DB2 to the level that you require. If not, you should repeat the process to determine if the same or different problems exist.

Tuning Techniques

DB2 tuning strategy involves several related factors, including speed, DASD utilization, virtual storage utilization, concurrency, processing large tables, use of the bind process, size of active logs, and size and location of your DB2 catalog and directory. A discussion of each of these factors follows. At the end of this chapter, "Tuning Strategy Summary" on page 202 includes a table listing most of these tuning factors and showing relationships among them.

Improving Speed

You can upgrade the speed of your DB2 subsystem by (1) reducing the number of I/O operations, (2) reducing the time needed to perform I/O operations, and (3) reducing the amount of processor resources consumed. This section describes ways to accomplish these goals.

Customers with production systems requiring high performance might consider turning off global trace. You should be aware, however, that this would present a serviceability exposure. In the event of a system failure, IBM service personnel would probably request that you turn on global trace and attempt to re-create the problem.

Reducing the Number of I/O Operations

Reducing the number of I/O operations is one method to improve DB2 speed. This section describes the various options available to help you accomplish this.

Establish Indexes: Adding appropriate indexes can reduce the amount of I/O required by eliminating the need to scan an entire table to find the desired records. This is the most important factor for reducing I/O. However, there is a trade off, because each update requires a corresponding update to the applicable indexes.

As a guideline, consider creating indexes for frequently used, read-only tables that contain more than five data pages. Try to create indexes on columns frequently used in searches, including joins.

For read-only tables that are accessed infrequently, consider creating indexes only if the tables contain more than 10 data pages. Otherwise, maintaining an index could become more costly than occasionally scanning the tables.

For frequently updated tables, consider creating indexes only when the tables contain more than 15 data pages.

When deciding whether to create an index, you must balance the reduction in I/O required to reference the table against the additional I/O required to update it.

Invoke RUNSTATS: The RUNSTATS utility collects statistics about DB2 objects and stores this information in the DB2 catalog. This information is used during bind to choose the path in accessing data. If RUNSTATS is not run, an inefficient access path may be selected for subsequently bound application plans, resulting in unnecessary I/O operations. It is especially important that the RUNSTATS utility be used on table spaces that contain frequently accessed tables, tables involved in a join or sort (ORDER BY), tables against which SELECT statements having many search arguments are performed, and tables having a long life.

Reduce OPEN/CLOSE Operations: For frequently used table spaces and indexes, specify CLOSE NO in the CREATE TABLESPACE and CREATE INDEX statements. This will cause the data set to be left open when not in use, avoiding the need to repeatedly open and close the data set. As a result, fewer I/O operations will be performed.

For infrequently used table spaces and indexes, such as those accessed only by batch applications, specify CLOSE YES in the CREATE TABLESPACE and CREATE INDEX statements. This will reduce the unnecessary allocation of virtual storage.

Specify Proper Space Reservation Parameters: The PCTFREE and FREEPAGE parameters of the CREATE and ALTER TABLESPACE and CREATE and ALTER INDEX commands can help reduce access time to needed table spaces.

When data is loaded, PCTFREE reserves a percentage of each page as free space, which is used later when inserting or updating data. If no free space is available to hold the additional data, it must be stored on another page, perhaps in a distant location.

When several records are physically located out of sequence, performance suffers. Therefore, for tables requiring frequent inserts and updates, assign PCTFREE a value greater than the default of 5 (that is, 5 percent of the page is blank). However, for tables with few updates or inserts, consider assigning PCTFREE a value of 0 to save storage space. For indexes, the default is 10. The maximum value you can specify is 99.

FREEPAGE specifies how often DB2 is to leave a full page of free space when data is loaded. Free pages are available for inserts and updates in the same way that free space on a page is available. Free pages are particularly important for a table whose records fill a whole page; in that case, there could be no free space on the page. In most other cases, increase PCTFREE and leave FREEPAGE at its default value of 0. The maximum value you can specify is 255.

You can change the values of PCTFREE and FREEPAGE any time, but they are effective only after a LOAD or the LOAD part of a REORG utility has been performed. To determine the amount of free space currently on a page, run the RUNSTATS utility and examine the PERCACTIVE column of SYSTABLEPART. See "Using the RUNSTATS Utility" on page 167 for details.

When you specify a sufficient amount of free space, the advantages are:

Faster placement of row by space algorithm
Better clustering of rows (giving faster access)
Fewer overflows
Less frequent REORGs
Less information locked in a page
Fewer index pages splits

The disadvantages are:

More DASD occupied
Less information per I/O
More pages for SCAN
Possibly more index levels

The following are recommendations for free space:

- Tailor the amount of free space to the anticipated growth of the data. Leave sufficient free space for expected inserts/expanded rows and no free space for non-expanding data.
- Leave free pages for inserts of big rows (larger than half a page), because you may be unable to use free space in a used page.
- Consider additional free space to lessen REORG activities.
- Consider extra free space for small tables/shared table spaces to increase concurrency.

Increase Buffer Pool Sizes: Large buffer pool sizes enable frequently used data to be kept in virtual storage, reducing the amount of I/O necessary. However, you should choose pool sizes backed by real storage to minimize paging I/O.

Using a single 4K-byte buffer pool (BP0) will give better overall performance than using multiple buffer pools. The buffer pool should be as large as possible, considering the real storage and expanded storage you have available. Larger buffer pool sizes result in fewer I/O operations and therefore better response time. A large buffer pool may also prevent I/O contention for the most frequently used DASD devices, particularly the catalog tables and frequently referenced user tables and indexes. In addition, a large buffer pool is of benefit when the DB2 sort is used (ORDER BY), because I/O contention on the device containing the temporary table space (DSNDB07.DSNTMP01) will be reduced.

Monitor buffer pool usage with DB2PM to ensure that your buffer pools are large enough to take advantage of the performance features of DB2. For example, if the percentage of available buffers drops below 25 percent, sequential prefetch will be disabled. Another important threshold to monitor is buffer critical, which occurs when less than 5 percent of the buffer pages are available. When no buffer pages are available, buffer pool expansion will occur. This is a very expensive operation that should be avoided. To avoid buffer pool expansion, consider setting the maximum number of buffers in the buffer pool equal to the minimum number. To do this, use the MIN (BUFMIN00) and MAX BP0 BUFFERS (BUFMAX00) parameters on the Storage Sizes Panel (DSNTIPE). (The IBM-supplied default values for these parameters are equal values.)

The use of a 32K-byte buffer pool should be carefully considered. Data in table spaces that use a 32K-byte buffer pool is stored and allocated as 8 records, each 4K

bytes in size. Inefficiencies can occur if small records are stored in table spaces that use a 32K-byte buffer pool.

Control SYSLGRNG Size: For recovery purposes, log RBA information is recorded in the SYSLGRNG data set when a table space is first changed and when a changed table space is closed. SYSLGRNG size may grow rapidly, causing multiple I/O operations each time a table space is changed. Such an increase is especially likely if frequently used table spaces or indexes were created with the CLOSE YES option in the CREATE TABLESPACE or CREATE INDEX statements. Monitor the SYSLGRNG data set size periodically and run the MODIFY utility to delete old records.

Reducing the Time Needed to Perform I/O Operations

You can reduce the time needed to perform individual I/O operations in several ways, each of which is described in this section.

Allocate Additional Temporary Table Spaces: If your applications require large concurrent sorts, allocate additional temporary table spaces in the temporary data base (DSNDB07). This helps minimize I/O contention.

During the install or migration process, you allocated two table spaces: one for 4K-byte buffering and one for 32K-byte buffering. You can allocate additional tables spaces for either of these pools. With user-defined data sets, you can spread temporary table spaces across different volumes. In addition, you can place temporary table spaces on multiple devices to support large temporary files.

To create additional temporary table spaces, use statements similar to those in job DSNTIJTM. This is the job you ran during install or migration to define the temporary table spaces. It is stored as a member of the *outprefix*.DSNSAMP created during install or migration.

Otherwise, you can use DDL to create additional temporary table spaces. However, before you issue the DDL, you must stop all temporary table space users by stopping DSNDB07. Restart DSNDB07 after you finish executing the DDL. For instance, issue the following statements, where *xyz* is the name of the table space:

```
-STOP DATABASE (DSNDB07)
CREATE TABLESPACE XYZ IN DSNDB07
    BUFFERPOOL BP0
    CLOSE NO
    USING VCAT DSNC130
    DSETPASS DBADMIN
GRANT USE OF TABLESPACE DSNDB07.XYZ TO PUBLIC;
-START DATABASE (DSNDB07)
```

You cannot run DB2 utilities or CREATE TABLE statements against the temporary data base (DSNDB07).

Use Fast Devices: Assign the most frequently used data sets to the faster DASD devices at your disposal. You may do this by creating a partitioned table space, which would allow you to split a table into different data sets and place the frequently used data on a fast device.

Alter Data Set Distribution: Frequently used data sets should be allocated across your available DASD so that I/O operations are distributed as evenly as possible. This will help reduce I/O contention.

Ensure Sufficient Primary Allocation Quantity: Specifying sufficient primary allocation for frequently used data sets minimizes I/O time, because the data will not be physically located at different places on the DASD device. This will enhance DB2 performance.

It may be helpful to list the VTOC occasionally to determine the number of secondary allocations that have been made for your more frequently used data sets.

If you discover that the data sets backing frequently used table spaces or indexes have an excessive number of extents, and if the data sets are user-defined, you can use access method services to reallocate the affected data sets using a larger primary allocation quantity. If the data sets were created using STOGROUPs, you can use the procedure for modifying the definition of table spaces presented in *IBM DATABASE 2 Data Base Planning and Administration Guide.*

Reducing the Amount of Processor Resources Consumed

Many factors affect the amount of processor resources that DB2 consumes. This section describes ways to reduce DB2 consumption of these resources.

Use a Processor with Hardware Cross-Memory Operations: Software simulation of cross-memory operations is costly in terms of processor utilization. Using a processor with hardware cross-memory operations can reduce consumption of processor resources.

Reuse Threads: For high volume transactions, reusing threads can help performance significantly. For IMS/VS, process multiple input messages with one transaction by setting PROCLIM to a value greater than 1. Alternatively, you can reuse threads with WFI (wait for input) and class scheduling. See "IMS/VS Attachment Facility Options" on page 158 for details. For CICS, you can enhance thread reuse through specifications in the RCT. See "CICS Attachment Facility Options" on page 164 for details.

Skip DB2 Authorization Check for CICS: When a thread is reused, DB2 normally needs to check the authorization of the new user in order to execute the plan associated with the thread. In some environments, it is possible to skip this authorization check and its associated resource costs without causing a security exposure. The CICS attachment facility will skip the DB2 authorization check and execute a CICS plan only if it:

1. Reuses threads;
2. Does not use dynamic SQL; and
3. It maps only to a specific transaction.

To influence this optimization, when you define a thread for a transaction in DB2's resource control table (RCT) and tune the thread for reuse, you can also specify the AUTH parameter to indicate what should be used for the authorization ID that is passed to DB2. If you specify AUTH(SIGNID), the authorization ID will not change. Each time the thread is reused it will have the same ID and authorization checking will be skipped. Remember to grant plan execution authority to the CICS sign-on ID, since authorization will be checked the first time this thread is used.

Reduce the Amount of Overhead Processing: To do this, use

- **CLOSE NO**: Specify the CLOSE NO option of the CREATE TABLESPACE and CREATE INDEX statements when defining frequently used indexes or table spaces. This will greatly reduce the number of OPEN/CLOSE operations and thereby reduce the processing load.
- **Traces**: Using the DB2 trace facility, particularly performance and global trace, consumes a large amount of processing resources. Suppressing these trace options significantly reduces processor overhead.

  Under the best of circumstances, global trace still requires 20 to 100 percent of overhead processing. If conditions permit at your site, the DB2 global trace should be turned off. You can do this by specifying NO for the TRACE AUTO START (OPTRCAUT) parameter of the Operator Functions Panel (DSNTIPO). (Note that the IBM-supplied default value for this parameter is YES.) Then, should the global trace be needed for serviceability, you can start it using the START TRACE command.

  The DB2 accounting and statistics class 1 trace costs only about 2 to 5 percent. We recommend you enable this ongoing collection of performance information, as it is necessary for capacity planning. You can do this by specifying YES for the SMF STATISTICS (OPSMFSTA) and SMF ACCOUNTING (OPSMFACT) parameters of the Operator Functions Panel (DSNTIPO). (Note that the IBM-supplied default values for these parameters is NO.)

  Consider turning on only the performance trace classes required to address a specific performance problem. The combined overhead of performance classes 1 through 8 and 11 and 12 is about 200 percent.

  Performance trace classes 1 through 3 cost about 25 percent overhead. You may want to turn on class 3, which gathers the SQL statement text. This will make it much easier to track dynamic SQL calls (through QMF, for example). For further information on these traces, see "Using DB2 Trace" on page 172.

  Suppressing the IRLM and MVS trace options also reduces overhead, though not as significantly.
- **Fixed-length columns**: Use fixed-length columns rather than varying-length columns, particularly in tables containing many columns. This will reduce processor use.
- **Locking Options**: You can specify certain locking options to reduce the amount of processing resources consumed. Using RR (repeatable read) option when binding and the PC=NO option in the START IRLMPROC command will reduce processor overhead and result in faster communication between DB2 and the IRLM.

  You can also reduce processing overhead by using table space locks rather than page locks. Specifying LOCKSIZE TABLESPACE in the CREATE TABLESPACE statement forces a lock on the entire table space. This prevents other applications from accessing data with the table space while that table space is being updated. During this time, the processing overhead of locking and unlocking individual pages is avoided. However, any other applications trying to access this data are suspended (and possibly abended if the suspension time exceeds the value specified for WAIT TIME (IRLMWAIT) on IRLM Panel 1 (DSNTIPI). Therefore, you should use table space locking only for table spaces that require a low level of concurrency or that have read-only data.

A less drastic step might be to specify LOCKSIZE ANY in the CREATE TABLESPACE statement and specify a small value for the LOCK PER TABLE SPACE (IRLMLKTS) parameter on IRLM Panel 2 (DSNTIPJ) or on the Locking Update Panel (DSNTIPK). Doing this allows you to specify the maximum number of page locks a single application can hold concurrently against a given table space. When the application reaches the limit, DB2 escalates all page locks to a table space lock. By specifying a small value for LOCKS PER TABLE SPACE (IRLMLKTS), you reduce the number of page locks that are held, thereby reducing consumption of processor resources.

Improving DASD Utilization

Allocating data sets, using the Data Facility Hierarchical Storage Manager (DFHSM) to manage data sets, and using edit routines can affect your DASD utilization.

Allocating Data Sets: The main factor affecting the amount of DASD storage space DB2 uses is the way data sets are allocated. Specifying appropriate secondary allocation quantities can improve your DASD storage space utilization.

In general, your primary allocation quantity should be large enough to handle your anticipated storage needs, and secondary allocation quantities should be sufficient to allow your applications to continue operation until the data set can be reorganized.

When less than half of the secondary allocation quantity remains available in the current extent, DB2 obtains the next extent; therefore, you should be very careful about the primary versus secondary allocation sizes. Primary allocation quantities that are too small coupled with large secondary allocation quantities can result in a large amount of unused space.

Using DFHSM to Manage Data Sets: DFHSM works with software accompanying DB2 to automatically manage space and data availability among storage devices in your system. You can use DFHSM to ensure DASD space is managed efficiently by moving data sets that have not been used recently to slower, less expensive storage devices.

Data management occurs daily. Data sets can be deleted or migrated to another device. The space manager specifies the time data management will occur as well as how long data sets are kept before deletion or migration. All DFHSM operations can also be performed manually.

When using DFHSM with DB2, note that

- DFHSM can automatically migrate and recall log and image copy data sets.
- DB2 single-volume table spaces and index spaces can also be automatically migrated and recalled when DFHSM is operating in an MVS/XA environment. The appropriate DFHSM parameters must be specified to invoke the access method services support for EXPORT by control interval. To do this, specify `SETSYS EXPORTESDS (CIMODE)`. Similarly, the IMPORT function of the access method services can also be used with DB2 table spaces and indexes if you specify the CIMODE parameter when operating in an MVS/XA environment.
- If DB2 needs an archive log data set or image copy data set that has been migrated by DFHSM, a recall begins automatically and DB2 will wait for the recall to complete before continuing.
- If you accepted the default value NO for the WAIT FOR RECALL (OPRECALL) parameter on the Operator Functions Panel (DSNTIPO), DB2 ini-

tiates recall for migrated table spaces and index spaces but does not wait for completion. At this point DB2 indicates that the resource is unavailable and DFHSM performs the recall, DFHSM using CIMODE processing.

If you specified YES for the WAIT FOR RECALL (OPRECALL) parameter, at data set open time DB2 will wait for DFHSM to recall migrated table spaces and index spaces. In this case all subsequent data set open and close operations must wait until the recall of the migrated data set is complete. Since recall may take several seconds, this could have a significant effect on system response time.

Never specify YES for this parameter if data sets are migrated to tape, as recall of the data set would require waiting for the tape to be mounted.

- Note that the DB2 directory, catalog, and associated indexes should not be placed on volumes that are managed by DFHSM.
- If a volume has a STOGROUP specified, it should only be recalled to volumes of the same device type as others in the STOGROUP.

In addition, you must coordinate the DFHSM automatic purge period, the DB2 log retention period, and MODIFY utility usage. Otherwise, the image copies or logs you may need during a recovery may have already been deleted.

For more information about DFHSM, see the *Data Facility Hierarchical Storage Manager User's Guide.*

Using Edit Routines: Edit routines encrypt and compress data before storing it. They can be defined for each table by using the EDITPROC clause of the CREATE TABLE statement. Each time a row is retrieved, inserted, or updated, the edit routine receives control. Because the edit routine compresses data, your DASD requirement is reduced and so is the number of required I/O operations.

However, edit routines can cause a slight increase in processor use because each row must be processed. In addition, your data will become dependent on your edit routine. Thus, edit routines cannot be added or dropped after a table has been created. Similarly, columns cannot be added to a table using an edit routine.

Improving Virtual Storage Utilization

This section provides specific information for virtual storage planning. For a general overview of some factors relating to virtual storage planning, refer to "DB2 Virtual Storage Layout" on page 47.

Modify IRLM Specifications: The IRLM lock control block structure can be placed in the IRLM private address space by accepting the default YES for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), or by specifying PC=YES on the START IRLMPROC command. This may be helpful in MVS/370; in MVS/XA, this problem is avoided with the use of ECSA.

You can also save virtual storage by using the LOCKSIZE TABLESPACE option on the CREATE TABLESPACE statements for large tables. For more information on specifying LOCKSIZE TABLESPACE, see Locking Options on page 191.

Use Only One 4K-Byte Buffer Pool: Using only one 4K-byte buffer pool (BP0) and minimizing its size will help to reduce the amount of virtual storage space DB2 requires. If you do not use a 32K-byte buffer pool, you will also save approximately 72K bytes of virtual storage overhead, which is required with the 32K-byte buffer pool (in addition to the buffer pool storage). As a result, however, tables resulting from a join cannot have rows larger than 4K bytes.

If you do not use a 32K-byte buffer pool, delete data set DSNDB07.DSNTMP02, which is a temporary table space used with the 32K-byte buffer pool.

Buffer pool size can also affect the number of I/O operations performed. For information about this, see "Increase Buffer Pool Sizes" on page 188.

Reduce the Number of Open Data Sets: You can reduce the number of open data sets by having multiple tables per table space, using fewer indexes, and/or specifying CLOSE YES in the CREATE TABLESPACE statements for infrequently used table spaces, such as those accessed only by batch applications. Specifying CLOSE YES for frequently used table spaces will increase processor overhead because the data set must be opened and closed repeatedly.

Reduce the Global Trace Table Size: You can save virtual storage by minimizing the size of the global trace table. You specify this size with TRACE SIZE (OPTRCSIZ) on the Operator Functions Panel (DSNTIPO). See "Operator Functions Panel: DSNTIPO" on page 79 for details.

Reduce the Use of DB2 Sort: The use of DB2 sort increases the load on the processor, virtual, and real storage, and I/O devices. If possible, use appropriate indexes rather than a sort to obtain needed information; this will improve response time and overall performance.

DB2 sort is invoked when these operations are being performed on columns that are not indexed in the necessary order:

GROUP BY
ORDER BY
Join
DISTINCT
UNION (except UNION ALL)

The virtual storage used by DB2 sort is taken from the storage that is available after the buffer pools, EDM pool, and other fixed-sized storage areas are allocated.

Ensure ECSA Size is Adequate: DB2 places some of its load modules into common storage. These modules require primary addressability to any address space, including the application's address space. Some control blocks are obtained from common storage and require global addressability.

In an MVS/XA environment, with few exceptions, the CSA-resident load modules are link-edited with the residency attribute of RMODE(ANY). When the modules are loaded, MVS/XA should place them in ECSA (above the 16 Mb line of virtual storage). With equally few exceptions, global control blocks are GETMAINed with the attribute of LOC=ANY. They should also be placed in ECSA by MVS/XA. This includes the IRLM lock control blocks. If you specify NO for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), ensure that you add the value you used for the maximum CSA space to the other ECSA space noted below.

ECSA size is specified with the CSA keyword in member IEASSYS*nn* in SYS1.PARMLIB. The IBM-supplied default value is 1 Mb, but experience has shown this value to be too low for running a DB2 system. In general, at least 2 to 4 Mb of ECSA is recommended with DB2 installed, plus the ECSA space for the IRLM and other subsystems.

At IPL time, the CSA size can be overridden by the operator. The syntax is:

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

where:

- *a* is the number of K bytes of CSA storage below the 16 Mb line
- *b* is the number of K bytes of CSA storage above the 16 Mb line

These values are rounded down (CSA) or up (ECSA) to the next 1 Mb boundary. For more information, see *MVS/Extended Architecture System Programming Library: Initialization and Tuning*.

If the ECSA size (*b* value above) is too small, MVS/XA will place DB2's global load modules and control blocks in CSA below the 16 Mb line instead of above it. This can be undesirable if, for example, the CSA below the 16 Mb line has been sized for a coexisting IMS/VS subsystem.

DB2 needs approximately 320K bytes of ECSA storage for the global load modules. Global control block ECSA storage requirements are a function of the number of connected address spaces and tasks, as well as the work requests to be performed on behalf of the connected tasks.

Carefully evaluate your existing specification of ECSA size and increase it if necessary. Monitoring CSA below the 16 Mb line can indicate whether or not you need to increase the size of ECSA. DB2 will use approximately 160K of key 7 storage below the 16 Mb line. CSA fragmentation may cause these values to be higher. However, if measurements show that DB2 is using far in excess of these values, specifying a larger ECSA may reduce CSA usage.

Install DFP Version 2: VSAM ICF catalog support in MVS/XA DFP Version 1 requires about 20K of key 0 below the 16 Mb line of CSA virtual storage for each opened catalog. MVS/XA DFP Version 2 introduces the catalog address space and provides that much CSA virtual storage relief per opened catalog.

If you have DFP Version 2 installed, use several ICF catalogs for DB2, each one having one or more aliases that might be defaulted in a storage group. However, unless you have DFP Version 2 installed, more ICF catalogs will require more CSA storage.

Improving Concurrent Data Access

Reducing the number of locking contentions improves concurrent data access. This section describe the various ways to reduce locking contentions.

Use LOCKSIZE PAGE and CS: The use of the CS (cursor stability) isolation level (on BIND) together with page locks, rather than table space locks, will maximize application concurrency. Do one of the following:

1. Specify LOCKSIZE PAGE as an option in the CREATE TABLESPACE statement.
2. Specify LOCKSIZE ANY for the CREATE TABLESPACE statement and

specify a large value for the LOCKS PER TABLE SPACE (IRLMLKTS) parameter. This parameter is on the update panel DSNTIPK. It allows you to specify the maximum number of page locks that can be concurrently held by a single application against a single table space. When the application reaches the limit, DB2 escalates all page locks to a table space lock. By specifying a large value for this parameter, you defer lock escalation to the table space level, thereby maximizing concurrency.

To maximize concurrency, you can also specify the PC=YES option of the START IRLMPROC command, which will reduce the amount of CSA usage, thereby allowing more storage for concurrent locks.

Increase the Number of SUBPAGES: Using many subpages will increase the possible concurrent access to frequently used indexes. The SUBPAGES parameter of the CREATE INDEX statement controls this.

Issue COMMIT and ROLLBACK when Appropriate: To avoid unnecessary lock contentions, issue a COMMIT statement as soon as possible after reaching a valid point of consistency. This includes read-only applications that are bound with RR. To prevent unsuccessful SQL statements (such as PREPARE) from holding locks, issue a ROLLBACK statement after a failure. However, issuing COMMIT and ROLLBACK may increase processor use slightly.

Improving Processing of Large Tables

The use of large tables can cause a heavy load on system resources. This section describes significant factors for working with large tables.

Use Indexes: Using indexes properly is critical with large tables. When an index is not used, the entire table must be scanned. For guidelines on how to establish indexes, refer to "Establish Indexes" on page 186. For information on determining index efficiency, see "Using the RUNSTATS Utility" on page 167.

Place Large Tables in Partitioned Table Spaces: Placing large tables in partitioned table spaces can help limit the cost of running utility jobs (such as REORG or image copy). Partitions are somewhat independent of each other. Each can be reorganized and recovered without affecting the other partitions. However, the entire table space is locked during that time.

Partitioning also allows you to place frequently used data on faster devices, improving the time needed to perform I/O operations.

Use Incremental Image Copies: For very large tables in which the number of pages updated is less than 1 percent, an incremental image copy may be faster than a full image copy.

In an incremental image copy, all pages containing any updated records are copied. These can then be merged into a full copy using the MERGE COPY utility. The decision whether to perform incremental or full image copy depends on the percentage of pages containing at least one updated record, not the number of records updated. Regardless of the size of the table, if more than 10 percent of the pages contain updated records, use full image copy (this saves the cost of a subsequent merge copy).

Reduce the Impact of DFSORT: The LOAD, REORG and RECOVER utilities use DFSORT. To reduce the cost of these sorts, try to:

- Use cylinder allocation for sort work data sets. This ensures that the optimal sort technique is used.
- Allocate sort work data sets on fast DASD devices (like 3380). Sort performance is very sensitive to the data transfer rate of the device.

To enhance the efficiency of DFSORT, specify SIZE=MAX and set MAXLIM in the ICEMAC macro and the region size to more than 512Kb (these are DFSORT parameters). If MAXLIM is greater than the region size and if sufficient real storage is available to back the region, DFSORT performance usually will improve with a larger region size.

Reduce the Use of DB2 Sort: Using DB2 sort increases the load on the processor, virtual and real storage, and I/O devices. Whenever possible, use appropriate indexes rather than a sort to get needed information; this will improve response time and overall performance. Reducing the use of DB2 sort is particularly important for large tables.

DB2 sort is invoked when the following operations are performed on columns that are not indexed in the necessary order:

GROUP BY
ORDER BY
Join
DISTINCT
UNION (except UNION ALL)

Use Optimum Lock Size: Use LOCKSIZE TABLESPACE if minimum use of system resources and maximum locking speed is a higher priority than concurrent access to data. Keep in mind that LOCKSIZE TABLESPACE will allow only one application to update a table space during the time that the table space lock is held. You specify the duration of the lock with the ACQUIRE and RELEASE parameters on the BIND command.

Use LOCKSIZE PAGE if concurrent access to shared data is a higher priority than speed and use of system resources. LOCKSIZE PAGE will ensure concurrent access to a table space for all applications running at the same time, but the cost in system resources and the time required to do page locking is much higher than for table space locking.

Use LOCKSIZE ANY to balance concurrency versus speed and use of system resources, where neither factor is of overriding concern. With LOCKSIZE ANY, all applications begin with page locking protocol and continue to use page locking as long as the number of page locks concurrently held by the application in the table space stays below the install parameter LOCKS PER TABLE SPACE (IRLMLKTS). This assures full concurrency. However, if the LOCKS PER TABLE SPACE limit is reached, the locking protocol escalates to table space locking, and the remainder of the application runs under the table space lock. The LOCKS PER TABLE SPACE limit is the point at which it is more efficient in terms of system resources to give the application dedicated access to the table space than to continue page locking. For more information on specifying LOCKS PER TABLE SPACE, see the explanation of that parameter on page 195.

Using the Bind Process

This section discusses the bind considerations affecting application plan performance.

Minimize the Impact of Bind Validity Checking: The VALIDATE option of the bind (and rebind) subcommand specifies one of the following:

1. All static SQL statements must be fully validated by the bind process; or
2. Validity and/or authorization checking is to be performed at execution time for the SQL statements that (1) reference objects that don't exist at BIND time and (2) exercise privileges that have not been granted yet. This is the default.

Deferring the validity checking until execution time may impact application performance, depending on the number of statements flagged and how many times they are executed. Its use should be avoided as much as possible. Ensure all objects are created and all privileges are granted before bind, and select the VALIDATE(BIND) option.

Control Locks on Resources: The ACQUIRE and RELEASE parameters of the bind process enable you to control when table spaces cited in your application are to be locked and later released. You may specify any of three combinations of these parameters; each is explained in detail in *IBM DATABASE 2 Command and Utility Reference.* The option offering the greatest concurrent access of a table space, ACQUIRE(USE)/ RELEASE(COMMIT), is discussed here.

The ACQUIRE(USE)/RELEASE(COMMIT) specification allows DB2 to lock the resource only when needed and to release it as soon as possible. Thus, an application will not lock a needed resource longer than necessary. This is important if a program contains many SQL statements that are rarely executed because they will access data only in certain circumstances. Also, if you access the same table space repeatedly, ACQUIRE(USE)/RELEASE(COMMIT) is recommended.

However, this specification can increase the frequency of deadlocks. Deadlocks occur when two programs each have locked a resource the other needs, preventing either program from proceeding.

Ensure Optimal Path Selection: No option can control the path selection used in the bind process. However, some statistical information stored in the DB2 catalog is used as input to this function. This information (see Figure 79 on page 168) is updated by the RUNSTATS utility. In addition, the EXPLAIN statement and EXPLAIN options of BIND and REBIND tell you the methods DB2 used to access your data.

The following list describes how you can use RUNSTATS, rebinding, and EXPLAIN to help influence the path selection.

- Execute the RUNSTATS utility after loading a table space and before binding application plans that will access the table space. In addition, execute RUNSTATS periodically so you can compare the output of one execution to previous executions. This way, you can detect performance problems early.
- Rebind high-performance applications, and applications that are responsible for most of the accesses to a table space, after a table space they use is reorganized and the RUNSTATS utility is run. Also, rebind these types of applications after a RUNSTATS utility run if you believe the table space characteristics have changed significantly since the applications were last bound. This will enable the bind path

selection function to use the latest available information when selecting the access path.

- Use the SQL EXPLAIN statement and the EXPLAIN options of BIND and REBIND to find out the access and processing methods DB2 has chosen for an SQL statement. After reviewing this information, you may want to change the SQL statement, or the application imbedding it, to improve performance. In addition, if you notice that DB2 did not use an index to access the data, you may want to define an index to improve performance. For guidelines on when to create indexes, see "Establish Indexes" on page 186.

  Before using the SQL EXPLAIN statement or the EXPLAIN option of BIND and REBIND, you must create tables to hold the information produced. For more information, see *IBM DATABASE 2 Data Base Planning and Administration Guide* and *IBM DATABASE 2 SQL Reference*.

Determining the Size of Active Logs

The total capacity provided for the active log can affect DB2 performance significantly. Four parameters affect the capacity of the active log. In each case, increasing the value you specify for the parameter increases the capacity of the active log. The parameters are listed below:

- NUMBER OF LOGS (LOGSNUM) on the Log Data Sets Panel (DSNTIPL) controls the number of active log data sets you create.
- ARCHIVE LOG FREQ (NUMHRARC) on the Sizes Panel (DSNTIPD) provides an estimate of how often active logs data sets are to be copied to the archive log.
- UPDATE RATE (NUMCOMHR) on the Sizes Panel (DSNTIPD) estimates the number of data base changes (updates, inserts, deletes) expected per hour.
- CHECKPOINT FREQ (OPCHKFRQ) on the Operator Functions Panel (DSNTIPD) specifies the number of log records that DB2 will write between checkpoints.

The capacity you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might, during rollback and restart, need to access data in the archive log. This takes a considerable amount of time. When you are deciding how many active log data sets to use and how large each should be, consider that:

- *Performance is negatively affected when DB2 must access data on an archive log data set.* This can occur during rollback or restart. Access to archived information can be delayed a considerable length of time if a unit is unavailable, or if a volume mount is required (for example, a tape mount).

  During rollback, DB2 does not share access to an archive log data set with multiple units of recovery. In other words, one unit of recovery retrieves all records from an archive log data set before another unit of recovery is allowed to access that archive log data set. Data base locks, virtual storage, and other resources are not freed until the rollback completes. For restart, DB2 will not be available for new work until restart processing is complete.
- *At least one checkpoint is taken each time DB2 switches to a new active log data set. If the data sets are too small, checkpoints occur too frequently. As a result, data base writes will not perform efficiently.* You should provide enough active log space for 10 checkpoint intervals. For estimation purposes, you can assume that a

single checkpoint writes 24K bytes (or 6 control intervals) of data to the log. A checkpoint interval is defined by the number you specify for checkpoint frequency (see "Operator Functions Panel: DSNTIPO" on page 79). Make sure that the number you specify times 10 is less than the number of changes per hour times the number of hours per archive.

- *When selecting values for the parameters described above, avoid creating a few very large active log data sets.* This will happen if you specify a large number for ARCHIVE LOG FREQ (NUMHRARC) on the Sizes Panel (DSNTIPD), and a small number for NUMBER OF LOGS (LOGSNUM) on the Log Data Sets Panel (DSNTIPL). This often causes active log data set shortages.
- *Avoid creating many very small data sets.* This will happen if you specify a small number for ARCHIVE LOG FREQ and a large number for NUMBER OF LOGS. It will cause operational problems if tapes are used for archiving.
- *When archiving to tape, adjust the size of your active log data set so that it approximately matches the size of your tapes.* Because each active log data set is off-loaded to a different volume, a tape volume contains only one log data set. To avoid wasting tape media,

Certain processes cause a large amount of information to be logged, requiring a large amount of log space. Consider the following:

- The utility operations REORG and LOAD LOG(YES) cause all reorganized or loaded data to be logged. For example, if a table space contains 200 megabytes of data, this data, as well as control information, will be logged when this table space is the object of a REORG utility job.

  The utility operations issue intermediate sync-points based on the availability of buffers. This diminishes the amount of logging required for rollback and restart if the operation or DB2 itself should fail.

  As a guideline, specify LOAD LOG(YES) when adding a small percentage of records or issuing a REORG on a small table. This will create additional logging, but eliminate the need for a full image copy. When loading many records or issuing a REORG on large tables, specify LOAD LOG(NO) and take a full image copy immediately after the LOAD or REORG.
- The amount of logging performed for applications depends on how much data is changed. Certain SQL operators are quite powerful, making it easy to modify a large amount of data with a single statement. These statements include:
  - INSERT with a subselect from another table
  - DELETE all rows in a table

  Each of these operations result in the logging of all data base data that are changed. For example, if a table contains 200 megabytes of data, that data and control information will be logged if all of the rows are deleted in a table using the SQL DELETE operator. No intermediate sync-points are taken during this operation.

  When loading a large amount of data, use the LOAD utility rather than the SQL INSERT statement whenever possible. This allows you to control the logging performed for the utility. Because the utility takes intermediate sync-points, the demand for log information at rollback and restart is minimized.

Try to create one table per table space. Rather than deleting all rows or dropping the table itself, use the SQL DROP statement to drop the entire table space. This operation does not log the data involved.

Changing Catalog and Directory Size and Location

Consider using the procedure in this section if one of the scenarios below applies to your site:

- Your DB2 catalog and directory data sets have many secondary extents. Using this procedure, you can consolidate these extents into a data set with a single extent, thereby improving performance.
- You expect a significant increase in the use of DB2 objects. Using this procedure, you can allocate additional space for the DB2 catalog and directory, thereby preventing potential performance problems.
- Your DB2 catalog or directory data sets are not using a significant part of their space. Using this procedure, you can free some of that unused space.
- You want to move some DB2 catalog or directory data sets to another device for performance or space reasons.

To change the size or location of DB2 catalog or directory data sets for any one of the scenarios identified above, you must perform a data base recovery. A hierarchy of recovery dependencies determines the order in which you should try to recover data sets. This order is discussed in *IBM DATABASE 2 Operation and Recovery Guide*.

To perform the recovery, observe the following steps. These steps assume your changes will affect the DB2 directory (DSNDB01) and the DB2 catalog (DSNDB06). If your changes will affect only one of these data bases, modify the following steps accordingly.

1. Use the COPY utility to copy all the table spaces involved. This step is optional, but recommended. It will speed the recovery process and minimize the possibility of losing data.

   The table spaces in DSNDB01 are:

DBD01	SCT02
SYSUTIL	SYSLGRNG

   The table spaces in DSNDB06 are:

SYSCOPY	SYSGROUP	SYSUSER
SYSDBASE	SYSPAUT	SYSVIEWS
SYSDBAUT	SYSPLAN	

   For an example of using the COPY utility, see *IBM DATABASE 2 Command and Utility Reference*.
2. Issue the -STOP DATABASE (DSNDB01,DSNDB06) command.
3. Use the access method services DELETE command to delete each data set involved.

4. Use the access method services DEFINE CLUSTER command to redefine a data set for each table space involved. On the DEFINE CLUSTER command, specify the new size or location you want. For an example of using the DEFINE command, see *IBM DATABASE 2 Command and Utility Reference.*
5. Issue the -START DATABASE (DSNDB01,DSNDB06) ACCESS (UT) command to start the data bases for utility only access. SYSADM authority is required; if an authorization table space is involved, the Install SYSADM may be required to perform the work.
6. Run the RECOVER utility against the spaces involved.
7. Issue the -START DATABASE (DSNDB01,DSNDB06) ACCESS (RW) command to restart the data bases.

The catalog and directory should now have the sizes and locations you specified in step 4.

Tuning Strategy Summary

Figure 86 includes most of the tuning options discussed in this chapter. The table column headings identify different performance goals, such as improving speed and increasing concurrency. The rows at the left of the table identify techniques that can help you improve DB2 performance, such as using indexes, dropping unneeded objects, and invoking the RUNSTATS utility.

An X at the intersection of a column and a row indicates that using the technique listed at the left of the X will help you achieve the goal listed above the X. For example, beneath "Techniques" is an entry for "Use RUNSTATS"; the two Xs to the right of this entry show that RUNSTATS will help you improve DB2 speed and bind operations.

Technique	Speed	DASD Storage	Virtual Storage	Concurrent Access	Table Management	Optimize Bind
Use indexes	X				X	
Drop objects	X					
Use RUNSTATS	X					X
Reduce OPEN/CLOSE	X		X			
Increase free space	X					
Increase buffer pools	X		X			
Reduce authorization checking	X					
Control SYSLGRNG size	X					
Cross-memory operation	X					
Reduce overhead processing	X			X	X	X
Traces	X		X			

Figure 86 (Part 1 of 2). Summary of Tuning Techniques

Technique	Speed	DASD Storage	Virtual Storage	Concurrent Access	Table Management	Optimize Bind
Secondary allocation		X				
Change IRLM specifications			X			X
Reduce sorts			X		X	
Use commit/rollback statements	X					
REORG	X					
Increase subpages	X			X		
Use partitioning					X	
Use incremental image copy					X	
Use validate option				X		X
Use acquire/release options					X	
Table space locks	X					
Page locks				X		

Figure 86 (Part 2 of 2). Summary of Tuning Techniques

Chapter 9. Updating DB2 Install and Migration Parameters

This chapter describes how to modify the parameters you specified when installing or migrating DB2. This process allows you to tailor DB2 more precisely to your needs.

During install or migration, you entered values for numerous parameters either by using ISPF panels or by running the DSNTINST CLIST directly.

There are only two parameters that cannot be updated: the CATALOG ALIAS (VCATALOG) on the Data Parameters Panel (DSNTIPA2), and the associated DEFINE CATALOG (VCATSTAT). The CATALOG ALIAS parameter establishes an alias name for your ICF catalog. This name is also used as the high-level qualifier name for DB2 VSAM data sets. The DEFINE CATALOG parameter controls the creation of the ICF catalog. To change either of these parameters, you must reinstall DB2. If you do this, you will lose your ICF catalog and consequently all the data associated with your DB2 subsystem.

Aside from these two parameters, you can modify all the parameters you previously specified. Some of these parameters can be modified using a series of update panels that DB2 provides. These panels are similar to panels you used to install or migrate DB2. Other parameters can be modified using other means. This chapter discusses how to modify all parameters, whether you use the panels or another means.

Using the Update Panels

The update process does not generate a complete set of install or migration jobs. It generates only one job: DSNTIJUZ. This job assembles and link-edits the DB2 data-only initialization parameter module, DSNZPARM (or whatever you choose to call it on the Main Panel), and the application program's default module, DSNHDECP.

DSNZPARM contains the expansion of five macros; the parameters you specify during the update process are mapped to one of these macros. The names of the macros appear in Figure 87.

Macro Name	Content
DSN6ENV	Processing environment parameters: 370 or XA
DSN6SPRM	Data base parameters
DSN6ARVP	Parameters for archive log data set characteristics
DSN6LOGP	Log processing parameters used by the DB2 log manager
DSN6SYSP	Systems parameters for message routing, tracing, SMF statistics, and the number of log records written between checkpoints

Figure 87. DSNZPARM Macros

The DSNTINST CLIST performs calculations on some of the parameters you enter during the update process.

To invoke the ISPF update session:

1. Allocate the DB2 panel library and message library, concatenated to the ISPF panel library.
2. Invoke ISPF.
3. Select option 6 on the main ISPF menu.
4. Enter:

   ```
   EXEC 'DSN130.DSNCLIST(DSNTINST)'
   ```

Next, ISPF will display the Main Panel (DSNTIPA1). This panel allows you to select the different options available in an ISPF tailoring session.

Using the Main Panel For Update: DSNTIPA1

Figure 88 shows the Main Panel (DSNTIPA1). This is the same Main Panel you used for install and migration.

The Main Panel is the first and last panel you use during the update tailoring session. When you complete the session, this panel is displayed again. This panel uses ISPF panel services to remember certain panel values and re-displays them each time you run the DSNTINST CLIST. Upon successful completion of CLIST runs, the OUTPUT MEMBER NAME (OUTMEM) and DATA SET NAME PREFIX (OUTPUT) will not be re-displayed as you end your session and then re-invoke the CLIST. This will help you avoid accidentally overwriting your previous output.

This panel allows you to control the processing of the DSNTINST CLIST. Some of the values shown to the right of the arrows are the subsystem default values used by the CLIST; the others are examples of values you can enter.

```
DSNTIPA1           INSTALL, UPDATE, AND MIGRATE DB2 - MAIN PANEL
===>

Check parameters and reenter to change:

 1  INSTALL TYPE         ===> U
                                     I for new Install
                                     U for Update
                                     Data set name(member) to migrate
Enter name of your input data sets (DSNLOAD, DSNMACS, DSNSAMP, DSNCLIST):
 2  PREFIX               ===> DSN130
 3  SUFFIX               ===>

Enter to set or save panel values (by reading or writing the named members):
 4  INPUT MEMBER NAME    ===> DSNTIDXA  Enter to read old panel values
 5  OUTPUT MEMBER NAME   ===>           Enter to write new panel values

Enter name to create edited JCL and CLIST output:
 6  DATA SET NAME PREFIX ===>

 7  DSNZPARM NAME        ===> DSNZPARM  Enter name of new DB2 parameter module

PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 88. Installation Main Panel: DSNTIPA1

In the following descriptions of the parameter fields, the names in parentheses are the parameter names used within the CLIST.

1. INSTALL TYPE (INSTYPE)

Accept the default U, because you are doing an update process. For the update process, each panel to be used must be selected, displayed, and changed individually.

2. PREFIX (LIBPREFX)

Use the data set name prefix you used when you installed DB2.

3. SUFFIX (LIBSUFFX)

Use the data set name suffix you used when you installed DB2.

4. INPUT MEMBER NAME (INMEM)

You can use an output parameter member name that was created in a previous session.

5. OUTPUT MEMBER NAME (OUTMEM)

Enter the name you want for the output parameter member produced to save the values you change on subsequent panels during this update session.

6. DATASET NAME PREFIX (OUTPUT)

Specify the name you want to append to the output data sets produced by DSNTINST. This dataset prefix should not generate the name of a data set that already exists, or the data set will be overwritten.

7. DSNZPARM NAME (OUTZPRM)

If the output data set name prefix (OUTPUT) is also specified, a new DSNZPARM member will be created and the name entered for this line is used for the name of the initialization parameter module.

After you have made your entries on this panel, press the ENTER key to continue to the panels you choose to update.

For details about each of the parameters and how to use them for updating, see Chapter 4, "Install and Migration Parameters" on page 58.

Individual Update Menus Panel: DSNTIPB

Figure 89 shows the panel you will see after pressing the ENTER key on the Main Panel.

```
DSNTIPB          UPDATE DB2 - SELECTION MENU
===>

 Select one of the following:

 1  ARCHIVE LOG DATA SETS
 2  MAIN STORAGE SIZES (NUMBERS OF CONCURRENT USERS, BUFFER SIZES)
 3  APPLICATION PROGRAMMING DEFAULTS
 4  LOG DATA SETS
 5  LOCKING
 6  OPERATOR COMMUNICATION, TRACING, ACCOUNTING
 7  PROTECTION, PASSWORDS
 8  DATA BASES TO START AUTOMATICALLY









PRESS:  ENTER to select   END to exit   HELP for more information
```

Figure 89. Individual Update Menu Panel: DSNTIPB

On the command line, enter a number from 1 to 8 which corresponds to the panel you want to access:

1. Archive Log Data Sets Panel (DSNTIPA)
2. Storage Sizes Panel (DSNTIPE)
3. Application Programming Defaults Panel (DSNTIPF)
4. Log Data Sets Update Panel (DSNTIPG)
5. Locking Update Panel (DSNTIPK)
6. Operator Functions Panel (DSNTIPO)
7. Protection Update Panel (DSNTIPQ)
8. Data Bases and Spaces to Start Automatically Panel (DSNTIPS)

When the panel you selected is displayed, enter the new parameters and then press the ENTER key to return to the Update Selection Menu Panel (DSNTIPB). Press END to leave the Update Selection Menu Panel and return to the Main Panel (DSNTIPA1).

Three of the panels listed above are unique to the update process: DSNTIPG, DSNTIPK, and DSNTIPQ. These are discussed in that order in this chapter. For information about the other panels, see Chapter 4, "Install and Migration Parameters" on page 58.

Log Data Sets Update Panel: DSNTIPG

The DB2 log is a major factor that affects DB2 performance and data integrity. Figure 90 shows the panel you use to update the log parameters.

```
DSNTIPG          UPDATE DB2 - LOG DATA SETS
===>

Enter data below:

 1  NUMBER OF COPIES===> 2          1 or 2
 2  INPUT BUFFER    ===> 28K        Size in bytes (28K-60K)
 3  OUTPUT BUFFER   ===> 400K       Size in bytes (32K-4000K)
 4  WRITE THRESHOLD ===> 20         Number buffers filled before write (1-256)









PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 90. Log Data Sets Update Panel (DSNTIPG)

1. NUMBER OF COPIES (LOGSTWO)

Specify the number of copies of the active log: 2 or 1. If you accept the default, which is **2**, you invoke dual logging. DB2 then maintains two copies of the active log. We strongly recommend that you use dual logging at all times to protect subsystem data integrity.

If you change the number of log copies, you must run the CHANGE LOG INVENTORY utility. If you add a second copy of the log, you must also define the data sets. See *IBM DATABASE 2 Operation and Recovery Guide* for information about how to do this.

2. INPUT BUFFER (LOGINPUT)

Specify the size of buffers used during recovery operations. Use a value from 28K to 60K. The default is 28K for both MVS/370 and MVS/XA.

The buffers used for reading active or archive logs during recovery can affect the performance of the subsystem. The buffers should be compatible with the size of the log records. Each time a read operation is started, a buffer of the specified size is allocated.

3. OUTPUT BUFFER (LOGOUTPT)

Specify the amount of storage used for buffering log operations. Use a value from 32K to 40,000K. The default is 400K for MVS/XA and 200K for MVS/370.

4. WRITE THRESHOLD (LOGTHRSH)

Specify the number of buffers to be filled before starting a write. Use a value from 1 to 256. The default is 10 for MVS/370 and 20 for MVS/XA.

Find a balance between performance and data integrity when setting this parameter. A large value reduces the number of writes per minute, thereby increasing performance. A small value increases the number of writes per minute, thereby increasing the integrity of the active log data.

Locking Update Panel: DSNTIPK

```
DSNTIPK          UPDATE DB2 - LOCKING
===>

Enter data below:

 1  LOCKS PER TABLE SPACE ===> 1000       Maximum before lock escalation
 2  LOCKS PER USER        ===> 10000      Maximum before resource unavailable









PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 91. Installation Locking Update Panel: DSNTIPK

1. LOCKS PER TABLE SPACE (IRLMLKTS)

Sets parameter NUMLKTS in macro DSN6SPRM, which specifies the maximum number of page locks that may be held concurrently by a single application against a single table space for which LOCKSIZE ANY has been specified. This maximum includes locks against any data pages and index pages (or subpages) that are acquired on behalf of the application when it accesses the table space.

To understand how this parameter functions, you should understand how DB2 selects locks when LOCKSIZE ANY is specified on a CREATE TABLESPACE statement.

DB2 makes the decision at bind time as to what locking level will be used at run time. (For dynamic SQL, bind is done at run time as the first step of execution.)

When a BIND is issued for a table space for which you have specified LOCKSIZE ANY, DB2 usually selects page locking. However, DB2 selects table space locking if page locking is inefficient to fulfill the requested isolation level for the data base operation to be performed.

As soon as DB2 selects page locking for a table space, in response to a specification of LOCKSIZE ANY, it establishes and maintains a count of the number of page locks concurrently held by the application against the table space. When and if that count reaches the value you specified for LOCKS PER TABLE SPACE, DB2 escalates the page lock to a table space lock. It stops acquiring page locks, and it releases all page

locks that are currently held by the application. Any remaining operations for the application will be performed under the table space lock.

The escalation only pertains to the application (and table space) that reached the specified limit.

The minimum value you can specify for LOCKS PER TABLE SPACE is 0. The maximum for MVS/370 is 25000; the maximum for MVS/XA is 50000.

If you specify 0 for this parameter, you disable lock escalation. In this case, when a BIND is issued against a table space for which LOCKSIZE ANY has been specified, DB2 will almost always select a table space lock.

The default value for this parameter is 1000 for MVS/XA and 200 for MVS/370. If, however, you specified NO for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), we recommend that you change this value to consider the available lock space. For MVS/370, you may need to reduce this value to 50. For MVS/XA, you may be able to accept the default value if adequate ECSA space is available.

2. LOCKS PER USER (IRLMLKUS)

This parameter sets the parameter NUMLKUS in macro DSN6SPRM, which specifies the maximum number of page locks that may be held concurrently by a single application against all table spaces in the subsystem. This maximum includes locks on data pages and index pages (or subpages) that are acquired when the application accesses table spaces. This limit applies to all table spaces for which page locking is in effect. That is, LOCKSIZE PAGE or LOCKSIZE ANY was specified.

As soon as execution begins for an application, DB2 establishes and maintains a count of the total number of page locks concurrently held by the application across all table spaces. When and if that count reaches the value you specify for LOCKS PER USER, DB2 issues a "resource unavailable" condition ("00C90096") and an SQL return code -904. DB2 does not process the SQL request that encountered the limit.

If the application is in a must-complete or must-abort state, requests for page locks are granted, even though they exceed the specified limit.

The minimum value you can specify for LOCKS PER USER is 0. The maximum for MVS/370 is 25000; the maximum for MVS/XA is 100000. DB2 assumes that 200 bytes of storage are required for each lock.

If you specify 0 for this parameter, you disable the user lock limit function.

The default value for this parameter is 10000 for MVS/XA and 2000 for MVS/370. If, however, you specified NO for CROSS MEMORY (IRLMPC) on IRLM Panel 2 (DSNTIPJ), we recommend that you change this value to consider the available lock space. For MVS/370, you may need to reduce this value to 500. For MVS/XA, you may be able to accept the default value if adequate ECSA space is available.

Protection Update Panel: DSNTIPQ

Security needs change often for the system administrator. The integrity of the subsystem depends heavily on the parameters that affect security. You should alter the parameters as your needs change. The Protection Update Panel, shown in Figure 92, is used for updating DB2 protection information.

```
DSNTIPQ          UPDATE DB2 - PROTECTION
===>

Enter data below:

 1  ARCHIVE LOG PW     ===> DBADMIN    Archive log data sets password
 2  ARCHIVE LOG RACF   ===> NO         RACF protect archive log data sets
 3  USE PROTECTION     ===> YES        DB2 authorization enabled.  YES or NO
 4  SYSTEM ADMIN 1     ===> SYSADM     Authid of system administrator
 5  SYSTEM ADMIN 2     ===> SYSADM     Authid of system administrator
 6  SYSTEM OPERATOR 1  ===> SYSOPR     Authid of system operator
 7  SYSTEM OPERATOR 2  ===> SYSOPR     Authid of system operator
 8  UNKNOWN AUTHID     ===> IBMUSER    Authid of default (unknown) user









PRESS:  ENTER to continue   END to exit   HELP for more information
```

Figure 92. Installation Protection Update Panel: DSNTIPQ

1. ARCHIVE LOG PW (PROTARCH)

Choose the password you need to define and access your archive log data sets. New and old archive data sets will be processed with this password.

If you change the password, you should also use MVS utility functions to change the password in the system PASSWORD data set; otherwise, operations that try to access old hardware log data sets may fail MVS authorization checking.

The CHANGE LOG INVENTORY utility must be used to remove this password protection. See *IBM DATABASE 2 Command and Utility Reference* for information on the CHANGE LOG INVENTORY utility.

2. ARCHIVE LOG RACF (PROTARAC)

Determine whether or not you want your archive log data sets to be RACF protected. (RACF provides additional security beyond the normal password security measures.) The default is NO.

This parameter is ignored by RACF if the DB2 user ID has the ADSP (automatic data set protection) attribute specified in the RACF user ID profile. If ADSP is not specified, this parameter specifies whether to protect archive log data sets.

For further information on RACF, see *Resource Access Control Facility (RACF) General Information Manual.*

3. USE PROTECTION (PROTAUTH)

Determine whether the authorization mechanism is enabled. In an emergency, you may need to disable authorization. The system administrator (SYSADM) has the authorization to access and correct the errors.

4. SYSTEM ADMIN 1 (PROTADMN)

This parameter establishes the authorization ID for the system administrator. The system administrator has authority over the entire DB2 subsystem. This authorization ID is one of two that will be allowed to recover certain subsystem tables. The default is SYSADM.

Further work is required to update this ID. You must also update the VSAM ownerid.

5. SYSTEM ADMIN 2 (PROTADM2)

This parameter establishes the authorization ID for the second Install SYSADM. The default is SYSADM.

Further work is required to update this ID. You must also update the VSAM ownerid.

6. SYSTEM OPERATOR 1 (PROTOPR1)

This parameter establishes the authorization ID for the first Install SYSOPR. The default is SYSOPR.

7. SYSTEM OPERATOR 2 (PROTOPR2)

This parameter establishes the authorization ID for the second Install SYSOPR. The default is SYSOPR. You must accept the default SYSOPR for either the first or the second system operator.

8. UNKNOWN AUTHID (PROTUNKN)

The default authorization ID can be changed by altering this value. This ID will be used when the connection verification function does not have a user ID from RACF, the JOB statement, and so forth. Change this value if your security needs for data set authorization change. Null is not a valid value.

Note: You will probably want the names in parameters 4 through 7 above to be existing TSO, IMS/VS or CICS users. You may also want SYSTEM ADMIN and SYSTEM ADMIN2 to be existing RACF users.

Updating the DSNTINST CLIST Parameters

This section discusses how you can change the value of DSNTINST CLIST parameters. It contains a brief description of each parameter and explains how the parameter can be changed.

The CLIST parameters are displayed in Figure 93, which consists of five columns:

- The first column, "ISPF Panel," identifies the suffix of the ISPF panel that contains the parameter. All panels are named DSNTIP*x*, where *x* is a 1- or 2-character suffix.
- The second column, "Panel Parameter," contains the descriptive information to the left of the panel entry fields.
- The third column, "Description," contains a short definition of the parameter or description of its use.
- The fourth column,"CLIST Parameter," contains the parameter names used within the DSNTINST CLIST. These are the names that are in parentheses in the parameter descriptions earlier in this book. The input and output parameter members also contain these names. (Figure 93 is alphabetically ordered on the names in this column.)
- The fifth column,"Notes," contains either a "Y," an "N," or a number between 1 and 12. "Y" means the parameter can be updated dynamically, using the CLIST support. If there is no "Y" in this column, it means there is no explicit support for changing the parameter. "N" means the parameter cannot be changed using CLIST. A number in this column corresponds to one of the notes explained at the end of Figure 93.

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPA	CATALOG DATA	Catalog archive log data sets: YES or NO	ARCHCTLG	Y
DSNTIPA	DEVICE TYPE	Archive log device type	ARCHDEVT	Y
DSNTIPA	MAX CONCUR-RENT	Maximum archive log input volumes	ARCHINPT	Y
DSNTIPA	RECORDING MAX	Maximum archive log data sets recorded	ARCHMAXV	Y
DSNTIPA	MSVGP1	Mass storage group for 1st copy of archive logs	ARCHMSGP	Y
DSNTIPA	MSVGP2	Mass storage group for 2nd copy of archive logs	ARCHMSG2	Y
DSNTIPA	ARCHIVE PREFIX	Prefix for 1st copy of archive logs	ARCHPRE1	Y
DSNTIPA	COPY 2 PREFIX	Prefix for 2nd copy of archive logs	ARCHPRE2	Y

Figure 93 (Part 1 of 6). CLIST Parameters

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPA	BLOCK SIZE	Archive log blocksize	ARCHSIZE	Y
DSNTIPA	NUMBER OF COPIES	Number of copies of archive log data sets per active log copy	ARCHTWO	Y
DSNTIPA	WTOR ROUTE CODE	Routing codes for WTOR message	ARCHWRTC	Y
DSNTIPA	WRITE TO OPER	Write to operator before archive mount: YES or NO	ARCHWTOR	Y
DSNTIPA	RETENTION PERIOD	Archive log retention period	ARCRETN	Y
DSNTIPL	BOOTSTRAP NAME 1	Name of 1st copy of BSDS	BSDSNAM1	1
DSNTIPL	BOOTSTRAP NAME 2	Name of 2nd copy of BSDS	BSDSNAM2	1
DSNTIPE	MAX BP0 BUFFERS	Maximum size of buffer pool BP0	BUFMAX00	Y
DSNTIPE	MAX BP1 BUFFERS	Maximum size of buffer pool BP1	BUFMAX01	Y
DSNTIPE	MAX BP2 BUFFERS	Maximum size of buffer pool BP2	BUFMAX02	Y
DSNTIPE	MAX BP32K BUFFERS	Maximum size of buffer pool BP32K	BUFMAX32	Y
DSNTIPE	MIN BPO BUFFERS	Minimum size of buffer pool BP0	BUFMIN00	Y
DSNTIPE	MIN BP1 BUFFERS	Minimum size of buffer pool BP1	BUFMIN01	Y
DSNTIPE	MIN BP2 BUFFERS	Minimum size of buffer pool BP2	BUFMIN02	Y
DSNTIPE	MIN BP32K BUFFERS	Minimum size of buffer pool BP32K	BUFMIN32	Y
-	CLIST tracing: N, L, C or S		CONTROL	Y
DSNTIPS	START OR DEFER	RESTART or DEFER named data bases	DBSTADFR	Y
DSNTIPS	START NAMES	Data bases to start automatically	DBSTARTx	Y
DSNTIPF	MINIMUM DIVIDE SCALE	Minimum number of digits displayed to the right of the decimal point.	DECDIV3	Y
DSNTIPF	DECIMAL POINT IS	Decimal point: period or comma	DECPOINT	11
DSNTIPF	CHARACTER SET	Character set: Alphameric or Katakana	DEFCHARS	11
DSNTIPF	LANGUAGE DEFAULT	Default host language	DEFLANG	Y
DSNTIPF	MIXED DATA	Allow double byte character set: YES or NO	DEFMIXED	11

Figure 93 (Part 2 of 6). CLIST Parameters

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPF	SQL STRING DELIMITER	SQL delimiter: apostrophe or quotation mark	DEFSQSTR	11
DSNTIPF	STRING DELIMITER	String delimiter: apostrophe or quotation mark	DEFSTRNG	Y
DSNTIPF	DATE FORMAT	Format used to represent dates	DEFDATE	12
DSNTIPF	TIME FORMAT	Format used to represent time	DEFTIME	12
DSNTIPF	LOCAL DATE LENGTH	Length for user-defined date format	DEFDATEL	12
DSNTIPF	LOCAL TIME LENGTH	Length for user-defined time format	DEFTIMEL	12
DSNTIPA1	INPUT MEMBER NAME	Data set containing parameter defaults	INMEM	Y
DSNTIPA1	INSTALL TYPE	Install type: I, U, or Migrate	INSTYPE	Y
DSNTIPI	AUTO START	Autostart of IRLM: YES or NO	IRLMAUTO	Y
DSNTIPI	TIME INTERVAL	Interval between lock processing	IRLMCYCL	8
DSNTIPJ	DEADLOCK CYCLE	Number of local deadlock cycles	IRLMDEDC	8
DSNTIPJ	DEADLOCK TIME	Seconds per deadlock cycle	IRLMDEDT	8
DSNTIPJ	IDENTIFIER	Identification number of the IRLM	IRLMIDEN	8
DSNTIPI	INSTALL IRLM	Install the IRLM procedure: YES or NO	IRLMINST	-
DSNTIPJ, DSNTIPK	LOCKS PER TABLE SPACE	Maximum locks per table space	IRLMLKTS	Y
DSNTIPJ, DSNTIPK	LOCKS PER USER	Maximum locks per user	IRLMLKUS	Y
DSNTIPJ	MAXIMUM CSA or ECSA	Maximum CSA or ECSA use by IRLM	IRLMMCSA	8
DSNTIPJ	PARTNER	Partner for this IRLM	IRLMPART	8
DSNTIPJ	CROSS MEMORY	Use cross memory program call: YES or NO	IRLMPC	8
DSNTIPJ	PROC NAME	Name of the IRLM procedure	IRLMPROC	9
DSNTIPI	SUBSYSTEM NAME	Name of the IRLM subsystem	IRLMSSID	9
DSNTIPI	START TIMEOUT	IRLM wait time in seconds; time DB2 will wait for IRLM to autostart	IRLMSTTO	9
DSNTIPJ	INTERNAL TRACE	IRLM automatic trace start	IRLMTRAC	8
DSNTIPI	WAIT TIME	Maximum IRLM timeout in seconds to wait for unavailable resource	IRLMWAIT	9

Figure 93 (Part 3 of 6). CLIST Parameters

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPA1	PREFIX	Prefix for library data set	LIBPREFX	2
DSNTIPA1	SUFFIX	Suffix for library data set	LIBSUFFX	2
DSNTIPL, DSNTIPG	INPUT BUFFER	Log input buffer size	LOGINPUT	Y
DSNTIPL, DSNTIPG	OUTPUT BUFFER	Log output buffer size	LOGOUTPT	Y
DSNTIPL	NUMBER OF LOGS	Number of active log data sets	LOGSNUM	3
DSNTIPL	DATA SET PREFIX 1	Prefix for 1st copy of active logs	LOGSPRE1	3
DSNTIPL	DATA SET PREFIX 2	Prefix for 2nd copy of active logs	LOGSPRE2	3
DSNTIPL, DSNTIPG	NUMBER OF COPIES	Number of active log copies: 1 or 2	LOGSTWO	Y
DSNTIPL, DSNTIPG	WRITE THRESHOLD	Log buffers to fill before write	LOGTHRSH	Y
DSNTIPM	AUTH MEMBER	MVS authorization member name	MVSAMEMB	4
DSNTIPM	AUTH SEQUENCE	MVS authorization sequence number	MVSASEQN	4
DSNTIPM	LINK LIST ENTRY	MVS link list suffix LNKLST	MVSLMEMB	4
DSNTIPM	LINK LIST SEQUENCE	MVS link list sequence number	MVSLSEQN	4
DSNTIPM	SUBSYSTEM MEMBER	Subsystem parmlib suffix IEFSSN	MVSSMEMB	4
DSNTIPM	SUBSYSTEM NAME	MVS subsystem name (to start DB2)	MVSSNAME	4
DSNTIPM	SUBSYSTEM PREFIX	MVS subsystem command prefix character	MVSSPREF	4
DSNTIPM	SUBSYSTEM SEQUENCE	MVS subsystem definition sequence	MVSSSEQN	4
DSNTIPD	COLUMNS	Average number of columns per table	NUMCOLUM	5
DSNTIPD	UPDATE RATE	Average changes per hour	NUMCOMHR	5
DSNTIPE	MAX BATCH CONNECT	Maximum concurrent batch users of DB2	NUMCONBT	Y
DSNTIPE	MAX USERS	Maximum concurrent DB2 users	NUMCONCR	Y
DSNTIPE	DATA BASES	Maximum concurrently open data bases	NUMCONDB	Y
DSNTIPE	MAX TSO CONNECT	Maximum concurrent TSO fore-ground users	NUMCONTS	Y

Figure 93 (Part 4 of 6). CLIST Parameters

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPD	DATA BASES	Maximum number of data bases	NUMDATAB	5
DSNTIPD	ARCHIVE LOG FREQ	Number of hours between log archiving	NUMHRARC	5
DSNTIPD	PLANS	Maximum plans at site	NUMPLANS	5
DSNTIPD	EXECUTED STMTS	Average number of SQL statements executed	NUMSTMTE	5
DSNTIPD	TABLES IN STMT	Average number of tables in SQL statements	NUMSTMTL	5
DSNTIPD	PLAN STATE-MENTS	Average number of SQL statements per plan	NUMSTMTS	5
DSNTIPD	TABLES	Average number of tables per data base	NUMTABLE	5
DSNTIPD	TABLE SPACES	Average number table spaces per data base	NUMTABSP	5
DSNTIPD	TEMPORARY 4K	Space for temporary table space 1	NUMTEMP1	10
DSNTIPD	TEMPORARY 32K	Space for temporary table space 2	NUMTEMP2	10
DSNTIPD	VIEWS	Average number of views per table	NUMVIEWS	5
DSNTIPO	CHECKPOINT FREQ	Checkpoint frequency	OPCHKFRQ	Y
DSNTIPO	WAIT FOR RECALL	Open data set processing to wait for automatic recall	OPRECALL	Y
DSNTIPO	MVS LEVEL	MVS level: 370 or XA	OPMVSLV	Y
DSNTIPO	WTO ROUTE CODES	Routing codes for unsolicited mes-sages	OPROUTCD	Y
DSNTIPO	SMF ACCOUNTING	Send accounting data to SMF: YES or NO	OPSMFACT	Y
DSNTIPO	SMF STATISTICS	Send statistics data to SMF: YES or NO	OPSMFSTA	Y
DSNTIPD	STATISTICS TIME	Time interval between statistics col-lection	OPSTATIM	Y
DSNTIPO	TRACE AUTO START	Automatic trace start: YES or NO	OPTRCAUT	Y
DSNTIPO	TRACE SIZE	Size of trace table	OPTRCSIZ	Y
DSNTIPA1	OUTPUT MEMBER NAME	Data set member to contain CLIST output	OUTMEM	Y
DSNTIPA1	DATA SET NAME PREFIX	Prefix for output data sets	OUTPUT	Y

Figure 93 (Part 5 of 6). CLIST Parameters

ISPF Panel	Panel Parameter	Description	CLIST Parameter	Notes
DSNTIPA1	DSNZPARM NAME	Member name for DSNZPARM	OUTZPRM	Y
DSNTIPP	SYSTEM ADMIN 1	1st Install SYSADM ID	PROTADMN	Y
DSNTIPP	SYSTEM ADMIN 2	2nd Install SYSADM ID	PROTADM2	Y
DSNTIPP	ARCHIVE LOG RACF	RACF protect archive log: YES or NO	PROTARAC	Y
DSNTIPP	ARCHIVE LOG PW	Password for archive log data sets	PROTARCH	Y
DSNTIPP	USE PROTECTION	Enable DB2 authorization: YES or NO	PROTAUTH	Y
DSNTIPP	BSDS PASSWORD	Password for BSDS	PROTBSDS	6
DSNTIPP	DIRECTORY/ CATALOG	Password for directory and catalog	PROTDIRC	6
DSNTIPP	LOG PASSWORD	Password for active log data sets	PROTLOGS	6
DSNTIPP	SYSTEM OPER-ATOR 1	1st Install SYSOPR ID	PROTOPR1	Y
DSNTIPP	SYSTEM OPER-ATOR 2	2nd Install SYSOPR ID	PROTOPR2	Y
DSNTIPP	UNKNOWN AUTHID	Unknown (default) authorization ID	PROTUNKN	Y
DSNTIPP	ICF CATALOG	VSAM password for VCATALOG	PROTVCAT	Y
-	-	Input install data on ISPF panels	SPF(YES)	Y
DSNTIPA2	CATALOG ALIAS	ICF catalog alias name	VCATALOG	N
DSNTIPA2	DEFINE CATALOG	Define the catalog: YES or NO	VCATSTAT	N
DSNTIPA2	VOLUME SERIAL 1	1st volume to hold data sets	VOLSDAT1	7
DSNTIPA2	VOLUME SERIAL 2	2nd volume to hold data sets	VOLSDAT2	7
DSNTIPA2	VOLUME SERIAL 3	3rd volume to hold data sets	VOLSDAT3	7
DSNTIPA2	VOLUME SERIAL 4	4th volume to hold data sets	VOLSDAT4	7
DSNTIPA2	VOLUME SERIAL 5	5th volume to hold data sets	VOLSDAT5	7
DSNTIPA2	VOLUME SERIAL 6	6th volume to hold data sets	VOLSDAT6	7
DSNTIPA2	PERMANENT UNIT NAME	Device type for ICF catalog and PDS	VOLSDEVT	7
DSNTIPA2	TEMPORARY UNIT NAME	Device type for temporary data sets	VOLTDEVT	7

Figure 93 (Part 6 of 6). CLIST Parameters

Notes to Figure 93:

1. The BSDS is accessed via JCL on the start procedure. The data set can be moved or expanded using access method services EXPORT and IMPORT.
2. Changing the library data set names requires editing a large number of data sets. Using the CLIST for an install can help produce the changes.

3. The log data sets may be accessed via stand alone access macros or modified via the CHANGE LOG INVENTORY utility. Access method services EXPORT and IMPORT may also be used. See *IBM DATABASE 2 Operation and Recovery Guide* for more information.
4. MVS modifications require research and editing. Study the updates made by job DSNTIJMV.
5. Changing the space for the DB2 directory and catalog data sets requires a data base recovery scenario. Copies of the table spaces are taken. The data bases or table spaces are stopped. The data sets are deleted and redefined using VSAM commands. Then the spaces are started for utility access only and the RECOVER utility is run. Finally, start the data bases or table spaces for normal use. See "Changing Catalog and Directory Size and Location" on page 201, and also *IBM DATABASE 2 Operation and Recovery Guide* for more information.
6. These passwords can be modified using the access method services ALTER command. The CHANGE LOG INVENTORY utility can then be run to tell DB2 the new passwords. The BSDS passwords must be entered by the operator each time that DB2 is started.
7. Changing volumes involves Notes 1, 3, and 5.
8. Changing these parameters requires editing the IRLM start procedure.
9. Changing these parameters requires changing the IRLM start procedure and/or changing the associated parameter in the parameter update job, DSNTIJUZ. The parameter update must be executed and DB2 started with the new parameters.
10. Changing these parameters can be done by deleting the data sets and redefining them either when DB2 is down or when the temporary data base (DSNDB07) is stopped. Job DSNTIJTM may provide a useful example.
11. These parameters can be updated. However, the update will modify SQL syntax. Investigate the impact of an update carefully.
12. These parameters can affect the way your applications run. Update these with caution.

Example of Invoking the CLIST Without ISPF

To invoke the CLIST, and **not** the display panels, use the TSO EXEC command and pass the parameter values in the invoking statement. This is called running in linear (as opposed to panel) mode. You can do this under TSO—in foreground or batch. It is also possible to type the parameter values directly into the INMEM member and then run the CLIST without passing the parameters explicitly. For information on how to find a description of a parameter, see Chapter 4, "Install and Migration Parameters" on page 58. Use the following instructions to update the parameters.

You must specify:

```
INSTYPE (U)
SPF (NO)
OUTPUT (dsname.prefix)
```

Following is an example of the TSO EXEC command:

```
EXEC 'DSN130.DSNCLIST(DSNTINST)' 'SPF(NO) INSTYPE(U) -
OUTPUT(DSN130.MODS)                                  -
INMEM(DSNTID04) OUTMEM(DSNTID05)                     -
LOGTHRSH(8) OPCHKFRQ(3000)                           -
PROTADMN(DBADM) ARCHMAXV(700)'
```

This example updates the CLIST directly with the following specifications:

- Output prefix character string DSN130.MODS
- Input member DSNTID04
- Output member DSNTID05
- Maximum of 8 buffers filled before a write
- 3000 records written between checkpoints
- System administrator authorization for DBADM
- 700 archive logs data sets recorded in the BSDS before a wrap

Appendixes

Appendix A. Initialization Parameter Module (DSNZPARM)

This appendix describes each of the five macros within the DB2 initialization parameter module, DSNZPARM. These macros contain the DB2 execution-time options you selected during the ISPF tailoring session.

DSNZPARM is generated by job DSNTIJUZ each time you install, migrate, or update DB2. For a description of DSNTIJUZ, see *IBM DATABASE 2 Install Guide.*

Reading the Syntax Diagrams

Each macro description contains a set of syntax diagrams. This section explains how to read these diagrams.

Syntax Conventions

You follow the syntax of the macro from left to right.

The ▶▶— symbol indicates the beginning of a macro.

The —▶◀ symbol indicates the end of a macro.

The —▶ symbol indicates that the macro syntax is continued.

The ▶— symbol indicates that the macro syntax resumes after a continuation.

Keep following the syntax diagrams until you reach the —▶◀ symbol (the end of the macro).

Words in uppercase letters (LIKE THIS) must be entered exactly as shown. Words in lowercase letters (like this) indicate that you must substitute a word for the lowercase word.

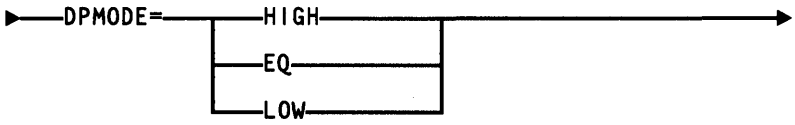
If a comma (,), equal sign (=), or set of parentheses (()) is shown with either upper- or lowercase words, it must be entered as part of the macro.

Required Parameters

A required parameter will appear on the same horizontal line (or continuation) as the macro:

Choice of Parameters

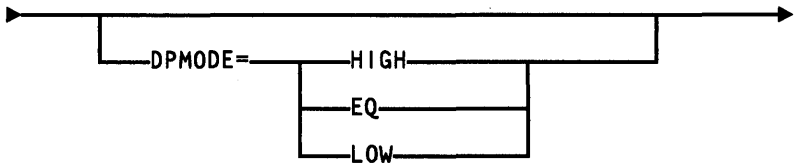
When you have a choice of parameters, the parameters appear in a vertical list with one of the parameters on the main path, like this:



The above example indicates that you must code one of the parameters.

Optional Parameters

If a macro has an optional parameter, the parameter will appear below the main path, like this:



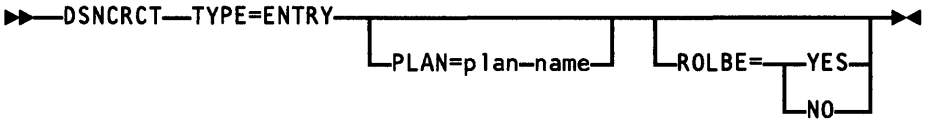
If you do not want to specify the parameter, you may bypass it by following the main path above it. Your coded command would be:

DSNCRCT TYPE=FINAL

If you want to specify the parameter, you follow the line that drops below the main path, contains the parameter, and rises back up to the main path. Your code would be:

DSNCRCT TYPE=FINAL DPMODE=HIGH

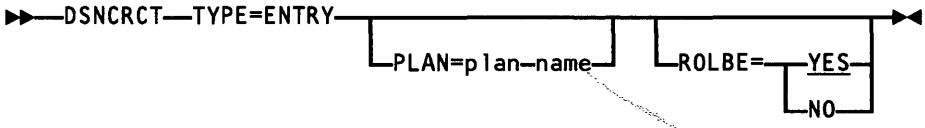
If you have a choice of several optional parameters, the parameters will appear in a vertical list below the main path.



You may specify either of the parameters or bypass them completely by following the main path above the parameter list.

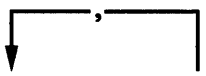
Default Parameters

Wherever possible, default parameters are underlined. In the following example, the default for ROLBE is *YES*.



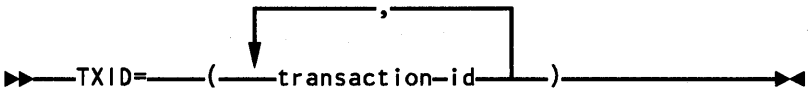
More Than One Parameter

The repeat symbol:



indicates that more than one parameter (or more than one value of a parameter) may be coded. The comma means you must separate with a comma those parameters that you specify.

If the repeat symbol extends to the borders of a word, as in the word "transaction-id" in the following example, the symbol applies to that word only. That is, you can code as many transaction-ids as you like, separating them with commas.



Because "transaction-id" is in lowercase letters, you replace this word with the actual transaction-id(s) you want to code. Your coded command might look like this:

TXID=(ACCT,ACCW,ACCU)

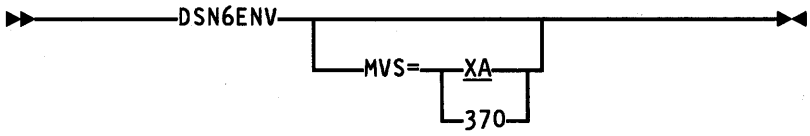
or perhaps

TXID=(ACCT)

Subsystem Environment Macro: DSN6ENV

The DSN6ENV macro specifies the environment in which DB2 will operate.

Note: This macro must be the first macro in any assembly of DB2 system parameters.



Description

DSN6ENV
This is the name of the macro; it must be coded exactly as shown, and it must be separated from any parameters by one or more blanks.

MVS
Specifies the environment in which DB2 must run.

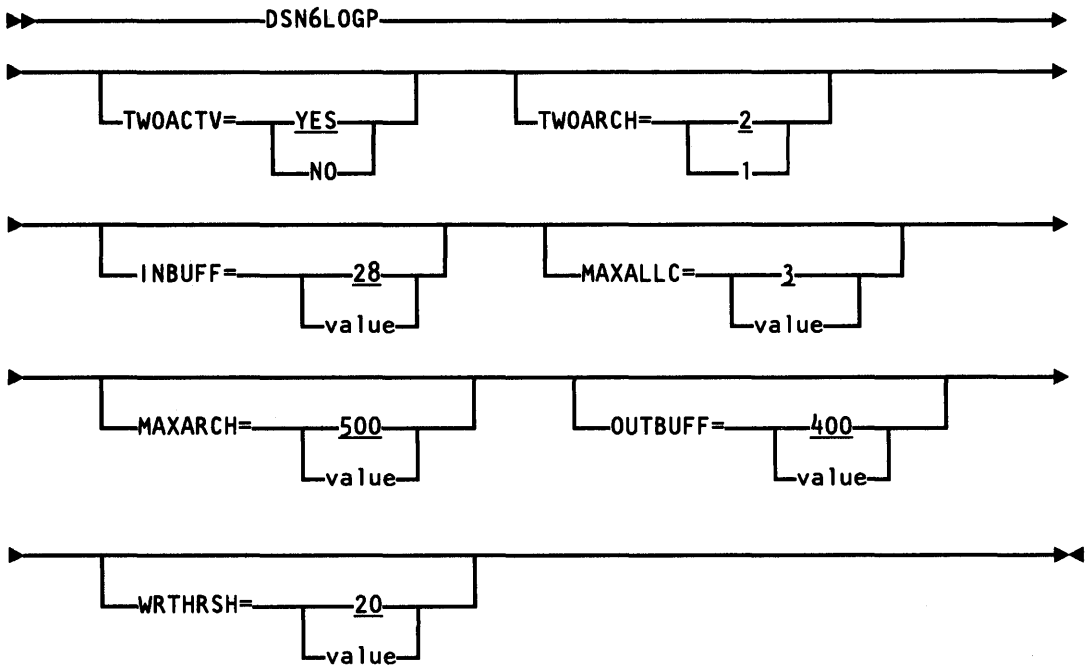
XA specifies an extended architecture environment.

370 specifies a standard environment (not extended architecture).

XA is the default.

Log Processing Options Macro: DSN6LOGP

The DSN6LOGP macro specifies log processing options to be used by the DB2 log manager.



Notes:

1. The macro name should be followed by one or more blanks before optional parameters are coded.
2. Optional parameters must be separated by commas (with no blanks).

Description

DSN6LOGP
This is the name of the macro. It must be coded exactly as it appears here, and it must be separated from any following optional parameters by one or more spaces.

TWOACTV
Specifies whether two copies of the active log are to be maintained (2 or 1).

2 is the default.

TWOARCH
Specifies whether off-loading will generate two copies of the archive log (YES or NO).

YES is the default.

INBUFF
Specifies the log input buffer pool size in kilobytes. For *value*, substitute a decimal value from 28 to 60. If the value you specify is not a multiple of 4, it

will be rounded up to the next multiple of 4. A separate buffer pool exists for each open archive log data set.

28 is the default.

MAXALLC
Specifies the maximum number of archive log input volumes to allocate at one time. This limits the number of devices and the amount of storage in use by the log manager for input processing. For *value*, substitute a decimal value from 1 to 99.

3 is the default.

MAXARCH
Specifies the maximum number of archive log volumes whose identifications will be recorded in the bootstrap data set. For *value*, substitute a decimal value from 10 to 1000.

500 is the default.

If two archive log data sets are maintained, the value you specify here is applied to each copy. That is, if the limit is set at 500, 500 copy-1 data sets and 500 copy-2 data sets will be recorded in the bootstrap data set.

When the number of archive data sets created exceeds the number specified by this option, the names of the oldest archive data sets will be deleted from the bootstrap data set.

OUTBUFF
Specifies the size of the output log buffer pool in kilobytes. There is one output buffer pool for each DB2 subsystem. Specify a decimal value from 32 to 4000. If the value you specify is not a multiple of 4, it will be rounded up to the next multiple of 4.

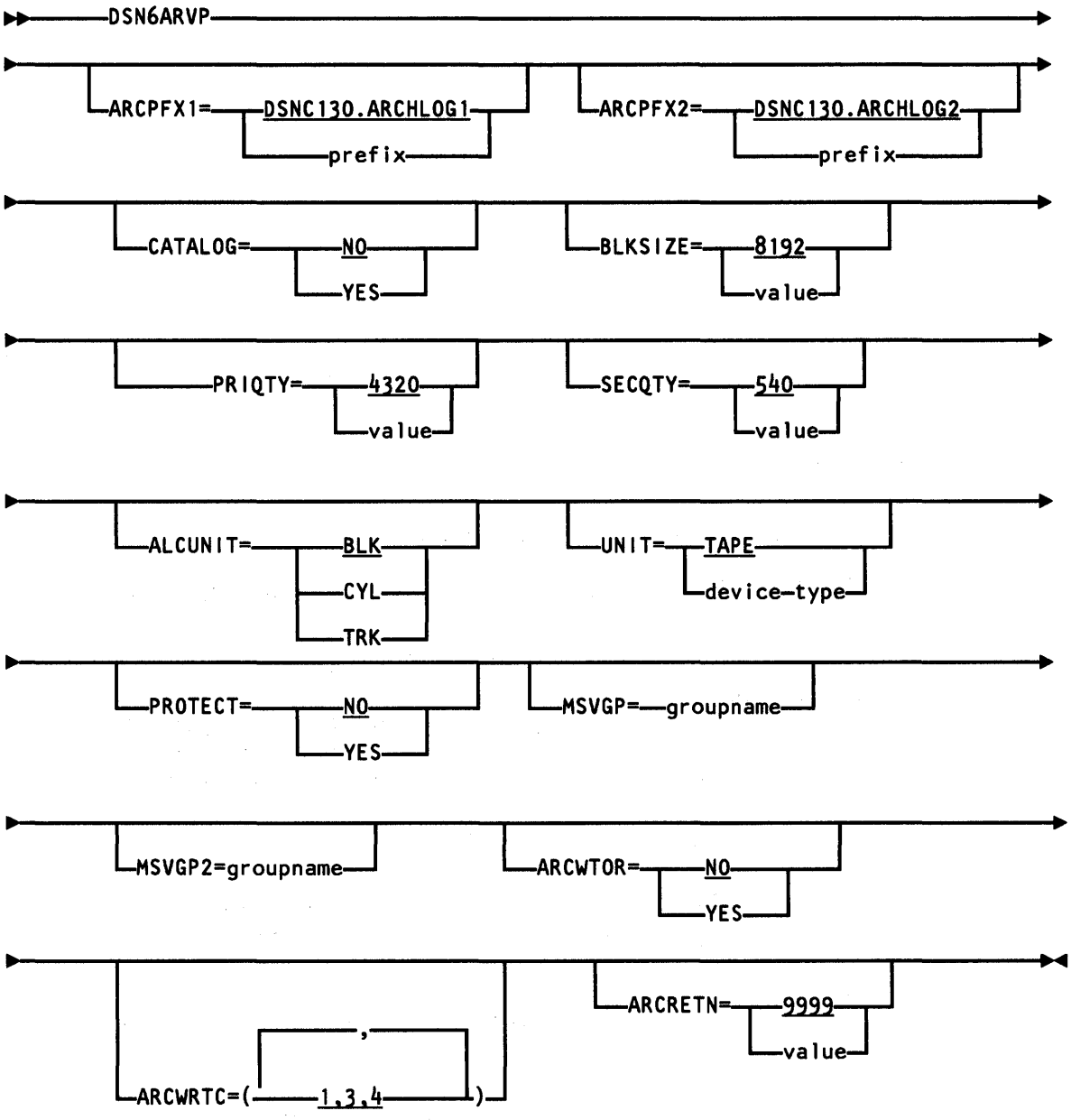
The default is 400.

WRTHRSH
Specifies the number of log output buffers to be filled to cause a physical write. Specify a decimal value from 1 to 256.

The default is 20.

Log Archive Characteristics Macro: DSN6ARVP

The DSN6ARVP macro specifies characteristics of each log archive data set.



Notes:

1. The macro name should be followed by one or more blanks before optional parameters are coded.
2. Optional parameters must be separated by commas (with no blanks).

Description

DSN6ARVP
This is the name of the macro. It must be coded exactly as it appears here, and it must be separated from any following optional parameters by one or more spaces.

ARCPFX1
Specifies a prefix to be used for copy-1 archive log data sets.

The data set name is a multi-level name. It consists of this prefix plus a last-level name formed from an alphabetic character and a 7-digit sequence number that is assigned by the system when the active log is off-loaded to an archive log data set. For *prefix*, substitute a string of 1 to 35 characters.

DSNC130.ARCHLOG1 is the default.

ARCPFX2
Specifies a prefix to be used for copy-2 archive log data sets.

This prefix will be used (as described above for ARCPFX1) to form names for copy-2 archive log data sets. For *prefix*, substitute a string of 1 to 35 characters.

DSNC130.ARCHLOG2 is the default.

CATALOG
NO specifies that DB2 is not to catalog archive log data sets.

YES specifies that DB2 is to catalog all archive log data sets when space for them is initially allocated. A flag will be set in the bootstrap data set, indicating that the data set was cataloged, and subsequent allocations will be made using the catalog.

NO is the default.

BLKSIZE
Specifies the data set block size. For *value*, substitute a decimal value from 8192 to 28672. If the value you specify is not a multiple of 4096, it will be rounded up to the next multiple of 4096.

8192 is the default.

PRIQTY
Specifies the primary space allocation (cylinders, tracks, or blocks, depending on the specification of the ALCUNIT parameter) for a DASD data set. For *value*, substitute a decimal value.

4320 is the default.

SECQTY
Specifies the secondary space allocation (cylinders, tracks, or blocks, depending on the specification of the ALCUNIT parameter) for a DASD data set. For *value*, substitute a decimal value.

540 is the default.

ALCUNIT
Specifies the allocation unit in which primary and secondary space allocations are obtained.

BLK is the default.

UNIT
Specifies the device type to be allocated when creating a new archive log data set. For *device-type*, substitute a valid specification of a tape or DASD unit. TAPE is the default. (To use this default, a site must have specified TAPE as a valid generic name at the time the system was generated.)

If DASD units are specified, only storage-class volumes will be used.

PROTECT
Specifies whether the archive log data sets will be RACF protected at initial allocation time (NO or YES).

NO is the default.

If you specify YES to password protect the archive log data sets, MVS will prompt the operator for the password whenever it is necessary for DB2 to read the data set. RACF class TAPEVOL must be active if your archive log goes to tape. If RACF class TAPEVOL is not active, and you are archiving to tape, and YES is specified for this parameter, archive will fail.

MSVGP
For *groupname* specify the name of the first mass storage volume group to be used for archive log data sets. The default is null.

MSVGP2
For *groupname* specify the name of the second mass storage volume group to be used for archive log data sets. The default is null.

ARCWTOR
Specifies whether a WTOR (write to operator with reply) message is to be issued (YES or NO) before allocation of an archive log data set.

NO is the default.

ARCWRTC
Specifies the routing codes for ARCWTOR. Enter from 1 to 14 routing codes.

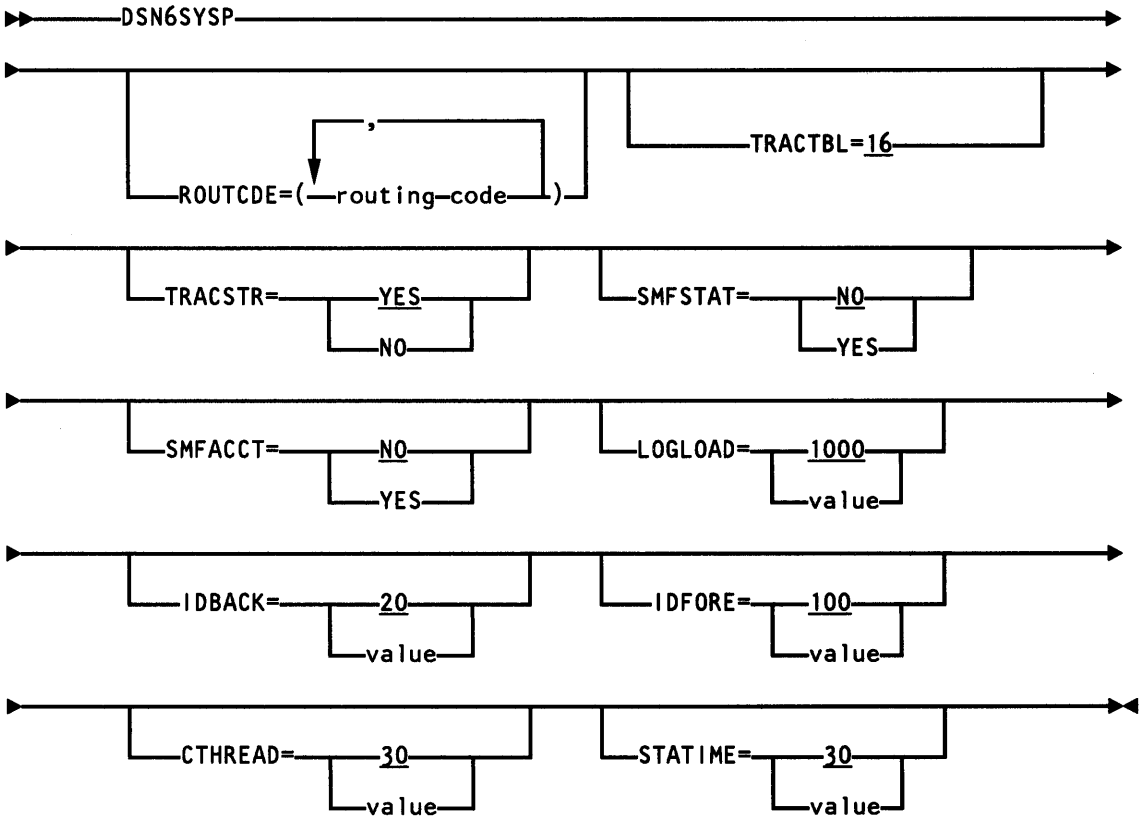
1,3,4 is the default.

ARCRETN
Specifies the number of days DB2 retains archive log data sets. For *value*, substitute a decimal value from 0 to 9999.

9999 is the default.

System Parameters Macro: DSN6SYSP

The DSN6SYSP macro is an initialization macro that specifies several system parameters.



Notes:

1. The macro name should be followed by one or more blanks before optional parameters are coded.
2. Optional parameters must be separated by commas (with no blanks).

Description

DSN6SYSP
This is the name of the macro. It must be coded exactly as it appears here, and must be separated from any following optional parameters by one or more spaces.

ROUTCDE
Specifies the routing code(s) to be assigned to system messages. For *routing-code*, substitute one or more routing codes. A routing code is a decimal value. If you specify more than one code, separate them by commas, and enclose them in parentheses.

If you do not specify routing codes (using this macro), system messages will be routed using a default routing code of 1 (meaning "master console action").

Note: For a description of valid routing codes, refer to the description of the WTO (write to operator) macro instruction in *MVS/Extended Architecture Message Library: Routing and Descriptor Codes* or *MVS/Extended Architecture System Programming Library: Supervisor Services and Macro Instructions.*

TRACTBL
Specifies the size of the trace table as a multiple of 4K bytes. Specify a decimal value from 1 to 99. 16 is the default, defining a 64K-byte trace table.

Approximately 100 events are recorded in every 4K-byte segment of the trace table.

TRACSTR
Specifies whether the resident trace table is to be started automatically during subsystem initialization (YES or NO). If you specify NO, the trace is suppressed, but it can be started with an operator command at any time following initialization.

YES is the default.

SMFSTAT
Specifies whether statistical data is gathered and written to the System Management Facility (YES or NO).

NO is the default.

SMFACCT
Specifies whether accounting data is gathered and written to System Management Facilities (YES or NO).

NO is the default.

LOGLOAD
Specifies a decimal value that represents the number of log records to be written to the DB2 log between the start of one checkpoint and the start of another. For *value*, substitute a decimal number between 200 and 500000.

1000 is the default (that is, 1000 records are to be written between checkpoints).

IDBACK
Specifies the maximum number of concurrent connections from batch jobs and utilities. Specify a decimal number between 1 and 100. Attempted connections in excess of the number you specify will fail.

The default is 20.

IDFORE
Specifies the maximum number of concurrent connections from TSO foreground (DB2I or SPUFI). Specify a decimal number between 1 and 1000. Attempted connections in excess of the number you specify number will fail.

The default is 100.

CTHREAD
Specifies the maximum number of concurrent threads for DB2. This includes threads for IMS/VS, CICS, TSO (foreground and batch) and utilities. Specify a decimal number between 1 and 200. Attempts to create threads in excess of the number you specify will be queued.

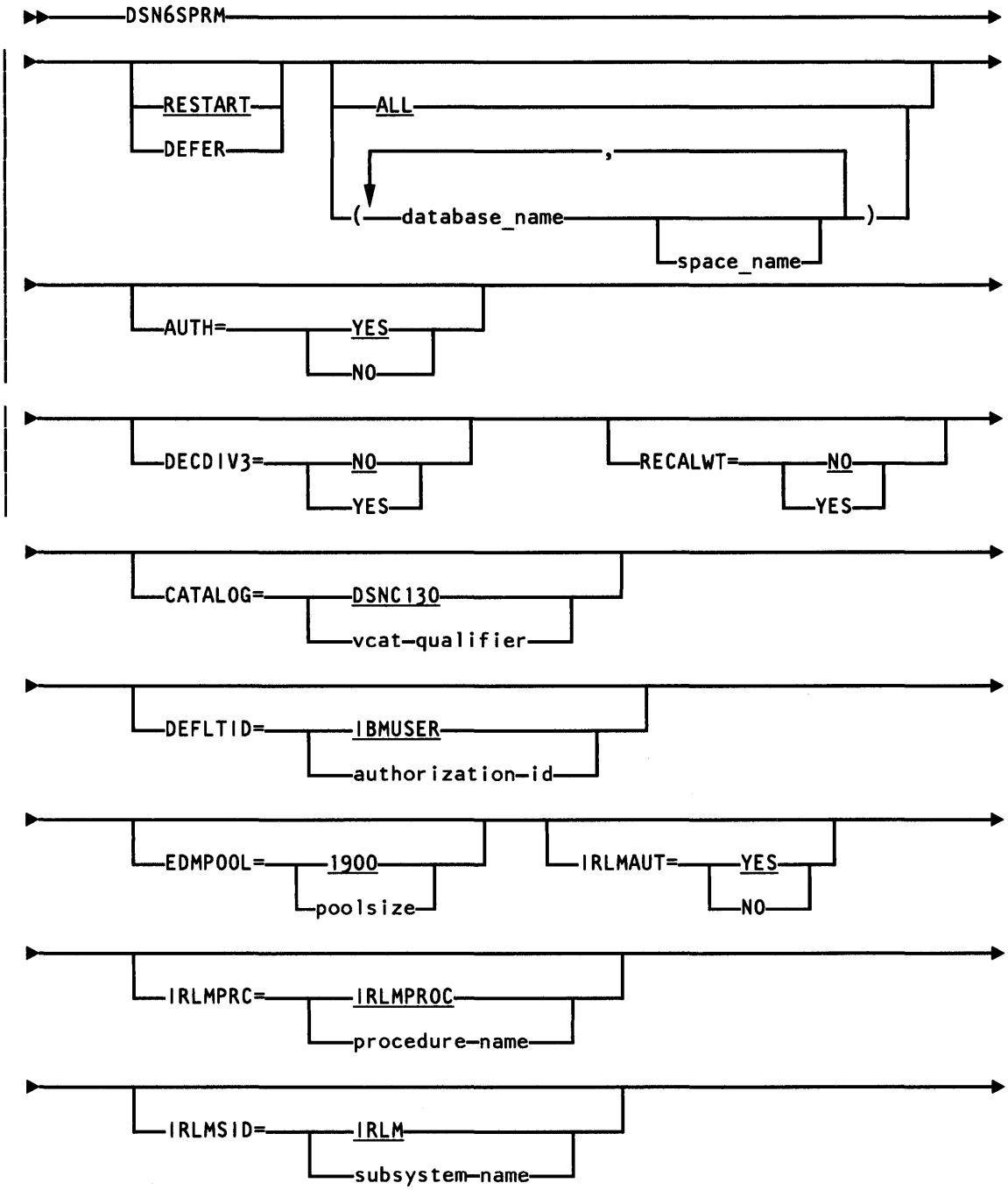
The default is 30.

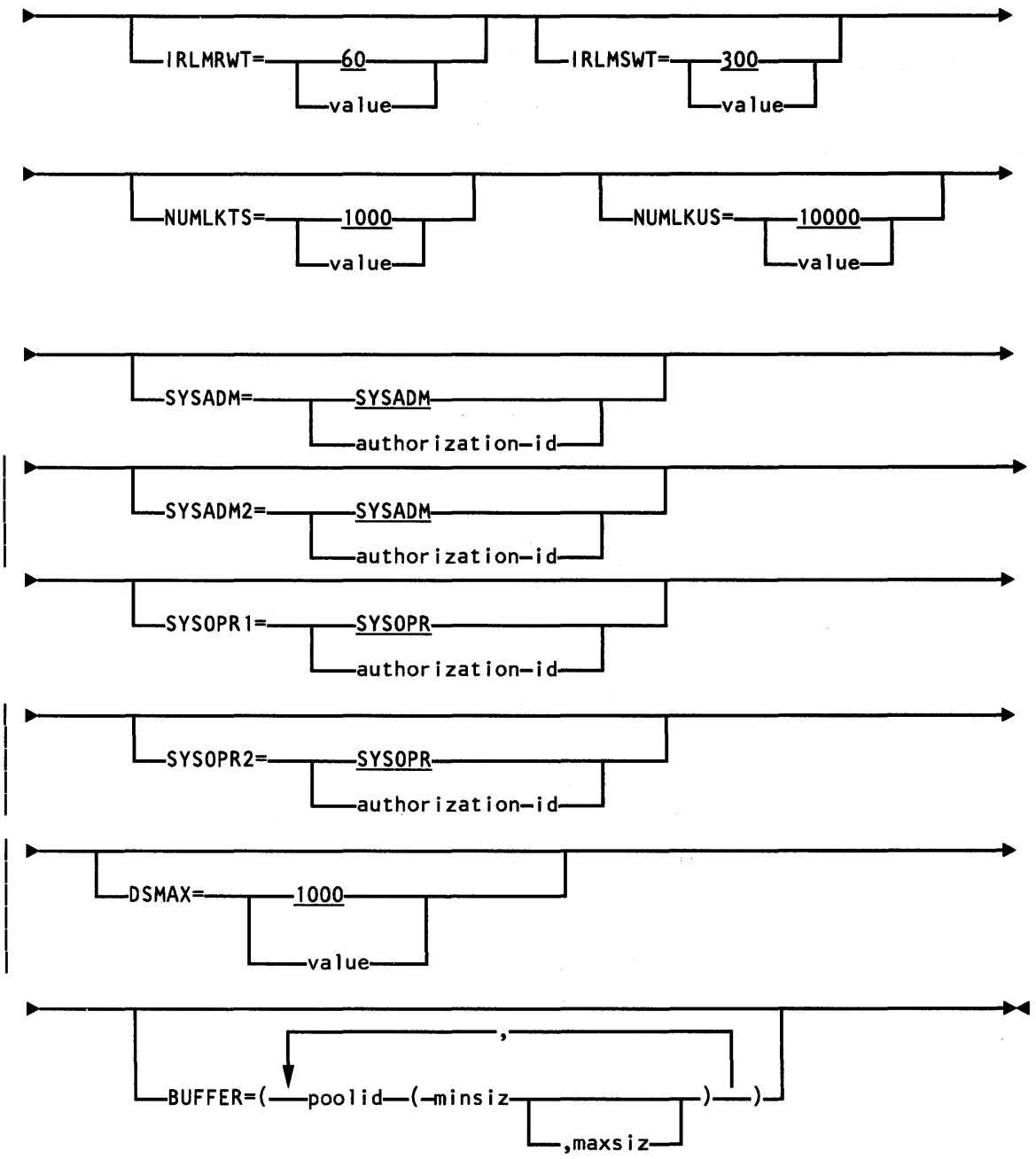
STATIME
Specifies the time interval, in minutes, for statistics records to be produced. For *value*, substitute a decimal value from 1 to 1440.

30 is the default.

Data Base Parameters Macro: DSN6SPRM

The DSN6SPRM macro specifies data base parameters.





Notes:

1. The macro name should be followed by one or more blanks before parameters are coded.
2. Parameters must be separated by commas (with no blanks).

Description

DSN6SPRM
This is the name of the macro. It must be coded exactly as it appears here, and must be separated from any following optional parameters by one or more spaces.

RESTART/DEFER
Specifies whether DB2 is to perform automatic restart processing for the listed objects when DB2 is started (RESTART or DEFER).

RESTART is the default.

After RESTART or DEFER, indicate the objects you want restarted or deferred by specifying one of the following:

- ALL to have DB2 automatically restart or defer all objects requiring restart processing.

  ALL is the default. DEFER,ALL includes the DB2 system data bases DSNDB01, DSNDB04, DSNDB06 and DSNDB07. For information about the restart process and the implications of deferring objects at restart time, see *IBM DATABASE 2 Operation and Recovery Guide*.

- Data base name(s) in the form:

  ```
  database_name
  ```

  to have DB2 automatically restart or defer all spaces in one or more data base(s).

- Table space or index space name(s) in the form:

  ```
  database_name.space_name,database_name.space_name,.....
  ```

  to have DB2 automatically restart or defer one or more table or index spaces.

AUTH
Specifies whether the DB2 authorization mechanism is to be activated (YES or NO).

YES is the default.

DECDIV3
Specifies whether at least three digits should be displayed to the right of the decimal point after decimal division (YES or NO). Certain accounting applications may choose this option. If the default NO is accepted, the rules for decimal division in SQL are followed. See *IBM DATABASE 2 SQL Reference* for more information about decimal division in SQL.

RECALWT
Specifies whether to wait for automatic recall (using DFHSM) to be performed for migrated data sets at data set open time.

NO indicates that if a DB2 table or index space has been migrated, DB2 will initiate the recall and indicate resource unavailable to the user. This is the default.

YES indicates that DB2 will wait for DFHSM to perform the automatic recall at data set open time.

If you specify YES for this parameter and a data set is being recalled, all subsequent data set open and close operations must wait until the recall of the migrated data set is complete. Since recall may take several seconds, this could have a significant effect on system response time.

Never specify YES for this parameter if data sets are migrated to tape, as the recall would require waiting for the tape to be mounted.

Note that the DB2 directory, catalog, and associated indexes should not be placed on volumes that are managed by DFHSM.

CATALOG
Specifies an ICF catalog high-level qualifier identifying all DB2 system data sets defined for system data bases (DSNDB01, DSNDB06, and DSNDB07). For *vcat-qualifier*, substitute a character string of up to 8 characters.

DSNC130 is the default.

DEFLTID
Specifies a default authorization ID that will be used whenever an authorization ID cannot be determined by the system. For *authorization-id*, substitute a character string of up to 8 characters. Null is not a valid value for this parameter.

IBMUSER is the default.

EDMPOOL
Specifies an Environmental Descriptor Manager (EDM) pool size. The Environmental Descriptor Manager (EDM) pool size is automatically calculated and expressed in kilobytes. "Main Storage Size" on page 50 contains a discussion on how DB2 calculates the EDMPOOL size.

For the complete calculation of the EDM pool size, use the DSNTINST CLIST.

1900K is the default.

IRLMAUT
Specifies whether DB2 will automatically start the IMS/VS Resource Lock Manager (YES or NO).

YES is the default.

IRLMPRC
Specifies the IMS/VS Resource Lock Manager procedure name. For *procedure-name*, substitute a character string of up to 8 characters.

IRLMPROC is the default.

IRLMSID
Specifies the name of the IMS/VS Resource Lock Manager subsystem. For *subsystem-name*, substitute a character string of up to 8 characters.

IRLM is the default.

IRLMRWT
Specifies the maximum IRLM timeout in seconds. This is the time that DB2 will wait for a locked resource. The maximum value is 3600.

60 is the default.

IRLMSWT
Specifies the IRLM wait time in seconds. This is the time that DB2 will wait for the autostart of the IRLM. The maximum value is 3600.

300 is the default.

NUMLKTS
Specifies the maximum number of page locks that an application can hold against any single table space for which LOCKSIZE ANY has been specified before DB2 escalates the locking level to table space. The minimum value for this parameter is 0, which will deactivate the lock escalation function.

1000 is the default.

NUMLKUS
Specifies the maximum number of page locks that an application can hold in total against all table spaces in the subsystem. If this limit is reached, DB2 returns "resource unavailable" ("00C90096") and does not process the SQL request. The minimum value for this parameter is 0, which deactivates the function.

10000 is the default.

SYSADM
Specifies the authorization ID of the initial system administrator. For *authorization-id*, substitute a character string of up to 8 characters.

SYSADM is the default.

SYSADM2
Specifies the authorization ID of the second system administrator.

The default is SYSADM.

SYSOPR1
Specifies the authorization ID of the initial system operator. For *authorization-id*, substitute a character string of up to 8 characters.

SYSOPR is the default. To avoid serious system problems, you must accept the default value SYSOPR for either SYSOPR1 or SYSOPR2.

SYSOPR2
Specifies the authorization ID of the second system operator. For *authorization-id*, substitute a character string of up to 8 characters.

The default is SYSOPR. To avoid serious system problems, you must accept the default value SYSOPR for either SYSOPR1 or SYSOPR2.

DSMAX
Specifies the maximum number of concurrently accessible data sets. Each concurrently accessible data set requires approximately 2.1K bytes of main storage.

1000 is the default.

BUFFER

The BUFFER parameter specifies the maximum and minimum sizes of the subsystem's buffer pools.

poolid

Specifies the buffer pool to which the "minsiz" and "maxsiz" values apply. This parameter can have the following values:

Value	Meaning
BP0	4K buffer pool 0
BP1	4K buffer pool 1
BP2	4K buffer pool 2
BP32K	32K buffer pool

minsiz

Specifies the minimum number of pages to be allocated to the buffer pool indicated by the "poolid" parameter. The value specified must be an integer and may not exceed the value of the "maxsiz" parameter. If 0 is specified, the indicated buffer pool is not allocated. If the BUFFER parameter is coded, "minsiz" is required; if the BUFFER parameter is not coded, the default values by buffer pool are:

poolid	Default MVS/XA	Default MVS/370
BP0	224	56
BP1	0	0
BP2	0	0
BP32K	12	3

maxsiz

Specifies the maximum number of pages to be allocated to the indicated buffer pool. The value specified must be an integer and may not be less than the "minsiz" parameter. If the "maxsiz" parameter is not specified, the default values by buffer pool are:

poolid	Default MVS/XA	Default MVS/370
BP0	224	56
BP1	0	0
BP2	0	0
BP32K	12	3

Appendix B. RACF Examples for DB2

This appendix assumes the use of RACF generic profiles rather than discrete profiles. It shows examples of:

- Accommodating DB2 started tasks with RACF. For more information, see:
  System Programming Library: Resource Access Control Facility (RACF)
  Resource Access Control Facility (RACF) Security Administrator's Guide
- A RACF router table
- Authorizing DB2 and DB2 users with RACF

You may also find helpful information about RACF in the following books:

Resource Access Control Facility (RACF) Auditor's Guide
Resource Access Control Facility (RACF) User's Guide

Accommodating DB2 Started Tasks with RACF

Figure 94 shows a job that will set up RACF Version 1 Release 7 to accommodate DB2 started tasks. This example shows a site with two DB2 subsystems. The subsystem IDs are DSN and SSTR. The comments in the example explain what each command does. You can change this example to meet your site's needs.

```
//ICHRIN03 JOB 1,'STC USERID TABLE'
//*
//*   REASSEMBLE MODULE ICHRIN03 TO SPECIFY THE RACF USERIDS TO
//*   BE ASSOCIATED WITH DB2 STARTED PROCEDURES.  THE TABLE HAS ROOM
//*   FOR EXPANSION IF NEW PROCEDURES ARE ADDED FOR ADDITIONAL
//*   SUBSYSTEMS.  THE COUNT FIELD MUST BE SUPERZAPPED AS WELL
//*   AS SUPERZAPPING IN THE NAME(S) OF THE ADDED PROCEDURES.
//*
//*   AN IPL WITH A CLPA (OR AN MLPA SPECIFYING THIS LOAD MODULE)
//*   IS REQUIRED FOR THESE CHANGES TO TAKE EFFECT.
//*
//*
//ASM     EXEC PGM=IEV90,PARM='NOOBJECT,DECK'
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//SYSPUNCH DD DSN=&&OBJ,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(2,1))
//SYSIN DD *
*
*    STARTED PROCEDURE ID TABLE.  THE PROCEDURES 'DSNMSTR'
*    AND 'DSNDBM1' ARE ADDED BY THE DB2 INSTALL PROCESS FOR THE
*    SUBSYSTEM ID DSN.
*
ICHRIN03 CSECT
COUNT    DC    AL2(((ENDTABLE-BGNTABLE)/$ENTLEN)+32768) NUM ENTRIES
BGNTABLE DC    CL8'DSNMSTR',CL8'SYSDSP',CL8' ',X'00',XL7'00'  DB2 DSN
$ENTLEN  EQU   *-BGNTABLE
         DC    CL8'DSNDBM1',CL8'SYSDSP',CL8' ',X'00',XL7'00'  DB2 DSN
*
         DC    CL8'SSTRMSTR',CL8'SYSDSP',CL8' ',X'00',XL7'00'  DB2 SSTR
         DC    CL8'SSTRDBM1',CL8'SYSDSP',CL8' ',X'00',XL7'00'  DB2 SSTR
*
         DC    CL8'IMS13',CL8'IMS',CL8' ',X'00',XL7'00'  IMSVS CNTL
         DC    CL8'CICS17',CL8'CICS',CL8' ',X'00',XL7'00'  CICS
ENDTABLE DS    0D
         DC    8CL($ENTLEN)' ' SPARE ENTRY - REQUIRES ZAP OF COUNT
         END
//LKED    EXEC PGM=IEWL,PARM='XREF,LIST,LET,RENT'
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DSN=SYS1.LPALIB,DISP=SHR
//SYSLMOD  DD DSN=SYS1.LPALIB,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//OBJ      DD DSN=&&OBJ,DISP=(OLD,DELETE)
//SYSLIN   DD *
 INCLUDE OBJ
 NAME ICHRIN03(R)
//
```

Figure 94. Sample Version of RACF Module ICHRIN03

The example above shows how the RACF module ICHRIN03 can be used to associate a started task address space (for example, DSNMSTR and DSNDBM1) with a user name (for example, SYSDSP).

If the started task could be run as a job, this is equivalent to specifying USER=SYSDSP on the JOB statement of the jobs DSNMSTR and DSNDBM1. The example above also shows how to to associate a userid of *IMS* with the IMS/VS control region, and a userid of *CICS* for CICS. If you were running this task as a job, you could specify USER=IMS or USER=CICS, or you could identify them in the ICHRIN03 module.

Sample RACF Router Table

Figure 95 shows how the RACF module ICHRFR01 can be tailored to include DB2. The comments in the example explain what each command does. You can change this example to meet your site's needs.

```
//ICHRFR01 JOB 1,'RACF ROUTER TABLE'
//ASM     EXEC PGM=IEV90,PARM='NOOBJECT,DECK'
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//SYSPUNCH DD DSN=&&OBJ,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(2,1))
//SYSIN    DD *
ICHRFR01 CSECT
*********************************************************************
*                                                                   *
*  FOLLOWING ENTRIES FOR DSNR ADDED : RACF INVOCATION SUPPORT IN DB2 *
*                                                                   *
*********************************************************************
         ICHRFRTB CLASS=DSNR,REQSTOR=IDENTIFY,SUBSYS=DSN,             *
               ACTION=RACF
         ICHRFRTB CLASS=DSNR,REQSTOR=IDENTIFY,SUBSYS=SSTR,            *
               ACTION=RACF
         ICHRFRTB TYPE=END
         END
//LKED    EXEC PGM=IEWL,PARM='XREF,LIST,LET,RENT'
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DSN=SYS1.LPALIB,DISP=SHR
//SYSLIB   DD DSN=SYS1.LPALIB(ICHRFR01),DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//OBJ      DD DSN=&&OBJ,DISP=(OLD,DELETE)
//SYSLIN   DD *
 INCLUDE OBJ
/*
```

Figure 95. RACF Module ICHRFR01 Tailored to Include DB2

Figure 95 shows that subsystems DSN and SSTR in class DSNR with requestor IDENTIFY will have their requests passed to RACF (ACTION=RACF) for connection checking.

Class DSNR is included in the class descriptor table (CDT). Although classes in the CDT are represented in the router, you may need to alter the router this way because you change the DB2 subsystem name (to SSTR, for example) during DB2 tailoring.

Authorizing DB2 and DB2 Users with RACF

Figure 96 describes a sample environment, and the RACF commands used in defining this environment are listed.

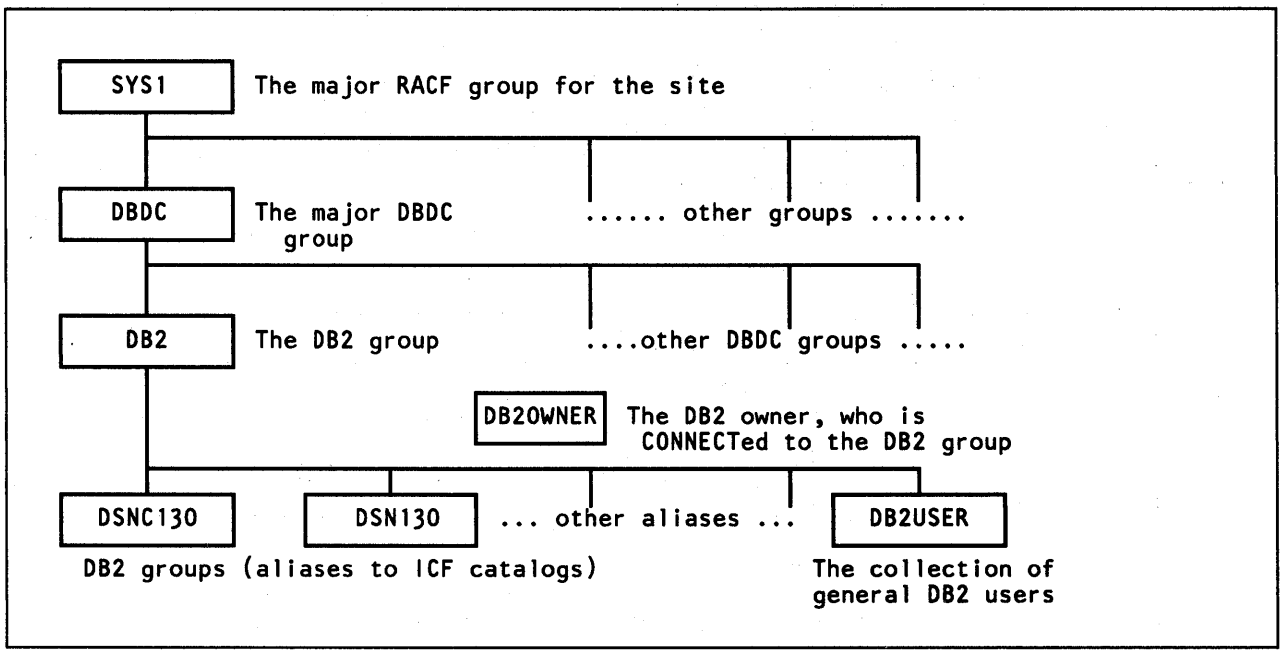


Figure 96. Sample DB2/RACF Environment

Some users and user groups are defined in Figure 97 on page 247.

The following steps define the groups, users, and data sets that are to be protected and controlled. Connections and permissions are made to suit the needs of the sample environment. Assume that the DBDC group already exists.

1. Activate the DSNR class, add DB2OWNER, and add the DB2 group.

   ```
   SETROPTS CLASSACT(DSNR)
   ```

   This activates the DB2 class. The asterisk (*) can be used instead of DSNR to activate all classes.

   The SETROPTS must be done by a user with the SPECIAL attribute. This is generally guarded very carefully in RACF environments, so it is wise to ask this user to do only this first step.

   ```
   ADDUSER  DB2OWNER  CLAUTH(DSNR USER) UACC(NONE)
   ```

   DB2OWNER now has class authorization for DSNR and USER. DB2OWNER can add users to the DB2/RACF environment, and the RDEFINE command entered later will be allowed. DB2OWNER will have control over and responsibility for the entire DB2/RACF security environment.

   ```
   ADDGROUP DB2 SUPGROUP(DBDC) OWNER(DB2OWNER)
   ```

   This creates the DB2 group under which all aliases will fall, and makes DB2OWNER the owner of that group.

   ```
   CONNECT DB2OWNER GROUP(DB2) AUTHORITY(JOIN) UACC(NONE)
   ```

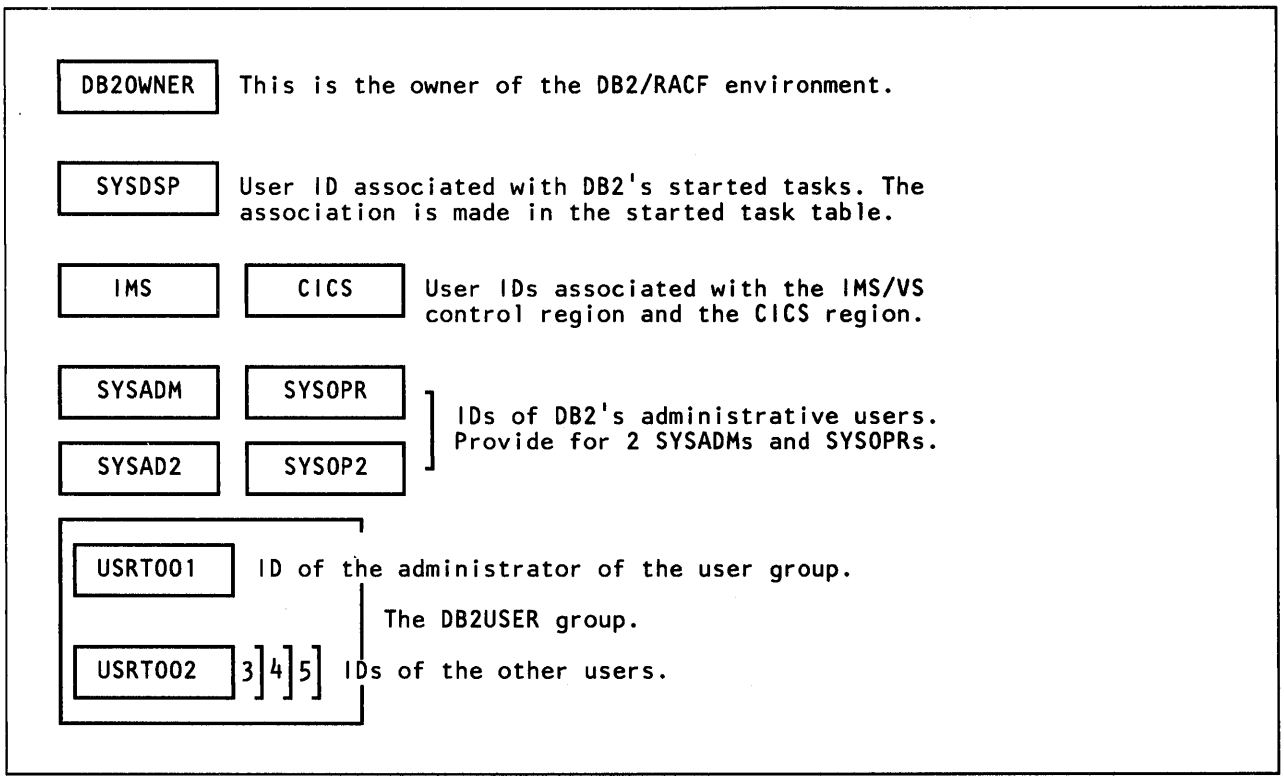


Figure 97. DB2/RACF Users and Groups

This connects DB2OWNER to the DB2 group with the authority to create new subgroups, add users, and manipulate profiles.

```
ALTUSER DB2OWNER DFLTGRP(DB2)
```

This makes DB2 the default group for DB2OWNER. Commands issued later by DB2OWNER will now have DB2 as the default group.

2. Add two more groups.

```
ADDGROUP DSNC130 SUPGROUP(DB2) OWNER(DB2OWNER)
```

This creates a group that will specifically control data sets that start with DSNC130. This is the DB2 Release 3 default for data bases and recovery logs.

```
ADDGROUP DSN130 SUPGROUP(DB2) OWNER(DB2OWNER)
```

This creates a group that will specifically control data sets that start with DSN130. This is the DB2 Release 3 default for distribution, target, SMP, and other install data sets.

3. Add the DB2 user ID and connect it.

```
ADDUSER SYSDSP UACC(NONE) DATA('USER ID FOR DB2 STARTED TASKS')
```

This statement is issued by DB2OWNER, so DB2OWNER is the owner of SYSDSP and the default group is DB2, which is the connect group (DFLTGRP) of DB2OWNER. Any resources that SYSDSP defines (data base data sets, for example) will have universal access of NONE, by default provided here.

```
CONNECT SYSDSP GROUP(DSNC130) AUTHORITY(CREATE) UACC(NONE)
```

This allows the DB2 started tasks to create data sets that start with DSNC130. The association of the SYSDSP *username* with these started tasks in ICHRIN03 provides the link between *username* and started task.

Universal access of NONE is provided. Those who can access data sets created by DB2 must be permitted access. This is an install decision of our sample site.

4. Add the DB2 administrative users and connect them.

```
ADDUSER (SYSADM SYSAD2 SYSOPR SYSOP2)

CONNECT (SYSADM SYSAD2 SYSOPR SYSOP2)
        GROUP(DSNC130) AUTHORITY(CREATE) UACC(NONE)

CONNECT (SYSADM SYSAD2) GROUP(DSN130)
        AUTHORITY(CREATE) UACC(NONE)
```

This adds the administrative users. SYSADMs and SYSOPRs can now create objects in DSNC130, and the SYSADMs can also create objects in the install libraries (DSN130). You will also want the system programmer to have access to DSN130.

5. Add general DB2 users and connect them.

```
ADDGROUP DB2USER SUPGROUP(DB2)

ADDUSER (USRT001 USRT002 USRT003 USRT004 USRT005) DFLTGRP(DB2USER)

CONNECT (USRT001 USRT002 USRT003 USRT004 USRT005)
     GROUP(DSNC130) AUTHORITY(CREATE) UACC(NONE)
```

Group DB2USER is created as a collection of all general DB2 users. Then DB2 is connected to the group. These users can now create data sets that start with DSNC130. In DB2 terms, this allows users to define spaces with STOGROUP where DB2 defines the data sets, or with VCAT where the user defines the data sets with access method services.

If DB2 or any of these users create data sets starting with DSNC130, the data sets will be RACF-protected. If other users try to create such data sets, they will be stopped by RACF, because RACF is always called to check for authorization.

6. Create some generic profiles for data sets.

   - For active logs:

     ```
     ADDSD 'DSNC130.LOGCOPY*' UACC(NONE)
     ```

   - For archive logs:

     ```
     ADDSD 'DSNC130.ARCHLOG*' UACC(NONE)
     ```

   - For BSDSs:

     ```
     ADDSD 'DSNC130.BSDS*' UACC(NONE)
     ```

   - For table spaces and index spaces:

     ```
     ADDSD 'DSNC130.DSNDBC.*' UACC(NONE)
     ```

   - For department STs spaces:

     ```
     ADDSD 'DSNC130.DSNDBC.ST*' UACC(NONE)
     ```

   - For anything else:

     ```
     ADDSD 'DSNC130.*' UACC(NONE)
     ```

   - For install libraries:

     ```
     ADDSD 'DSN130.*' UACC(READ)
     ```

Although these are not all absolutely necessary ('DSNC130.*' would include them all), the sample shows how generic profiles can be created for active and archive logs, bootstrap data sets, other DB2 data sets (including table spaces and index spaces), and the distribution, target, SMP, and install libraries. It is a good idea to provide separate generic profiles for each type of data.

Notice the separate profile for department ST's table spaces and index spaces. These are the ones that USRT001 will administer.

Some parameters, such as universal access, could be different for the different data sets. In this example, install data sets (DSN130.) are universally available for READing.

If using generic profiles, specify NO on install panel DSNTIPP for Archive Log RACF. If you specify YES, DB2 asks RACF to create a discrete profile for each and every archive log created. If you are using generic profiles, you do not want this.

Note: To protect VSAM data sets, use the cluster name. The data and index component names need not be protected, because the cluster name is used for RACF checking.

7. Permit DB2 complete control over data sets.

```
PERMIT 'DSNC130.LOGCOPY*' ID(SYSDSP) ACCESS(ALTER)
PERMIT 'DSNC130.ARCHLOG*' ID(SYSDSP) ACCESS(ALTER)
PERMIT 'DSNC130.BSDS*' ID(SYSDSP) ACCESS(ALTER)
PERMIT 'DSNC130.DSNDBC.*' ID(SYSDSP) ACCESS(ALTER)
PERMIT 'DSNC130.DSNDBC.ST*' ID(SYSDSP) ACCESS(ALTER)
PERMIT 'DSNC130.*' ID(SYSDSP) ACCESS(ALTER)
```

DB2 started tasks (through the association with SYSDSP *username*) are now also permitted to delete the DSNC130 data sets that the connection with the DSNC130 group allowed them to create.

Now only DB2 can create, alter, use, and delete data sets starting with DSNC130.

8. Define the DB2 administrators' capabilities.

```
PERMIT 'DSNC130.LOGCOPY*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNC130.ARCHLOG*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNC130.BSDS*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNC130.DSNDBC.*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNC130.DSNDBC.ST*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNC130.*'
       ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
PERMIT 'DSNX..*' ID(SYSADM SYSAD2) ACCESS(ALTER)
```

This gives the DB2 administrative users complete control over DSNC130 data sets. The SYSADMs also have complete control over the install libraries. The system programmer should also be permitted this control.

9. Define capabilities of the general users.

```
PERMIT 'DSNC130.LOGCOPY*' ID(USRT001) ACCESS(READ)
PERMIT 'DSNC130.ARCHLOG*' ID(USRT001) ACCESS(READ)
PERMIT 'DSNC130.BSDS*' ID(USRT001) ACCESS(READ)
PERMIT 'DSNC130.DSNDBC.ST*' ID(USRT001) ACCESS(ALTER)
PERMIT 'DSNC130.DSNDBC.*' ID(USRT001) ACCESS(READ)
```

USRT001 can read the recovery log data sets but the other general users cannot.

USRT001 can create, use, and delete data base data sets that start with DSNC130.DSNDBC.ST. These belong to the ST department, and USRT001 is the administrator.

The other users can create DSNC130 data base data sets, but they must go to the administrator to delete them. The permission to delete (ALTER) allows any data set in the group (DSNC130.DSNDBC.ST* here) to be deleted. Other data base data sets (group DSNC130.DSNDBC.*) can be read by USRT001.

To maximize the granularity of RACF authorization, each user can be given his or her own DB2 data base, and ADDSD and PERMIT commands can be issued for each data base. It could even be taken to a table space level, but this is generally not desirable.

- To add the group:

  ```
  ADDSD 'DSNC130.DSNDBC.dbname.*' UACC(NONE)
  ```

- To access DB2:

  ```
  PERMIT 'DSNC130.DSNDBC.dbname.*' ID(SYSDSP) ACCESS(ALTER)
  ```

- To permit SYSADM:

  ```
  PERMIT 'DSNC130.DSNDBC.dbname.*'
         ID(SYSADM SYSAD2 SYSOPR SYSOP2) ACCESS(ALTER)
  ```

- To permit user administrator:

  ```
  PERMIT 'DSNC130.DSNDBC.dbname.*'
         ID(user-administrator) ACCESS(ALTER)
  ```

  The ability to create the data sets is provided earlier when these users are connected to the DSNC130 group. Again, these are decisions of our sample site.

10. Define the resources belonging to the DSNR class.

```
RDEFINE  DSNR  (DSN.BATCH  DSN.MASS  DSN.SASS) OWNER(DB2OWNER)
```

This identifies to RACF that these three resources belong to the DSNR class. When DB2OWNER was defined to RACF, the CLAUTH parameter was specified as DSNR so that RACF knows that DB2OWNER is authorized for class DSNR. DB2OWNER owns these profiles, and has the right to change them.

```
PERMIT  DSN.BATCH  CLASS(DSNR)  ID(DB2USER) ACCESS(READ)
PERMIT  DSN.BATCH  CLASS(DSNR)
    ID(SYSADM,SYSAD2,SYSOPR,SYSOP2) ACCESS(READ)
PERMIT  DSN.MASS   CLASS(DSNR)  ID(IMS)  ACCESS(READ)
PERMIT  DSN.SASS   CLASS(DSNR)  ID(CICS)  ACCESS(READ)
```

This permits users access to these environments.

Here, all users belonging to the DB2USER group and the SYSADMs and SYSOPRs can be TSO users, can run batch jobs, and can execute DB2 utilities. The IMS/VS control region can connect with the IMS user ID, and CICS can connect with the CICS user ID. Because DB2OWNER created these resources (with the RDEFINE statement), he has ALTER access to them.

The ACCESS(READ) operand allows use of DB2. If ACCESS(ALTER) was specified, they could manipulate profiles and use DB2. If ACCESS(NONE) was specified, any access would be denied. For DB2 purposes, ACCESS(READ) is the correct specification.

11. Access with the DB2 service aids.

    The DSN1COPY and DSN1PRNT service aids access DB2 table space and index space data sets outside DB2 control. The DSN1LOGP service aid accesses the recovery log data sets (active logs, archive logs, and the BSDS) outside DB2 control. CHANGE LOG INVENTORY and PRINT LOG MAP utilities access the BSDS.

    Because these are batch jobs, the JOB statement USER=*username* and PASSWORD=*.*password* must be entered to supply a valid user ID for RACF to check. The USRT001 user ID in our sample will allow the recovery logs to be read with the DSN1LOGP service aid (and any other program) and the data base data sets starting with DSNC130.DSNDBC.ST to be accessed by the DSN1PRNT and DSN1COPY service aids (and any other program). Data base data sets that don't fall into any other group can also be read, because the DSNC130.DSNDBC. group will catch any that don't belong to another more specific category.

    In our sample, general users have not been allowed to access the log data sets and BSDSs outside DB2 control. USRT001 can read them.

    RACF Version 1 Release 7 provides the facility to allow access to objects on a program basis rather than a user basis. Thus, users who are not authorized to access the log data sets could do so only if they executed the DSN1LOGP service aid. Access to data base data sets could be permitted if accessed with DSN1PRNT or DSN1COPY.

Appendix C. Interpreting DB2 Trace Output

When you activate a DB2 trace, it produces trace records based on the parameters you specified for the -START TRACE command. Each record identifies one or more significant DB2 events. You can use DB2 Performance Monitor (DB2PM), a separately licensed program, to format, print and interpret DB2 trace output. However, if you don't have DB2PM or wish to do your own analysis of the trace output, you can use the information in this appendix, as well as Appendix D, "DB2 Trace Record Descriptions" and Appendix E, "Accounting and Statistics Records." By examining a DB2 trace record, you can determine the type of trace that produced the record (statistics, accounting, performance, or global) and the event (that is, the IFCID) the record reports.

Trace records can be written to SMF or GTF. Regardless of whether you write the record to SMF or GTF, it contains four basic groups — (1) an SMF or GTF writer header section, (2) a self-defining section, (3) a product section, and (4) zero or more data sections. GTF records are blocked to 256 bytes; see page 184 for the logic necessary to process blocked records sent to GTF. The length and content of the writer header section also differs between SMF records and GTF records. The other sections of the records are the same for SMF and GTF.

Figure 98 shows the format of DB2 trace records.

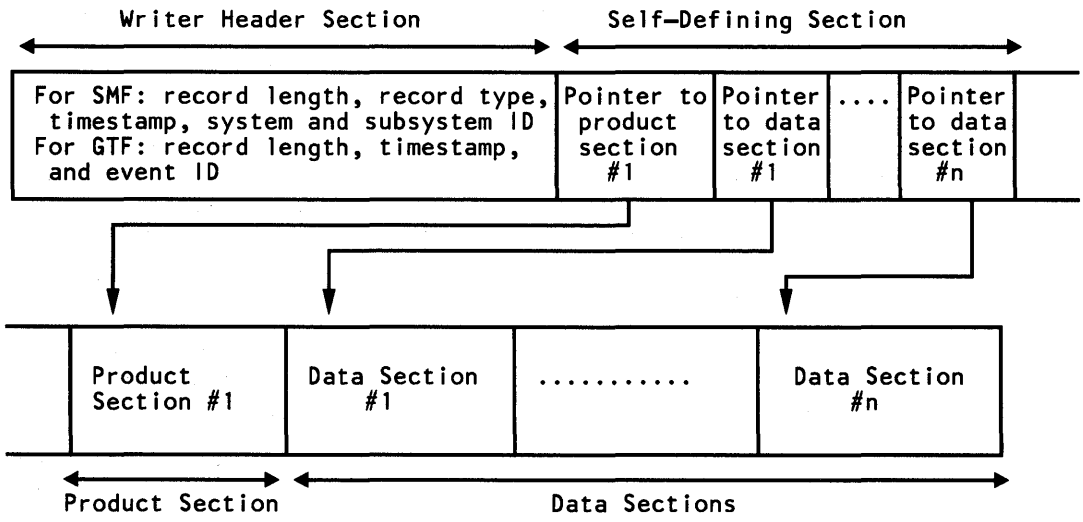


Figure 98. General Format of Trace Records Written by DB2

The writer header section begins at the first byte of the record and continues for a fixed length. (The GTF writer header is longer than the SMF writer header.) The self-defining section follows the writer header section. The product and data sections follow the self-defining section, but their locations within the record are not fixed. They "float" within the remainder of the record and must be located by pointers stored in the self-defining section.

The self-defining section follows the writer header section (both GTF and SMF). The first self-defining section always points to a special data section called the product section. The product section contains an IFCID. You will need to find this IFCID before you can interpret the record.

There are several types of records defined by the IFCIDs:

- IFCID 0001 is the statistics record for system services.
- IFCID 0002 is the statistics record for data base services.
- IFCID 0003 is the accounting record.
- IFCIDs greater than 0003 are the various performance trace records.

The product section also contains field QWHSNSDA, which indicates how many self-defining data sections the record contains. You can use this field to keep from trying to access data sections which do not exist. In trying to interpret the trace records, remember that the various keywords you specified when you started the trace will determine whether any data is collected. If no data has been collected, you can expect field QWHSNSDA to show a data length of zero.

To understand what is contained in each data section pointed to by the self-defining section, refer to Appendix D, "DB2 Trace Record Descriptions" and Appendix E, "Accounting and Statistics Records," or to the mapping macros supplied with DB2.

Different macros map different types of trace data: DSNDQWAS maps accounting records; DSNDQWST maps statistics records; DSNDQW00 and DSNDQW01 map performance records. Global serviceability records are not mapped.

Use the following procedure to determine the IFCID of a trace record using the appropriate mapping macro:

- Statistics records have an IFCID of 0001 or 0002. To access statistics data in SMF, use mapping macro "DSNDQWST SUBTYPE=ALL," and establish addressability using label "SM100." A GTF equivalent of DSNDQWST does not exist. To map statistics data to GTF, use macro DSNDQWST but substitute a GTF header for the SMF header. The GTF header is in macro DSNDQWGT.
- Accounting records have an IFCID of 0003. To access accounting data in SMF, use mapping macro "DSNDQWAS SUBTYPE=ALL," and establish addressability using label "SM101." A GTF equivalent of DSNDQWAS does not exist. To map statistics data to GTF, use macro DSNDQWAS but substitute a GTF header, which is in macro DSNDQWGT, for the SMF header.
- Performance records have an IFCID from 0004 to 0130. To access performance data in SMF, use mapping macro "DSNDQWSP SUBTYPE=ALL," and establish addressability using label "SM102." To access performance data in GTF, use mapping macro "DSNDQWGT SUBTYPE=ALL," and establish addressability using label "QWGT," the GTF header.

After establishing record addressability, you can examine the fields of the header section. These fields are described in Figure 99 on page 254. The first section explains SMF header fields; the second explains GTF header fields.

SMF Writer Header Section

The SMF writer header section begins at the first byte of the record. After establishing addressability, you can examine the header fields.

The fields are described in the Figure 99. The first three columns of the table show the field names for statistics, accounting, and performance records, respectively. The fourth column describes the field.

Statistics Label	Accounting Label	Performance Label	Field Description
SM100LEN	SM101LEN	SM102LEN	Total length of SMF record (halfword binary)
SM100SGD	SM101SGD	SM102SGD	ZZ bytes (2 bytes)
SM100FLG	SM101FLG	SM102FLG	System indicator (1 byte)
SM100RTY	SM101RTY	SM102RTY	SMF record type (1 byte binary) Statistics=100(dec), Accounting=101(dec), Performance=102(dec)
SM100TME	SM101TME	SM102TME	SMF record timestamp, time portion (4 bytes hex)
SM100DTE	SM101DTE	SM102DTE	SMF record timestamp, date portion (4 bytes hex)
SM100SID	SM101SID	SM102SID	System ID (4 characters)
SM100SSI	SM101SSI	SM102SSI	Subsystem ID (4 characters)
SM100STF	SM101STF	SM102STF	Reserved (1 byte)
SM100RI	SM101RI	SM102RI	Reserved (1 byte)
SM100BUF	SM101BUF	SM102BUF	Reserved (4 bytes)

Figure 99. Contents of SMF Writer Header Section

GTF Writer Header Section

The GTF writer header section begins at the first byte of the record. After establishing addressability, you can examine the fields of the header. The fields are described in Figure 100.

Label	Field Description
QWGTLEN	Length of Record (halfword)
Reserved	(halfword)

Figure 100 (Part 1 of 2). Contents of GTF Writer Header Section

Label	Field Description
QWGTAID	Application identifier (1 hex byte)
QWGTFID	Format ID (1 hex byte)
QWGTTIME	Timestamp; you must specify TIME=YES when you start GTF (8 hex bytes)
QWGTEID	Event ID: X'0FB9' (2 hex bytes)
QWGTASCB	ASCB Address (4 hex bytes)
QWGTJOBN	Job name (8 characters)
QWGTHDRE	Extension to header
QWGTDLEN	Length of data section (halfword)
QWGTDSCC	Segment control code (1 hex byte)
QWGTDZZ2	Reserved (1 hex byte)
QWGTSSID	Subsystem ID (4 characters)
QWGTWSEQ	Sequence number (fullword)
QWGTEND	End of GTF header

Figure 100 (Part 2 of 2). Contents of GTF Writer Header Section

Self-Defining Section

The self-defining section following the writer header contains pointers that enable you to find the product and data sections, which contain the actual trace data.

Each "pointer" is a descriptor containing 3 fields, which are:

1. The offset from the beginning of the record to the data section
2. The length of each item in the data section
3. The number of items occurring in the data section

   If this field contains "0," the data section is not in the record.

   If it contains a number greater than 1, multiple data items are stored contiguously within that data section. To find the second data item, add the length of the first data item to the address of the first data item (and so forth). Multiple data items within a specific data section always have the same length and format.

The offset field is a fullword; the length and number fields are halfwords.

Pointers occur in a fixed order, and their meanings are determined by the IFCID of the record. Different sets of pointers can occur, and each set is described by a separate DSECT. Therefore, to examine the pointers, you must first establish addressability using the DSECT that provides the appropriate description of the self-defining section. To do this:

1. Compute the address of the self-defining section.

The self-defining section begins at label "SM100END" for statistics records, "SM101END" for accounting records, and "SM102END" for performance records. It does not matter which mapping DSECT you use, because the length of the SMF writer header is always the same.

For GTF, use QWGTEND.

2. Determine the IFCID of the record.

   Use the first field in the self-defining section; it contains the offset from the beginning of the record to the product section. The product section contains the IFCID.

   The product section is mapped by DSNDQWHS; the IFCID is mapped by QWHSIID.

   For statistics records having an IFCID 0001, establish addressability using label "QWS0"; for statistics records having IFCID 0002, establish addressability using label "QWS1." For accounting records, establish addressability using label "QWA0." For performance records, establish addressability using label "QWT0."

After establishing addressability using the appropriate DSECT, use the pointers in the self-defining section to locate the record's data sections.

To help make your applications independent of possible future releases of DB2, always use the length values contained in the self-defining section rather than symbolic lengths that you may find in the macro expansions.

The relationship between the contents of the self-defining section "pointers" and the items in a data section is shown in Figure 101.

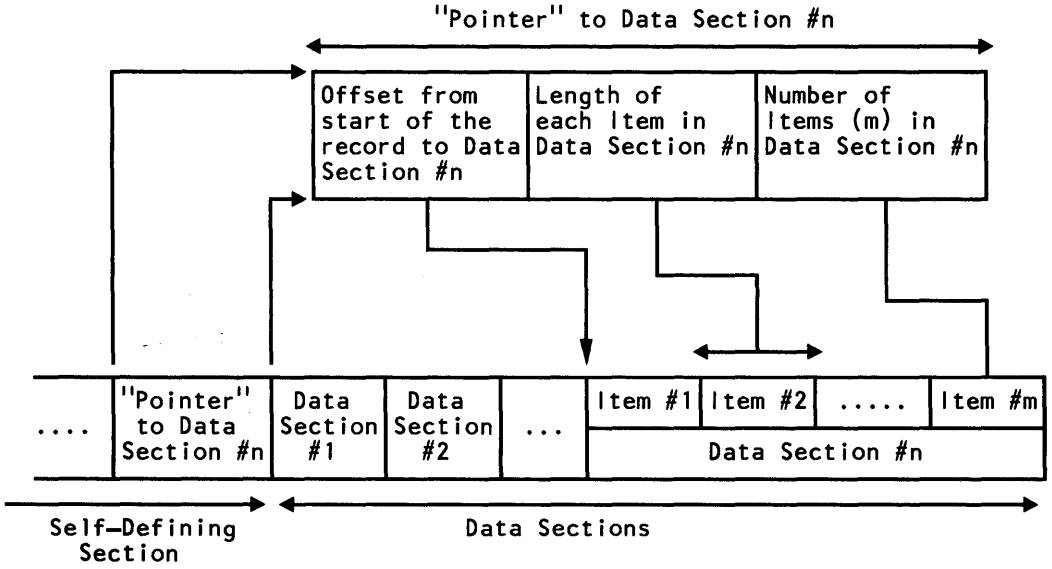


Figure 101. Relationship Between Self-Defining Section and Data Sections

Product Section

The product section for all record types consists of a standard header and, for accounting and performance records, also consists of a correlation header. If you start performance class 1 (which is statistics), the statistics records also have a correlation header.

The standard header identifies the:

RMID
IFCID
Timestamp
Writer sequence number
Trace(s) that created the record
Subsystem name
Release indicator
ACE address

The timestamp this header contains is different from the timestamp the writer header contains. The timestamp in the standard writer header is assigned through SMF or GTF processing. The timestamp appearing in the product section header is the STORE CLOCK VALUE. It is assigned by DB2.

The correlation header on accounting and performance records identifies:

The authorization ID
The plan name
The correlation ID
The connection name

Appendix D. DB2 Trace Record Descriptions

When you invoke a trace, DB2 produces IFCIDs — Instrumentation Facility Component Identification numbers. Each IFCID identifies a significant DB2 event and appears in the trace records that DB2 produces. IFCIDs are numbered from 0000 to 0139.

Many IFCIDs appear in pairs, such as 0006 and 0007 (begin and end read I/O), and 0023 and 0025 (begin and end utility). Others are not paired. IFCID 0024, for instance, records a phase change for a DB2 utility; a begin and end are not associated with this event.

Most IFCIDs are associated with one particular trace class. IFCID 0058 (end SQL statement), for instance, is associated with performance class 3 (SQL events). Some IFCIDs, however, are associated with more than one trace class. IFCID 0106, for instance, records the DB2 initialization parameters that were in effect when the trace was invoked. It is associated with more than 10 different trace classes.

This appendix consists of two sections. The first section lists and describes each IFCID that DB2 can generate.

The second section identifies each DB2 trace class (3 for statistics, 4 for accounting, and so forth) and lists the IFCIDs associated with each class.

These sections list and describe the IFCIDs. For detailed information about the fields that appear within each IFCID, examine the DSECTs of the assembler language mapping macros that describe the IFCIDs. The following chart indicates which mapping macro describes which IFCIDs.

IFCIDs	Mapping Macro
0000	Not mapped
0001 - 0002	DSNDQWST SUBTYPE=ALL
0003	DSNDQWAS SUBTYPE=ALL
0004 - 0057	DSNDQW00
0058 - 0122	DSNDQW01
0123 - 0129	Reserved
0130 - 0139	Serviceability only

Figure 102. IFCIDs and Corresponding Mapping Macros

The comments for many of the fields within these DSECTs are preceded by an "S." This indicates that the field is intended for use by IBM service personnel only.

Sequential IFCID Listing

Figure 103 describes each of the IFCIDs. It contains the IFCID, the identifier of the resource manager involved (RMID), the performance class, and a description of the event that the IFCID reports. For a list of the DB2 resource manager identifiers (RMIDs), see "Specifying a Resource Manager" on page 178.

IFCID	RMID	Performance Class	Event Description
0000	Actual RMID	N/A	Reserved for entry and exit tracing. A mapping macro is not provided for it. See *IBM DATABASE 2 Diagnosis Reference Volume 4: Service Traces* for further information.
0001	26	1	Reserved for system services statistics records (DSCF statistics). It is mapped by the assembler macro DSNDQWST SUBTYPE=0. See Appendix E, "Accounting and Statistics Records" for further information.
0002	26	1	Reserved for data base services statistics records (ADMF statistics). It is mapped by the assembler macro DSNDQWST SUBTYPE=1. See Appendix E, "Accounting and Statistics Records" for further information.
0003	26	2	Reserved for accounting records. It is mapped by the assembler macro DSNDQWAS. See Appendix E, "Accounting and Statistics Records" for further information.
0004	16	N/A	Identifies the beginning of a trace. It records the text that you used to invoke the trace.
0005	16	N/A	Identifies the end of a trace. It records the message text that you used to stop the trace.
0006	10	4	Identifies the data set before a read I/O operation. It records the buffer pool and page set ID, the first page number to be read, and the type of read request (read or sequential prefetch).
0007	10	4	Identifies the completion code after a read I/O operation. It records the number of pages read.
0008	10	4	Identifies the data set before a synchronous write I/O operation. It records the number of active buffers in the pool and the number of updated pages in the deferred write queue.
0009	10	4	Records the completion code after a synchronous or asynchronous write I/O operation.
0010	10	4	Identifies the data set before an asynchronous write I/O operation. It records the number of pages to be written, the number of active buffers in the pool, and the number of updated pages in the deferred write queue.

Figure 103 (Part 1 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0011	14	13	Identifies results of a validation exit call. It is written for every validated row.
0012	14	13	Identifies results of an edit exit call to encode a record. It is written for every written row.
0013	14	8	Records input to hash scan (DSHIHSET).
0014	14	8	Records end of hash scan.
0015	14	8	Identifies input to index scan (DSNIOSET).
0016	14	8	Identifies beginning of insert operation.
0017	14	8	Identifies input to sequential scan (DSNIRNXT). Written at the beginning of a table space or work file scan.
0018	14	8	Identifies end of an index scan, sequential scan, or insert operation.
0019	14	13	Identifies the results of an edit exit decode call. It is written for every row decoded.
0020	14	6	Identifies summary of page locks held. It records the maximum number of page locks held concurrently within the thread. It also records highest table space lock state, and whether lock escalation occurred for the table space and associated indexes. It does not contain a count of all suspensions or all locks acquired. This IFCID is written at COMMIT or ABORT time. Counters are not reset when this record is written.
0021	20	7	Identifies detail lock request taken on return from IRLM for every lock acquired, changed, or released. One record may correspond to multiple unlocks.
0022	22	3	Identifies the mini-plans generated. Miniplans are generated by the optimizer at BIND and SQL PREPARE.
0023	21	10	Identifies utility start information. It records the utility name, phase, and ID.
0024	21	10	Identifies utility object or phase change. It records the utility name, phase, and ID. It also indicates the number of items processed by the utility. These items can be pages (for COPY, RECOVER, MERGE, and RUNSTATS), records (for LOAD and REORG), or objects (for STOSPACE).
0025	21	10	Identifies utility end information. It records the utility name, phase, and ID. It also indicates the number of items processed by the utility. These items can be pages (for COPY, RECOVER, MERGE, and RUNSTATS), records (for LOAD and REORG), or objects (for STOSPACE).

Figure 103 (Part 2 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0026	20	9	Identifies a sort work file obtained. Records logical data set (may correspond to a physical data set).
0027	20	9	Identifies detailed sort information. It records the number of sequential records in this run.
0028	20	9	Identifies detailed sort information. It records the number of runs created and signifies the end of the sort phase.
0029	14	4	Identifies EDM I/O start activity.
0030	14	4	Identifies EDM I/O end activity. It records the number of calls to the data manager (requests for pieces of control block). Several directory I/Os may occur between 0029 and 0030.
0031	14	1	Identifies EDM pool full condition.
0032	04	5	Identifies begin wait for log manager.
0033	04	5	Identifies end wait for log manager.
0034	04	5	Identifies log manager wait for read I/O begin.
0035	04	5	Identifies log manager wait for read I/O end.
0036	04	5	Identifies log manager wait for non-I/O begin.
0037	04	5	Identifies log manager wait for non-I/O end.
0038	04	5	Identifies log manager wait for active log write I/O begin.
0039	04	5	Identifies log manager wait for active log write I/O end.
0040	04	5	Identifies log manager archive write I/O begin.
0041	04	5	Identifies log manager archive write I/O end.
0042	03	1	Identifies checkpoint started. Time is in the standard header.
0043	03	1	Identifies checkpoint ended. Time is in the standard header.
0044	20	6	Identifies lock suspend or an identify call to IRLM. It is written when an object (DBID, OBID) is not available to a thread. A thread remains suspended until the the object it requests becomes available, until a timeout occurs, or until a deadlock is detected. When one of these events occurs, a lock resume record (IFCID 0045) is written. A lock resume will occur for the previous lock suspend before another lock suspend can occur. IFCID 0044 is also written at DB2 start-up
0045	20	6	Identifies lock resume. It records the reason for the resume, noting whether it was a normal resume, the result of a deadlock, or the result of a timeout.

Figure 103 (Part 3 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0046	02	11	Identifies that current agent is about to switch execution unit. It records switch to new MVS dispatching unit.
0047	02	11	Identifies start of new SRB execution unit. It records new MVS dispatching unit under target ACE.
0048	02	11	Identifies completion of new SRB execution unit work request.
0049	02	11	Identifies start of new TCB execution unit. It records new MVS dispatching unit under target ACE.
0050	02	11	Identifies completion of new TCB execution unit work request.
0051	02	11	Identifies shared latch resumed. This IFCID is for serviceability only.
0052	02	11	Identifies shared latch wait. This IFCID is for serviceability only.
0053	02	11	Identifies shared latch resume. This IFCID is for serviceability only.
0054	20	6	Identifies lock contention.
0055	02	11	Identifies a shared latch resume. This IFCID is for serviceability only.
0056	02	11	Identifies exclusive latch wait. This IFCID is for serviceability only.
0057	02	11	Identifies exclusive latch resume. This IFCID is for serviceability only.
0058	22	3	Identifies the end of execution for an SQL statement. It is recorded following IFCIDs 0059 - 0066, which indicate the start of particular types of SQL statements. It records number of rows examined (for the right record type), number of rows processed (for any record type), and the number of rows inserted, updated, and deleted. It also records the number of pages scanned, and the number of rows qualified by the RDS and the data manager.
0059	22	3	Identifies start of SQL FETCH statement execution.
0060	22	3	Identifies start of SQL SELECT statement execution.
0061	22	3	Identifies start of SQL INSERT, UPDATE, and DELETE statement execution.
0062	22	3	Identifies start of SQL DDL statement execution.
0063	22	3	Identifies SQL statement produced by the parser. It is written for a static SQL BIND and for a dynamic SQL PREPARE.

Figure 103 (Part 4 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0064	22	3	Identifies start of SQL PREPARE statement execution.
0065	22	3	Identifies start of SQL OPEN CURSOR statement execution.
0066	22	3	Identifies start of SQL CLOSE CURSOR statement execution.
0067	26	14	Identifies start of accounting collection.
0068	07	2	Identifies the beginning of an ABORT request.
0069	07	2	Identifies the ending of an ABORT request.
0070	07	2	Identifies beginning of a COMMIT phase 2 request, IMS/VS or CICS.
0071	07	2	Identifies ending of a COMMIT phase 2 request, IMS/VS or CICS.
0072	07	2	Identifies beginning of a CREATE THREAD request.
0073	07	2	Identifies ending of a CREATE THREAD request.
0074	07	2	Identifies beginning of a TERMinate THREAD request.
0075	07	2	Identifies ending of a TERMinate THREAD request.
0076	07	1	Identifies beginning of an end of memory request, i.e. at TSO force.
0077	07	1	Identifies ending of an end of memory request.
0078	07	1	Identifies beginning of an end of task request. It is written at TCB, IMS/VS, and CICS abend.
0079	07	1	Identifies ending of an end of task request.
0080	07	2	Identifies beginning of establish exit request.
0081	07	2	Identifies ending of establish exit request.
0082	07	2	Identifies beginning of an IDENTIFY request. (IMS/VS, CICS, TSO connection)
0083	07	2	Identifies ending of an IDENTIFY request. (IMS/VS, CICS, TSO connection)
0084	07	2	Identifies beginning of a prepare to COMMIT request from IMS/VS or CICS for phase 1 of the COMMIT process.
0085	07	2	Identifies ending of a prepare to COMMIT request from IMS/VS or CICS for phase 1 of the COMMIT process.
0086	07	2	Identifies beginning of a signon request from IMS/VS or CICS.
0087	07	2	Identifies ending of a signon request from IMS/VS or CICS.
0088	07	2	Identifies beginning of a sync request in TSO.

Figure 103 (Part 5 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0089	07	2	Identifies ending of a sync request in TSO.
0090	23	10	Identifies command text of an entered DB2 command.
0091	23	10	Identifies completion status of a DB2 command.
0092	19	3	Identifies an access method services command start.
0093	02	11	Identifies when suspend was called. Suspension of TCB or SRB may or may not occur.
0094	02	11	Identifies when TCB or SRB was resumed. Store clock in standard header indicates resume time.
0095	22	3	Identifies sort started.
0096	22	3	Identifies sort ended. It records the number of records sorted, key sizes, work files, insertions, retrievals and initial runs.
0097	19	3	Identifies an access method services command completion. It records the command text and the access method services return code.
0098	06	12	Identifies before GETMAIN/FREEMAIN start - DSNSGMN.
0099	06	12	Identifies after GETMAIN/FREEMAIN end.
0100	06	12	Identifies before GETMAIN/FREEMAIN for pool related storage (expansion or contraction).
0101	06	12	Identifies after GETMAIN/FREEMAIN for pool related storage.
0102	06	1	Identifies detection of short on storage.
0103	06	1	Identifies setting off of short on storage.
0104	04	5	Identifies log data set mapping.
0105	10	1, 4, 6, 7, 8, 10, 13	Maps internal DBIDs and OBIDs to the actual database and table space referenced. This IFCID is generated at each -START TRACE.
0106	26	1 through 14	Identifies DB2 system parameters that are in effect when a trace started. It is generated for each -START TRACE.
0107	10	1, 4, 6, 7, 8, 10, 13	Identifies buffer manager OPEN/CLOSE information. It allows you to track individual opening and closing of objects within a trace period.
0108	20	10	Identifies BIND/REBIND beginning.
0109	20	10	Identifies BIND/REBIND end.
0110	20	10	Identifies BIND FREE beginning.
0111	20	10	Identifies BIND FREE end.

Figure 103 (Part 6 of 7). Sequential Listing of IFCIDs

IFCID	RMID	Performance Class	Event Description
0112	20	3	Identifies attributes for the plan after successful allocation of an allied thread.
0113	20	11	Identifies attributes for the plan after successful allocation of a system agent.
0114	04	5	Identifies start archive I/O wait.
0115	04	5	Identifies end archive I/O wait on DASD.
0116	04	5	Identifies end archive I/O wait on tape.
0117	04	5	Identifies begin archive read.
0118	04	5	Identifies end archive read.
0119	04	5	Identifies BSDS write I/O beginning.
0120	04	5	Identifies BSDS write I/O end.
0121	07	14	Identifies thread level entry into DB2.
0122	07	14	Identifies thread level exit from DB2.

Figure 103 (Part 7 of 7). Sequential Listing of IFCIDs

Performance Classes

There are 14 defined classes of performance trace:

background events
subsystem events
SQL events
buffer manager I/O and EDM requests
log manager I/O
lock information
lock detail
data manager detail
sort detail
utilities, bind, and commands
dispatching
storage manager
edit and validation
RARQ entry and exit

IFCID	RMID	Event
0001	26	System services statistics
0002	26	Data base services statistics
0031	14	EDM pool full
0042	03	Begin checkpoint
0043	03	End checkpoint
0076	07	Begin EOM
0077	07	End EOM
0078	07	Begin EOT
0079	07	End EOT
0102	06	On for short on storage
0103	06	Off for short on storage
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 104. Performance Class 1: Background Events

IFCID	RMID	Event
0003	26	Accounting
0068	07	Begin ABORT
0069	07	End ABORT
0070	07	Begin COMMIT phase 2
0071	07	End COMMIT phase 2
0072	07	Begin CREATE THREAD
0073	07	End CREATE THREAD
0074	07	Begin TERMINATE THREAD
0075	07	End TERMINATE THREAD
0080	07	Begin establish exit
0081	07	End establish exit
0082	07	Begin identify
0083	07	End identify
0084	07	Begin prepare to COMMIT
0085	07	End prepare to COMMIT
0086	07	Begin signon
0087	07	End signon
0088	07	Begin sync
0089	07	End sync
0106	26	System parameters in effect at trace invocation

Figure 105. Performance Class 2: Subsystem Events

Class 3 : SQL Events

IFCID	RMID	Event
0022	22	Mini-plans generated
0058	22	End SQL statement
0059	22	Begin FETCH SQL
0060	22	Begin SELECT SQL
0061	22	Begin INSERT, UPDATE, or DELETE
0062	22	Begin DDL
0063	22	Identify SQL statement
0064	22	Begin SQL PREPARE
0065	22	Begin OPEN CURSOR
0066	22	Begin CLOSE CURSOR
0092	19	Access method services command start
0095	22	Sort started
0096	22	Sort ended
0097	19	Access method services command end
0106	26	System parameters in effect at trace invocation
0112	20	Plan attributes for successful allied thread allocation

Figure 106. Performance Class 3: SQL Events

Class 4 : Buffer Manager I/O & EDM Requests

IFCID	RMID	Event
0006	10	Begin read I/O
0007	10	End read I/O
0008	10	Begin synchronous write I/O
0009	10	End synchronous or asynchronous write I/O
0010	10	Begin asynchronous write I/O
0029	14	Start EDM I/O request
0030	14	End EDM I/O request
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 107. Performance Class 4: Buffer Manager I/O & EDM Requests

Class 5 : Log Manager I/O

IFCID	RMID	Event
0032	04	Begin log manager wait
0033	04	End log manager wait
0034	04	Begin read I/O
0035	04	End read I/O
0036	04	Begin non-I/O wait
0037	04	End non-I/O wait
0038	04	Begin write I/O
0039	04	End write I/O
0040	04	Begin archive write
0041	04	End archive write
0104	04	Log manager data set mapping
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity
0114	04	Begin read I/O archive
0115	04	End read I/O archive DASD
0116	04	End read I/O archive tape
0117	04	Begin log wait archive
0118	04	End log wait archive
0119	04	Begin BSDS write I/O
0120	04	End BSDS write I/O

Figure 108. Performance Class 5: Log Manager I/O

Class 6 : Lock Information

IFCID	RMID	Event
0020	14	Page lock and table space lock summary
0044	20	Lock suspend
0045	20	Lock resume
0054	20	Lock contention
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 109. Performance Class 6: Lock Information

Class 7 : Lock Detail

IFCID	RMID	Event
0021	20	Every lock acquired, changed, or released
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 110. Performance Class 7: Lock Detail

Class 8 : Data Manager Detail

IFCID	RMID	Event
0013	14	Begin hash scan
0014	14	End hash scan
0015	14	Begin index scan
0016	14	End index scan
0017	14	Begin sequential scan
0018	14	End sequential scan
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 111. Performance Class 8: Data Manager Detail

Class 9 : Sort Detail

IFCID	RMID	Event
0026	20	Work file obtained
0027	20	Detailed sort information
0028	20	Detailed sort information
0106	26	System parameters in effect at trace invocation

Figure 112. Performance Class 9: Sort Detail

Class 10 : Utilities, Bind, and Commands

IFCID	RMID	Event
0023	21	Utility start
0024	21	Utility phase change
0025	21	Utility end
0090	23	DB2 command text
0091	23	DB2 command completion
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity
0108	20	Begin BIND/REBIND
0109	20	End BIND/REBIND
0110	20	Begin BIND FREE
0111	20	End BIND FREE

Figure 113. Performance Class 10: Utilities, Bind, and Commands

Class 11 : Dispatching

IFCID	RMID	Event
0046	02	Begin execution unit switch
0047	02	Begin new SRB
0048	02	End new SRB
0049	02	Begin new TCB
0050	02	End new TCB
0051	02	Shared latch resume (Serviceability only information)
0052	02	Shared latch wait (Serviceability only information)
0053	02	Shared latch resume (Serviceability only information)
0055	02	Shared latch resume (Serviceability only information)
0056	02	Exclusive latch wait (Serviceability only information)
0057	02	Exclusive latch resume (Serviceability only information)
0093	02	Suspend called
0094	02	Event resumed
0106	26	System parameters in effect at trace invocation
0113	20	Plan attributes for successful system agent allocation

Figure 114. Performance Class 11: Dispatching

Class 12 : Storage Manager

IFCID	RMID	Event
0098	06	Begin GETMAIN/FREEMAIN
0099	06	End GETMAIN/FREEMAIN
0100	06	Begin GETMAIN/FREEMAIN (pool related)
0101	06	End GETMAIN/FREEMAIN (pool related)
0106	26	System parameters in effect at trace invocation

Figure 115. Performance Class 12: Storage Manager

Class 13 : Edit and Validation

IFCID	RMID	Event
0011	14	Validation exit
0012	14	Edit exit - Encode
0019	14	Edit exit - Decode
0105	10	Buffer manager mapping of DBIDs and OBIDs
0106	26	System parameters in effect at trace invocation
0107	10	Buffer manager OPEN/CLOSE activity

Figure 116. Performance Class 13: Edit and Validation

Class 14 : RARQ Entry and Exit

IFCID	RMID	Event
0067	26	Accounting beginning
0106	26	System parameters
0121	07	RARQ entry
0122	07	RARQ exit

Figure 117. Performance Class 14: RARQ Entry and Exit

Class 15 : Reserved

Appendix E. Accounting and Statistics Records

This appendix presents details on the contents of the self-defining and data sections of DB2 statistics and accounting records. For introductory information about these records, see Appendix C, "Interpreting DB2 Trace Output" on page 252.

Self Defining Section

Descriptions of the self-defining section pointers, and the names of the labels that can be used to address them, are provided in the three tables that follow. Be sure to use the table that applies to the particular IFCIDs you are examining.

Label	Description
QWA01PSO QWA01PSL QWA01PSN	Offset from start of record to product data section Length of product data section items Number of items in product data section
QWA01R1O QWA01R1L QWA01R1N	Offset from start of record to instrumentation data section Length of instrumentation data section items Number of items in instrumentation data section
QWA01R2O QWA01R2L QWA01R2N	Offset from start of record to SQL statement data section Length of SQL statement data section items Number of items in SQL statement data section
QWA01R3O QWA01R3L QWA01R3N	Offset from start of record to buffer manager data section Length of buffer manager data section items Number of items in buffer manager data section
QWA01R4O QWA01R4L QWA01R4N	Offset from start of record to lock usage data section Length of lock usage data section items Number of items in lock usage data section
QWA01R5O QWA01R5L QWA01R5N	Reserved Reserved Reserved

Figure 118. Self-Defining Section Pointers in IFCID 0003 Accounting Records

Label	Description
QWS00PSO QWS00PSL QWS00PSN	Offset from start of record to product data section Length of product data section items Number of items in product data section
QWS00R1O QWS00R1L QWS00R1N	Offset from start of record to address space data section Length of address space data section items Number of items in address space data section

Figure 119 (Part 1 of 2). Self-Defining Section Pointers for IFCID 0001 Statistics Records

Label	Description
QWS00R2O	Offset from start of record to instrumentation destination data section
QWS00R2L	Length of instrumentation destination data section items
QWS00R2N	Number of items in instrumentation destination data section
QWS00R3O	Offset from start of record to instrumentation IFCID data section
QWS00R3L	Length of instrumentation data section items
QWS00R3N	Number of items in instrumentation data section
QWS00R4O	Offset from start of record to subsystem services data section
QWS00R4L	Length of subsystem services data section items
QWS00R4N	Number of items in subsystem services data section
QWS00R5O	Offset from start of record to command data section
QWS00R5L	Length of command data section items
QWS00R5N	Number of items in command data section
QWS00R6O	Offset from start of record to IFC checkpoint data section
QWS00R6L	Length of data section
QWS00R6N	Number of items in data section
QWS00R7O	Offset from start of record to latch manager data section
QWS00R7L	Length of latch manager data section items
QWS00R7N	Number of items in latch manager data section
QWS00R8O	Offset from start of record to agent services data section
QWS00R8L	Length of agent services data section items
QWS00R8N	Number of items in agent services data section
QWS00R9O	Offset from start of record to storage manager data section
QWS00R9L	Length of storage manager data section items
QWS00R9N	Number of items in storage manager data section
QWS00RAO	Reserved
QWS00RAL	Reserved
QWS00RAN	Reserved
QWS00RBO	Offset from start of record to log manager data section
QWS00RBL	Length of log manager data section items
QWS00RBN	Number of items in log manager data section
QWS00RCO	Reserved
QWS00RCL	Reserved
QWS00RCN	Reserved

Figure 119 (Part 2 of 2). Self-Defining Section Pointers for IFCID 0001 Statistics Records

Label	Description
QWS10PSO QWS10PSL QWS10PSN	Offset from start of record to product data section Length of product data section items Number of items in product data section
QWS10R1O QWS10R1L QWS10R1N	Offset from start of record to SQL statement data section Length of SQL statement data section items Number of items in SQL statement data section
QWS10R2O QWS10R2L QWS10R2N	Offset from start of record to bind data section Length of bind data section items Number of items in bind data section
QWS10R3O QWS10R3L QWS10R3N	Offset from start of record to buffer manager data section Length of buffer manager data section items Number of items in buffer manager data section
QWS10R4O QWS10R4L QWS10R4N	Reserved Reserved Reserved
QWS10R5O QWS10R5L QWS10R5N	Offset from start of record to lock usage data section Length of lock usage data section items Number of items in lock usage data section
QWS10R6O QWS10R6L QWS10R6N	Offset from start of record to EDM data section Length of EDM data section items Number of items in EDM data section

Figure 120. Self-Defining Section Pointers in IFCID 0002 Statistics Records

Data Sections

The actual accounting and statistical data in the records written by DB2 is located in the record's data sections. The data sections can be found by pointers that are stored in the record's self-defining section (described in "Self-Defining Section" on page 255).

Each data section contains many separate "items." The number of items that occurs in a given data section is contained in that data section's entry in the self-defining section. (The relationship between the self-defining section "pointers" and the items in a given data section is shown in Figure 101 on page 256.)

Before you can examine the data items stored in the various data sections pointed to by the self-defining section, you must:

- Compute the address of the particular data section that you want to examine and verify its presence (this is described in "Self-Defining Section" on page 255).
- Establish data section item addressability using the appropriate label. Each of the data sections is mapped by its own DSECT (which has already been generated for you by the SUBTYPE=ALL operand of the DSNDQWAS or DSNDQWST macro). The specific DSECT labels that you should use to establish data section addressability are given below with the descriptions of the individual data sections.

System Services Statistics Records: IFICD 0001

This section describes the contents of IFCID 0001 statistics record data sections. Be sure that you have read "Data Sections" on page 276.

Product Data Section

The product data sections of IFCID 0001 and 0002 statistics records are identical to the standard header for the IFCID 0003 accounting record. To examine the contents of the standard header (which begins at the first byte of the product data section), compute the address of the product data section, then establish addressability using label "QWHS."

Descriptions of the standard header fields, and the labels to use when addressing the fields, are given in Figure 121.

Label	Description
QWHSLEN	Length of standard header (halfword binary)
QWHSTYP	Header type (1 byte binary), defined by equates: QWHSHS01 = standard header QWHSHC02 = correlation header
QWHSRMID	DB2 resource manager ID (1 byte)
QWHSIID	Instrumentation ID (IFCID 2 bytes), defined by equates: QWHS0001 = system statistics QWHS0002 = data base statistics QWHS0003 = accounting
QWHSNSDA	Number of self defining fields (1 byte)
QWHSRN	DB2 release indicator (1 byte)
QWHSACE	ACE address (4 bytes, binary)
QWHSSSID	Subsystem ID (4 characters)
QWHSSTCK	Store clock value for header (8 bytes, hex)
QWHSISEQ	Sequence number for IFCID (4 bytes, binary)
QWHSWSEQ	Sequence number for destination (4 bytes, binary)
QWHSMTN	Active trace number mask (4 bytes, hexidecimal)

Figure 121. Contents of Product Data Section Standard Header

Address Space Data Section

The address space data section can be found by using the contents of "QWS00R1O," "QWS00R1L," and "QWS00R1N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the address space data section, you should establish its addressability using label "QWSA."

The labels to use when accessing the fields in the address space data section, and descriptions of the field contents, are given in Figure 122.

Label	Description
QWSAPROC	Last 4 characters of the name of the procedure used to start the address space or a constant
QWSAEJST	Accumulated TCB time for the address space (8 bytes)
QWSASRBT	Accumulated SRB time for the address space (8 bytes)

Figure 122. Contents of IFCID 0001 Statistics Address Space Data Section

The accumulated processor time for the DB2 address spaces enables you to obtain processor time that is not charged to a non-DB2 address space. Each of the three fields is contained within the address space for system services, the address space for data base services, and the address space for the IRLM. TCB time is incremented in the dispatchable home address space. Normally, the address space of the application is charged for the processor time even though the request is running in a DB2 address space. SRB time is accumulated in the address space where the SRB is scheduled.

Instrumentation Destination Data Section

The instrumentation destination data section can be found by using the contents of "QWS00R2O," "QWS00R2L," and "QWS00R2N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QWSB."

This data section shows the number of records that could not be externalized and the number of records that were successfully written. The labels to use when accessing the fields in the instrumentation destination data section, and descriptions of the field contents, are given in Figure 123.

Label	Description
QWSBNM	Name of destination, for example "SMF" (4 characters)
QWSBWSEQ	Unique sequence number (fullword binary)
QWSBSRSW	Count of records successfully written (fullword)
QWSBSRNW	Count of records unsuccessfully written (fullword)
QWSBSBUF	Count of buffer errors (halfword)
QWSBSACT	Count of not-active errors (halfword)
QWSBSRNA	Count of record-not-accepted errors (halfword)
QWSBSWF	Count of writer failures (halfword)
QWSBOTH1	Reserved (1 byte)
QWSBOTH2	Reserved (1 byte)
QWSBOTH3	Reserved (1 byte)
QWSBOTH4	Reserved (1 byte)

Figure 123. Contents of IFCID 0001 Statistics Instrumentation Destination Data Section

Instrumentation Data Section

The instrumentation data section can be found by using the contents of "QWS00R3O," "QWS00R3L," and "QWS00R3N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability using label "QWSC."

The labels to use when accessing the fields in the instrumentation data section, and descriptions of the field contents, are given in Figure 124.

Label	Description
QWSCIID	IFCID value (halfword), possible values are: Equate "QWHS0001" - Subsystem Statistics "QWHS0002" - Data Base Statistics "QWHS0003" - Accounting
QWSCISEQ	IFCID sequence number (fullword binary)
QWSCSRSW	Records successfully written for IFC (fullword)
QWSCSRNW	Records not written for IFC (fullword)
QWSCSRND	Count of records not desired (halfword)
QWSCSBNA	Count of buffer-not-available errors (halfword)
QWSCSCF	Count of collection failures (halfword)
QWSCOTH1	Reserved (1 byte)
QWSCOTH2	Reserved (1 byte)

Figure 124. Contents of IFCID 0001 Statistics Instrumentation Data Section

Subsystem Services Data Section

The subsystem services data section can be found by using the contents of "QWS00R4O," "QWS00R4L," and "QWS00R4N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability using label "Q3ST."

All count fields in this data section are fullword binary.

The labels to use when accessing the fields in the subsystem services data section, and descriptions of the field contents, are given in Figure 125 on page 280.

Label	Description
Q3STIDEN	Successful identify: Counter is incremented each time TSO, IMS/VS, CICS, or a utility successfully connects to DB2.
Q3STSIGN	Successful signon: Indicates the number of times the user authorization ID was updated (for example, new message from transaction manager).
Q3STCTHD	Successful create thread: Indicates the number of threads created. A thread is required before an application can use SQL. Once established, a thread may process one or multiple authorization IDs (sign-ons).
Q3STTERM	Successful terminate: Indicates the number of times that a thread, signon, or identify is terminated. Because a sign-on can be performed without a terminate, this count cannot give an exact count of how often a thread is terminated.
Q3STRIUR	Successful resolve in-doubt unit of recovery: The number of times requests to handle in-doubt work units were successfully processed.
Q3STPREP	Successful prepare to commit: Count of the number of successful requests for commit phase 1 (begin commit).
Q3STCOMM	Successful commit: Count of successful requests for commit phase 2. If the number of prepares ("Q3STPREP") is not equal to the number of commits ("Q3STCOMM"), then the unaccounted-for work units have either been aborted or become in-doubt.
Q3STABRT	Successful abort: Indicates the number of times that a unit of recovery was backed out. Possible reasons for the abort: - Application abend - Application roll back request - Application deadlocked on data base records - Thread abend caused by resource shortage - Application canceled by operator
Q3STSYNC	Successful synchronize: Number of synchronized commit requests. TSO applications use the synchronize request; IMS/VS and CICS applications normally use the prepare and commit sequence to commit work.
Q3STEXIT	Successful DSN3EXIT: DSN3EXIT is a function of the subsystem services subcomponent that establishes or removes an exit. For example, in TSO there should be one exit per user per session.
Q3STINDT	Number of in-doubt threads: Count of in-doubt threads shows the number of application failures after a successful prepare and before a successful commit. The failure can occur in the address space of the application, the transaction manager, DB2, or all of these.

Figure 125 (Part 1 of 2). Contents of IFCID 0001 Statistics Subsystem Services Data Section

Label	Description
Q3STMEOT	End-of-task: Shows the number of times a non-DB2 task abended while connected to DB2.
Q3STMEOM	End-of-memory: Shows the number of times a non-DB2 address space was deleted by MVS while it was connected to DB2.
Q3STSSSI	Total number of SSI calls: The number of subsystem interface calls includes: - EOT - EOM - Subsystem identify - Commands from MVS console - HELP requests
Q3STCTHW	Number of create thread requests which waited.

Figure 125 (Part 2 of 2). Contents of IFCID 0001 Statistics Subsystem Services Data Section

Command Data Section

The command data section can be found by using the contents of "QWS00R5O," "QWS00R5L," and "QWS00R5N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "Q9ST."

All fields in this data section are fullword binary counts.

The labels to use when accessing the fields in the command data section, and descriptions of the field contents, are given in Figure 126.

Label	Description
Q9STCTR0	Count of -DISPLAY DATABASE commands
Q9STCTR1	Count of -DISPLAY THREAD commands
Q9STCTR2	Count of -DISPLAY UTILITY commands
Q9STCTR3	Count of -RECOVER BSDS commands
Q9STCTR4	Count of -RECOVER INDOUBT commands
Q9STCTR5	Count of -START DATABASE commands
Q9STCTR6	Count of -START TRACE commands
Q9STCTR7	Count of -START DB2 commands
Q9STCTR8	Count of -STOP DATABASE commands

Figure 126 (Part 1 of 2). Contents of IFCID 0001 Statistics Record Command Data Section

Label	Description
Q9STCTR9	Count of -STOP TRACE commands
Q9STCTRA	Count of -STOP DB2 commands
Q9STCTRB	Count of -TERM UTILITY commands
Q9STCTRC	Count of -DISPLAY TRACE commands
Q9STCTRD	Reserved
Q9STEROR	Count of unrecognized commands, incremented if command verb or primary keyword cannot be determined.

Figure 126 (Part 2 of 2). Contents of IFCID 0001 Statistics Record Command Data Section

IFC Checkpoint Data Section

This data section can be found by using the contents of "QWS00R6O," "QWS00R6L," and "QWS00R6N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QWSD."

All fields in this data section are fullword binary counts.

The labels to use when accessing the fields in the command data section, and descriptions of the field contents, are given in Figure 127.

Label	Description
QWSDCKPT	Count of DB2 checkpoints
QWSDINV	Reason statistics was invoked 04 = DB2 startup 08 = DB2 shutdown 0C = Activated by command 10 = Activated by timer 14 = Activated by checkpoint 18 = Activated during trace 1C = Activated during accounting 20 = Reserved

Figure 127. Contents of IFCID 0001 Statistics IFC Checkpoint Data Section

Latch Manager Data Section

The latch manager data section can be found by using the contents of "QWS00R7O," "QWS00R7L," and "QWS00R7N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability using label "QVLS."

All fields in this data section are fullword binary counts.

Each field contains the number of latch contentions for its particular latch level. An excessive number of latch contentions reduces DB2 performance. Because the number of latch contentions cannot be affected by install and migration parameters, you should call IBM for assistance if you believe the number of latch contentions to be excessive.

The labels to use when accessing the fields in the latch manager data section, and descriptions of the field contents, are given in Figure 128.

Label	Description
QVLSLC01	Number of contentions, latch level 1
QVLSLC02	Number of contentions, latch level 2
QVLSLC03	Number of contentions, latch level 3
QVLSLC04	Number of contentions, latch level 4
QVLSLC05	Number of contentions, latch level 5
QVLSLC06	Number of contentions, latch level 6
QVLSLC07	Number of contentions, latch level 7
QVLSLC08	Number of contentions, latch level 8
QVLSLC09	Number of contentions, latch level 9
QVLSLC10	Number of contentions, latch level 10
QVLSLC11	Number of contentions, latch level 11
QVLSLC12	Number of contentions, latch level 12
QVLSLC13	Number of contentions, latch level 13
QVLSLC14	Number of contentions, latch level 14
QVLSLC15	Number of contentions, latch level 15
QVLSLC16	Number of contentions, latch level 16
QVLSLC17	Number of contentions, latch level 17
QVLSLC18	Number of contentions, latch level 18
QVLSLC19	Number of contentions, latch level 19
QVLSLC20	Number of contentions, latch level 20
QVLSLC21	Number of contentions, latch level 21
QVLSLC22	Number of contentions, latch level 22
QVLSLC23	Number of contentions, latch level 23
QVLSLC24	Number of contentions, latch level 24
QVLSLC25	Number of contentions, latch level 25
QVLSLC26	Number of contentions, latch level 26
QVLSLC27	Number of contentions, latch level 27
QVLSLC28	Number of contentions, latch level 28

Figure 128 (Part 1 of 2). Contents of IFCID 0001 Statistics Latch Manager Data Section

Label	Description
QVLSLC29	Number of contentions, latch level 29
QVLSLC30	Number of contentions, latch level 30
QVLSLC31	Number of contentions, latch level 31
QVLSLC32	Number of contentions, latch level 32

Figure 128 (Part 2 of 2). Contents of IFCID 0001 Statistics Latch Manager Data Section

Agent Services Data Section

The agent services data section can be found by using the contents of "QWS00R8O," "QWS00R8L," and "QWS00R8N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QVAS."

All fields in this data section are fullword binary counts.

The labels to use when accessing the fields in the agent services data section, and descriptions of the field contents, are given in Figure 129.

Label	Description
QVASSUSP	Physical suspends (waits): These suspends occur because of latch contention, lock contention, I/O, and execution unit switching.
QVASXSUS	Synchronous unrelated SRBs scheduled
QVASXSUT	Synchronous unrelated TCBs scheduled
QVASXAUS	Asynchronous unrelated SRBs scheduled
QVASXAUT	Asynchronous unrelated TCBs scheduled
QVASXSRS	Synchronous related SRBs scheduled
QVASXSRT	Synchronous related TCBs scheduled
QVASADUR	Allocation failure, unavailable resource: A plan has become unavailable. For example, it has been invalidated and an unsuccessful automatic bind has occurred.
QVASADDL	Allocation failure, deadlock: Allocation deadlocks can occur when resources are limited and contention is high. An example is a table space lock that cannot be acquired because it is locked by another application. For information on changing install and migration parameters, see Chapter 9, "Updating DB2 Install and Migration Parameters" on page 204.
QVASADIR	Allocation failure, invalid resource: A request was made for allocation of a plan that was unknown to DB2.

Figure 129. Contents of IFCID 0001 Statistics Agent Services Data Section

If you find that allocation failures are occurring, you may be able to correct the problem by changing the appropriate install or migration parameter value (for example, number of concurrent users, number of plans, or number of table spaces).

Storage Manager Data Section

The storage manager data section can be found by using the contents of "QWS00R9O," "QWS00R9L," and "QWS00R9N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QSST."

All count fields in this data section are fullword binary.

The labels to use when accessing the fields in the storage manager data section, and descriptions of the field contents, are given in Figure 130.

Label	Description
QSSTID	Control block ID (2 characters)
QSSTLEN	Length of control block (halfword)
QSSTDESC	Control block eye catcher (4 characters)
QSSTGPLF	Number of fixed pools created
QSSTFPLF	Number of fixed pools deallocated
QSSTFREF	Number of fixed pool segments freed
QSSTEXPF	Number of fixed pool segments expanded
QSSTCONF	Number of fixed pool segments contracted
QSSTGPLV	Number of variable pools created
QSSTFPLV	Number of variable pools deallocated
QSSTFREV	Number of variable pool segments freed
QSSTEXPV	Number of variable pool segments expanded
QSSTCONV	Number of variable pool segments contracted
QSSTGETM	Number of GETMAIN requests for other than fixed or variable pools
QSSTFREM	Number of FREEMAIN requests for other than fixed and variable pools
QSSTRCNZ	Number of nonzero returns from GETMAIN/FREEMAIN
QSSTCONT	Number of storage contractions caused by a shortage of storage
QSSTCRIT	Number of times a shortage of storage was detected
QSSTABND	Number of abends issued due to a shortage of storage

Figure 130. Contents of IFCID 0001 Statistics Storage Manager Data Section

Log Manager Data Section

The log manager data section can be found by using the contents of "QWS00RBO," "QWS00RBL," and "QWS00RBN," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QJST."

All count fields in this data section are fullword binary.

The labels to use when accessing the fields in the log manager data section, and descriptions of the field contents, are given in Figure 131.

Label	Description
QJSTID	Control block ID (2 characters)
QJSTLL	Control block length (halfword)
QJSTEID	Control block eye catcher (4 characters)
QJSTWRW	Number of log write requests, WAIT
QJSTWRNW	Number of log write requests, NOWAIT
QJSTWRF	Number of log write requests, FORCE
QJSTWTB	Number of waits because of unavailable write buffer
QJSTRBUF	Number of reads satisfied from output buffers
QJSTRACT	Number of reads satisfied from active log data set
QJSTRARH	Number of reads satisfied from archive log data set
QJSTWTL	Number of reads delayed because archive allocation limit reached
QJSTBSDS	Total number of BSDS accesses
QJSTBFFL	Number of active log output control intervals created
QJSTBFWR	Number of calls to write active log buffers
QJSTALR	Number of archive log read allocations
QJSTALW	Number of archive log write allocations
QJSTCIOF	Number of control intervals off-loaded

Figure 131. Contents of IFCID 0001 Statistics Log Manager Data Section

Field Usage Notes:

- **"QJSTWRW," "QJSTWRNW," "QJSTWRF":** The sum of these write request counts indicates the amount of active log activity. If the percentage of forced writes is too high, it can affect subsystem performance.
- **"QJSTWTB":** This count indicates the degree of synchronization between the active log buffer and the active log device.
- **"QJSTRBUF," "QJSTRACT," "QJSTARH":** These three read counters indicate the degree of subsystem log record retrieval efficiency. You can use these counts to adjust the number of output buffers and the total active log capacity to maximize DB2 performance.

- **"QJSTWTL":** If this count is not zero, it indicates a need for more archive log data sets.
- **"QJSTALR," "QJSTALW":** These counters indicate the frequency of archive log OPEN/CLOSE activity. A high read allocation count indicates a need for more active log data sets.

Data Base Services Statistics Records: IFCID 0002

This section contains descriptions of the contents of the IFCID 0002 statistics record data sections. Be sure that you have read "Data Sections" on page 276.

Product Data Section

The product data section of the IFCID 0002 statistics record is identical to the product data section of the IFCID 0001 statistics record described in "Product Data Section" on page 277. Both product data sections are identical to the standard header described in "Product Data Section" on page 294.

SQL Statement Data Section

This data section enables you to monitor the types of SQL statements the applications at your site are using most frequently, and can help you tune your DB2 subsystem for maximum performance.

The SQL statement data section can be found by using the contents of "QWA01R2O," "QWA01R2L," and "QWA01R2N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the SQL statement data section, you should establish its addressability using label "QXST."

All the counts in the SQL statement data section are fullword binary values.

The labels to use in accessing the SQL statement data section fields, and descriptions of the fields, are given in Figure 132.

Label	Description
QXID	Control block ID (halfword)
QXLEN	Control block length (halfword)
QXEYE	Control block eye catcher (4 characters)
QXSELECT	Count of SELECT statements
QXINSRT	Count of INSERT statements
QXUPDTE	Count of UPDATE statements
QXDELET	Count of DELETE statements
QXDESC	Count of DESCRIBE statements
QXPREP	Count of PREPARE statements

Figure 132 (Part 1 of 2). Contents of Accounting Record SQL Statement Data Section

Label	Description
QXOPEN	Count of OPEN cursor statements
QXCLOSE	Count of CLOSE cursor statements
QXCRTAB	Count of CREATE TABLE statements
QXCRINDX	Count of CREATE INDEX statements
QXCTABS	Count of CREATE TABLESPACE statements
QXCRSYN	Count of CREATE SYNONYM statements
QXCRDAB	Count of CREATE DATABASE statements
QXCRSTG	Count of CREATE STOGROUP statements
QXDEFVU	Count of CREATE VIEW statements
QXDRPIX	Count of DROP INDEX statements
QXDRPTA	Count of DROP TABLE statements
QXDRPTS	Count of DROP TABLESPACE statements
QXDRPDB	Count of DROP DATABASE statements
QXDRPSY	Count of DROP SYNONYM statements
QXDRPST	Count of DROP STOGROUP statements
QXDRPVU	Count of DROP VIEW statements
QXALTST	Count of ALTER STOGROUP statements
QXFETCH	Count of FETCH statements
QXALTTS	Count of ALTER TABLESPACE statements
QXALTTA	Count of ALTER TABLE statements
QXALTIX	Count of ALTER INDEX statements
QXCMTON	Count of COMMENT ON statements
QXLOCK	Count of LOCK TABLE statements
QXGRANT	Count of GRANT statements
QXREVOK	Count of REVOKE statements
QXINCRB	Count of incremental binds excluding prepares
QXLABON	Count of label on statements

Figure 132 (Part 2 of 2). Contents of Accounting Record SQL Statement Data Section

Bind Data Section

The bind data section can be found by using the contents of "QWS10R2O," "QWS10R2L," and "QWS10R2N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QTST."

All count fields in this data section are fullword binary.

You can use these counts to help determine the frequency of binds and allocations. They keep track of all bind operations and contain, for example, the count of successfully bound static SQL statements and authorization (validity) checks that DB2 had to make on the SQL statements. For more information about validity checking, see "Using the Bind Process" on page 198.

The labels to use when accessing the fields in the bind data section, and descriptions of the field contents, are given in Figure 133.

Label	Description
QTID	Control block ID (halfword)
QTLEN	Control block length (halfword)
QTEYE	Control block eye catcher (4 characters)
QTALLOCA	Requests to allocate a bound plan for an agent. Represents the number of times DB2 was requested to create a thread by the attachment facility for the user. Does not include allocations for DB2 system agents.
QTALLOC	Number of successful bound plan allocations. Represents the number of allocation attempts identified by QTALLOCA that completed successfully.
QTABINDA	Number of times automatic bind was attempted. Occurs when the plan has been invalidated by modifications to the declarations of the data referenced by the programs bound as part of the plan.
QTABIND	Number of successful automatic binds, represents the number of automatic bind attempts (QTABINDA) that completed successfully.
QTINVRID	Requests to allocate a nonexistent plan ID. Represents the number of agent allocation attempts identified by QTALLOCA that completed unsuccessfully because the plan did not exist.
QTBINDA	Number of BIND ACTION (ADD) subcommands issued. The sum of QTBINDA, QTBINDR, and QTTESTB equals the total number of BIND subcommands.
QTBINDR	Number of BIND ACTION (REPLACE) subcommands issued. The sum of QTBINDA, QTBINDR, and QTTESTB equals the total number of BIND subcommands.
QTTESTB	Number of bind subcommands issued without a plan ID. The sum of QTBINDA, QTBINDR, and QTTESTB equals the total number of BIND subcommands.
QTPLNBD	Number of plans successfully bound and kept for future agent allocations. Represents the number of bind attempts identified by QTBINDA and QTBINDR that completed successfully. Counter does not increment for BIND subcommands with no plan ID specified, identified by QTTESTB.

Figure 133 (Part 1 of 2). Contents of IFCID 0002 Statistics Bind Data Section

Label	Description
QTREBIND	Number of REBIND subcommands issued.
QTRBINDA	Number of attempts to rebind an individual plan. This number may be larger than QTREBIND because multiple plan IDs can be specified on a single REBIND subcommand.
QTPLNRBD	Number of times a plan was successfully rebound. Represents the number of rebind attempts identified by QTRBINDA that completed successfully.
QTFREE	Number of FREE subcommands issued
QTFREEA	Number of attempts to free an individual plan. This number may be larger than QTFREE because multiple plan IDs can be specified on a single FREE subcommand.
QTPLNFRD	Number of times a plan was successfully freed. Represents the number of free attempts identified by QTFREEA that completed successfully.
QTAUCHK	Represents total number of authorization checks performed for a plan.
QTAUSUC	Represents total number of authorization checks (QTAUCHK) performed for a plan that were authorized.
QTDSOPN	Number of data sets currently open
QTMAXDS	Maximum number of data sets opened concurrently

Figure 133 (Part 2 of 2). Contents of IFCID 0002 Statistics Bind Data Section

Buffer Manager Data Section

The buffer manager data section can be found by using the contents of "QWS10R3O," "QWS10R3L," and "QWS10R3N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). This data section is stored in the SMF record as a repeating section of one or more items; one item is provided for each buffer pool that is currently in use, or has been in use since the DB2 subsystem was last started. To find the second item, add the length of the first item to the address of the first item, and so on. The length of each item is contained in "QWS10R3L." The number of items can be found in "QWS10R3N" in the self-defining section. After you have computed the location of the data section, you should establish its addressability using label "QBST."

All fields in this data section are fullword binary counts.

The labels to use when accessing the fields in the bind data section, and descriptions of the field contents, are given in Figure 134.

Label	Description
QBSTPID	Identifies the buffer pool for a particular occurrence. 0 for BP0, 1 for BP1, 2 for BP2 and 80 for BP32.
QBSTGET	Count of physical data base page requests. Data manager first access on behalf of a request or data manager access data on a page other than page being processed.
QBSTRIO	Media manager read requests. Result of a get page function from a page access request or a multiple page request by the buffer manager's prefetch function initiated by read from a utility (QBSTGET). Due to the data manager look aside capability, the value of QBSTRIO is always less than or equal to the count of page requests (QBSTGET). The ratio of QBSTRIO to QBSTGET should be as close to zero as possible.
QBSTBPX	Count of successful buffer pool expansions. A buffer is needed to access a data base page not in the buffer pool and all the buffers are non-reuseable. This helps identify applications which might function more efficiently with a larger buffer pool.
QBSTXFL	Count of buffer pool expansion failures when the pool was already at its maximum specified size. Increasing MAXPAGES can help prevent this problem.
QBSTXFV	Count of buffer pool expansion failures because of virtual storage shortage. Check data base services virtual storage for areas that can be reduced (for example, other buffer pools).
QBSTSWS	Count of updates performed against data base system pages. This counter is incremented by one each time a row in a system page is updated.
QBSTSWU	Count of unit of work pages updated.
QBSTPWS	Count of system pages written to DASD. The ratio of system page writes (PWS) to system page updates (SWS) suggests a higher level of efficiency as the ratio approaches 0. Factors that affect this ratio are: -Buffer pool size -Concurrent buffer pool usage by multiple transactions -Real storage availability -Data base page update of the same page by transactions
QBSTPWU	Count of unit of work (UW) pages written to DASD. Ratio of UW page writes (QBSTPWU) to UW page updates (QBSTSWU) should always approximate 1.
QBSTWIO	Count of write I/Os performed by the media manager. Batching of write I/O (for example, writing multiple pages per single call to media manager) should be as high as possible.
QBSTCBA	Instantaneous sample of the number of buffers in the buffer pool that were active (that is, in the non-reusable status) at the time of the request to transfer the buffer manager statistical data to the SMF record being produced.

Figure 134 (Part 1 of 2). Contents of IFCID 0002 Statistics Buffer Manager Data Section

Label	Description
QBSTRPI	Number of times a buffer being PAGEFIXed for read I/O does not have real storage frame backing. The ratio of reads with frames (QBSTRPI) required to read I/Os (QBSTRIO) increases efficiency as the ratio approaches 0. Factors which affect the ratio are: -Availability of real storage -Transaction rate -Buffer pool size
QBSTWPI	Number of times a buffer being PAGEFIXed to perform write I/O does not have real storage frame backing. PAGEFIX process must perform an I/O operation to retrieve the page's content from MVS DASD to a new real frame. Ratio of write page-ins (QBSTPWS) to pages written (QBSTPWU) indicates efficiency as it approaches 0.
QBSTDSO	This counter is incremented each time a data set assigned to the buffer pool is successfully opened.
QBSTIMW	Count of the number of immediate writes for system page.
QBSTSEQ	Count of the number of sequential prefetches requested.
QBSTSPP	Count of the pages read because of sequential prefetch requests.
QBSTSPD	Count of the sequential prefetch requests disabled because of an unavailable buffer resources.
QBSTREE	Count of the times that a sequential prefetch request is disabled because of an unavailable read engine.
QBSTWEE	Count of the times that a write engine is not available for I/O.
QBSTDWT	Count of the times that a deferred write threshold is reached.
QBSTDMC	Count of the times that a WPHFWT threshold is reached.

Figure 134 (Part 2 of 2). Contents of IFCID 0002 Statistics Buffer Manager Data Section

Lock Usage Data Section

The lock usage data section can be found by using the contents of "QWA01R4O," "QWA01R4L," and "QWA01R4N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the lock usage data section, you should establish its addressability using label "QTXA."

All fields in the lock usage data section are fullword binary values.

The labels to use when accessing the fields in the lock usage data section, and descriptions of the field contents, are given in Figure 135.

Label	Description
QTXADEA	Deadlock count
QTXASUS	Suspend count
QTXATIM	Timeout count
QTXALES	Count of lock escalations to shared mode.
QTXALEX	Count of lock escalations to exclusive mode.
QTXANPL	Maximum number of page locks held.

Figure 135. Contents of Accounting Record Lock Usage Data Section

Field Usage Notes:

- **Deadlock count ("QTXADEA"):** The deadlock count indicates the number of lock contentions between two or more agents. A change in the way DB2 resources are accessed by applications may reduce this contention (for example, with IMS/VS use class scheduling).
- **Suspend count ("QTXASUS"):** The suspend count indicates the number of times a lock could not be obtained and the agent was suspended. Suspensions are highly dependent on the application and table space locking protocols.
- **Timeout count ("QTXATIM"):** The timeout count indicates that a unit of work was suspended for a time that exceeded the timeout value. To minimize the number of timeouts, check locking protocols and the scheduling of work (including utility jobs).
- **Count of lock escalations to shared mode ("QTXALES") and Count of lock escalations to exclusive mode ("QTXALEX"):** These counts measure the total lock escalation activity of an application (accounting record) or of the subsystem (statistics record). These counts reflect the NUMLKTS installation parameter value versus the degree of page locking in the subsystem or an application. They can help you evaluate and adjust the NUMLKTS parameter and/or the LOCKSIZE parameter of a table space.
- **Maximum number of page locks held ("QTXANPL"):** This is a count of the maximum number of page locks concurrently held by a single application during its execution. This count corresponds to NUMLKUS install parameter value. It can help you evaluate and adjust NUMLKUS and/or help you select other locking parameters.

EDM Pool Usage Data Section

The EDM pool usage data section can be found by using the contents of "QWS10R6O," "QWS10R6L," and "QWS10R6N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the data section, you should establish its addressability, using label "QISE."

All count fields in this data section are fullword binary.

You can use these counts to help determine EDM pool usage and if the EDM pool is too small or too large.

Label	Description
QISEFAIL	Number of failures due to pool full (word)
QISEPAGE	Number of pages in EDM pool (word)
QISECTG	Number of request for cursor table sections (word)
QISECTL	Number of loaded cursor table sections (word)
QISECT	Number of pages used for cursor table (word)
QISEFREE	Number of free pages in free chain (word)
QISEDBD	Number of pages used for data base descriptors (DBD) (word)
QISESKCT	Number of pages used for skeleton cursor tables (word)
QISEDBDG	Number of requests for DBD (word)
QISEDBDL	Number of loads of DBD (word)

Figure 136. Contents of IFCID 0002 Statistics EDM Pool Usage Section

Note: Character-defined fields in data sections may contain hexadecimal data.

Accounting Record Data Sections: IFCID 0003

This section contains descriptions of the contents of the accounting record data sections. Be sure you have read "Data Sections" on page 276.

Product Data Section

The accounting record product data section always consists of one data item that is made up of two "headers." The first of these is called a "standard header," and also occurs in the product data sections of the IFCID 0001 and IFCID 0002 statistics records. The second is called a "correlation header" and occurs in the accounting record product data section.

Each of these headers contains its own length field. The sum of these two length fields is equal to the length of the entire data item, which is contained in field "QWA01PSL" in the self-defining section.

To examine the contents of the standard header (which begins at the first byte of the product data section), compute the address of the product data section, then establish addressability using label "QWHS."

Descriptions of the standard header fields, and the labels to use when addressing the fields, are given in Figure 121 on page 277.

The correlation header, found in the accounting record product data section, contains information about the connection to DB2. This connection information identifies the authorization ID that the accounting information contained in the other data sections applies to. The connection name indicates the subsystem connected to DB2; the correlation ID indicates the active agent (TSO job name, IMS/VS PST number and PSB, or CICS transaction) performing the work.

To examine the contents of the correlation header, compute its address by adding the length of the standard header (contained in "QWHSLEN") to the address of the product data section, then establish addressability using label "QWHC."

Descriptions of the correlation header fields, and the labels to use when addressing the fields, are given in Figure 137. Only significant fields are included; reserved and filler fields are omitted from the figure.

Label	Description
QWHCLEN	Length of correlation header (halfword binary)
QWHCTYP	Header type (1 byte, always equal to equate "QWHSHC02")
QWHCAID	Authorization ID (8 characters)
QWHCCV	Correlation ID value (12 characters)
QWHCCN	Connection name value (8 characters)
QWHCPLAN	Plan name (8 characters)
QWHCPREV	Reserved (8 characters)

Figure 137. Contents of Accounting Record Product Data Section Correlation Header

Figure 138 shows some of the values that the connection information in the correlation header can have, depending on the type of connection to DB2.

Field Label	Batch	TSO Invoked Interactively	IMS/VS	CICS
QWHCAID	User-ID on JOB statement	LOGON-ID	Message-driven regions: SIGNON-ID or LTERM Non-msg-driven regions: ASXBUSER if RACF used, otherwise PSB name.	USER-ID, TERM-ID, TRAN-ID, or other as specified in Resource Control Table (RCT)
QWHCCV	JOBNAME	LOGON-ID	PST#.PSBNAME	connection_type. thread_type. thread_#.tran-ID
QWHCCN	BATCH	TSO	ims-id	cics-id
QWHCPLAN	PLAN NAME	PLAN NAME	PLAN NAME	PLAN NAME

Figure 138. Connection Information for Different Types of Connection

Instrumentation Data Section

The instrumentation data section can be found by using the contents of "QWA01R1O," "QWA01R1L," and "QWA01R1N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255). After you have computed the location of the instrumentation data section, you should establish its addressability using label "QWAC."

If an accounting trace class 2 is not active, the values for fields QWCASC, QWACAJST, QWACASRB, and QWACARNA will be 0. Similarly, if an accounting trace class 3 is not active, the values for fields QWACAWTI, QWACAWTL, and QWACARNE will be 0.

The labels to use in accessing the instrumentation data section fields, and descriptions of the fields, are given in Figure 139 and Figure 140 on page 297.

Label	Description
QWACBSC	Beginning store clock value (8 characters)
QWACESC	Ending store clock value (8 characters)
QWACBJST	For IMS/VS and TSO, beginning job step ASCB TCB timer value; for CICS, low value CICS attach TCB time (8 characters)
QWACEJST	For IMS/VS and TSO, ending job step ASCB TCB timer value; for CICS, high value CICS attach TCB time (8 characters)
QWACBSRB	Beginning ASCB SRB timer value (8 characters)
QWACESRB	Ending ASCB SRB timer value (8 characters)
QWACRINV	Reason for accounting invocation (4 bytes, hexidecimal), see Figure 140 on page 297.
QWACNID	Network ID value (16 characters)
QWACCOMM	Number of commit phase two invocations (fullword binary)
QWACABRT	Number of abort requests (fullword binary)
QWACASC	Accumulated elapsed time in DB2 (8 characters)
QWACAJST	For IMS/VS, TSO, and connections through call attachment facility, accumulated ASCB TCB time in DB2; for CICS, accumulated DB2 CICS attach TCB time in DB2 (8 characters)
QWACASRB	Accumulated ASCB SRB time in DB2 (8 characters)
QWACAWTI	Accumulated I/O elapsed time in DB2 (8 characters)
QWACAWTL	Accumulated lock & latch elapsed time in DB2 (8 characters)
QWACARNA	Number of DB2 entry/exit events processed (fullword binary)
QWACARNE	Number of wait trace events processed (fullword binary)

Figure 139. Contents of Accounting Record Instrumentation Data Section

Value (dec.)	Reason Code Meaning
8	New user, authorization ID changed
12	Deallocation, normal program termination
16	End of task, application termination
20	End of task, application abend
24	End of memory, abnormal termination
28	Resolve INDOUBT
32	-STOP DB2 MODE (FORCE) command
40	End of task, application termination
44	End of task, application abend
48	End of memory, abnormal termination
52	Resolve INDOUBT
56	-STOP DB2 MODE (FORCE) command

Figure 140. Reason Code Meanings for Field QWACRINV

Field Usage Notes:

- **Beginning and ending store clock values ("QWACBSC," "QWACESC"):** You can use this data to determine the elapsed time of the application. Threads that do not terminate (such as, CICS primed threads and IMS/VS wait for input message regions) may have an ending clock value that includes the time the thread was inactive and waiting to perform work.
- **For IMS/VS and TSO, beginning and ending ASCB TCB timer values; For CICS, low and high value CICS attach TCB times ("QWACBJST," "QWACEJST"):** These values are for the agent address space for the time the authorization ID was considered connected.

  The authorization ID is considered connected from the time of successful allocation to the time the connection is terminated or a new user is successfully signed on. The first SQL call is normally considered the connection point. If the SQL call is at the beginning of the application, the accounting starts at that point and ends when the thread terminates or a new user is signed on. If the application abends, the accounting information is collected out of the end-of-task routine.
- **Beginning and ending ASCB SRB timer values ("QWACBSRB," "QWACESRB"):** These values are for the agent address space for the time that the authorization ID was considered connected.
- **Count of commit phase two invocations ("QWACCOMM"):** This count indicates the number of times that a unit of recovery successfully completed and the associated commit duration locks were released.
- **Count of abort invocations ("QWACABRT"):** This count indicates the number of times that a unit of recovery was backed out.
- **Accumulated elapsed time in DB2 ("QWACASC"):** This value indicates the elapsed store clock time a thread spent in DB2. When a call is received from an allied address space, the store clock value is saved and on exit from DB2 the ending store clock time is used to calculate the total time in DB2. The result is

added to the previous saved elapsed time in DB2. Note this field is only used if accounting classes 1 and 2 are active.

- **Accumulated TCB time in DB2 ("QWACAJST"):** This value indicates the allied agent TCB time a thread spent in DB2. When a call is received from an allied address space, the agent TCB time value is saved and, on exit from DB2, the ending TCB time is used to calculate the total agent TCB time in DB2. The result is added to the previous saved TCB time in DB2. For IMS/VS, TSO, and connections using the call attachment facility, the TCB time is obtained from the ASCB. For CICS, the time is obtained from the individual DB2 CICS attach TCB. Note this field is only used if accounting classes 1 and 2 are active.
- **Accumulated ASCB SRB time in DB2 ("QWACASRB"):** This value indicates the allied agent ASCB SRB time a thread spent in DB2. When a call is received from an allied address space, the agent ASCB SRB time value is saved and, on exit from DB2, the ending ASCB SRB time is used to calculate the total agent ASCB SRB time in DB2. The result is added to the previous saved ASCB SRB time in DB2. Note this field is only used if accounting classes 1 and 2 are active.
- **Accumulated I/O elapsed time in DB2 ("QWACAWTI"):** This value indicates the elapsed time the allied agent waited for I/O in DB2. When DB2 makes an I/O request for an allied agent, the store clock time is saved and, when the I/O completes, the ending time is used to calculate the total elapsed I/O time. The result is added to the previous saved elapsed I/O wait time in DB2. Note this field is only used if accounting classes 1 and 3 are active.
- **Accumulated lock & latch elapsed time in DB2 ("QWACAWTL"):** This value indicates the elapsed time the allied agent waited for locks and latches in DB2. When DB2 makes a lock or latch request that results in a wait for an allied agent, the store clock time is saved and, when the event completes, the ending time is used to calculate the total elapsed wait time. The result is added to the previous saved lock and latch wait time in DB2. Note this field is only used if accounting classes 1 and 3 are active.
- **Number of DB2 entry/exit events processed ("QWACARNA"):** This value indicates the number of entry and exit events processed to calculate the elapsed time in DB2 and the processor times. Note this field is only used if accounting classes 1 and 2 are active.
- **Number of wait events processed ("QWACARNE"):** This value indicates the number of entry and exit from wait events processed to calculate the I/O, lock, and latch elapsed wait times. Note this field is only used if accounting classes 1 and 3 are active.

SQL Statement Data Section

The SQL statement data section of IFCID 0003 accounting records is identical to the SQL statement data section of IFCID 0002 statistics records, as described in "SQL Statement Data Section" on page 287.

Buffer Manager Data Section

The buffer manager data section can be found by using the contents of "QWA01R3O," "QWA01R3L," and "QWA01R3N," which are located in the record's self-defining section (see "Self-Defining Section" on page 255).

One occurrence of this data section will be present for each buffer pool used; if all four buffer pools are used, there will be four occurrences of this data section (QWA01R3N = 4).

After you have computed the location of the buffer manager data section, you should establish its addressability using label "QBAC."

All fields in the buffer manager data section are fullword binary values.

The labels to use when accessing the fields in the buffer manager data section, and descriptions of the field contents, are given in Figure 141.

Label	Description
QBACPID	Pool ID: 0 for BP0, 1 for BP1, 2 for BP2, and 80 for BP32—identifies the buffer pool of a particular thread.
QBACGET	This counter is incremented by GET PAGE requests on a thread basis.
QBACBPX	This counter is incremented each time a GET PAGE is issued and causes a buffer pool expansion. An expansion can only occur when all the pool's buffers are in a non-reusable state. The thread that is expanded will remain in that state until a COMMIT or ABORT statement is issued. QBACBPX helps identify those applications which might function more efficiently if a larger buffer pool were provided.
QBACSWS	This counter is incremented every time a record residing in a system page is updated. The record may be accessed by multiple threads. It accounts for all updates, except for those done using certain utilities specifying a unit of work (UW) page write.
QBACSWU	This counter is incremented by one each time a unit of work (UW) is updated. The UW page is dedicated to a single thread which has exclusive control of the page.
QBACRIO	This counter is incremented for Media Manager synchronous read requests. Asynchronous I/O requests are not counted. In addition, this count varies for specific threads in accordance with DB2 activity, DB2 internal buffering algorithms, the number of concurrent users, and the size of the buffer pools being used. It may also vary across different DB2 releases.
QBACSEQ	This counter is incremented for Media Manager prefetch read requests. It indicates prefetch has been requested. The number of buffers actually read may be from 0 to 4 in Release 3.

Figure 141. Contents of Accounting Record Buffer Manager Data Section

Lock Usage Data Section

The lock usage data section of IFCID 0003 accounting records is identical to the lock usage data section of IFCID 0002 statistics records, described in "Lock Usage Data Section" on page 292, with one additional field located at the end of the section.

Maximum number of page locks held ("QTXANPL") is a count of the maximum number of page locks concurrently held by a single application during its execution. This count corresponds to the NUMLKUS installation parameter value. It can help you evaluate and adjust NUMLKUS and/or help you select other locking parameters.

Glossary

ABEND reason code. A 4-byte hexadecimal code that uniquely identifies a problem with DB2. A complete list of DB2 ABEND reason codes and their explanations is contained in *IBM DATABASE 2 Messages and Codes.*

abnormal end of task (ABEND). Termination of a task, a job, or a subsystem because of an error condition that cannot be resolved during execution by recovery facilities.

abort. The process of restoring recoverable data to a prior point of consistency by undoing the uncommitted changes made by a thread.

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 will fit on the active log.

address space connection. The result of connecting an allied address space to DB2. Each address space containing a task connected to DB2 has exactly one address space connection, even though more than one task control block (TCB) may be present. See *allied address space, task control block.*

agent. A representation within DB2 of a user of DB2 services. See also *allied agent* and *system agent.*

allied address space. An area of storage external to DB2 that is connected to DB2 and is therefore capable of requesting DB2 services.

allied agent. A representation within DB2 of a user of DB2 services that exists in an allied address space and is connected to DB2.

application. A program or set of programs that perform a task; for example, a payroll application.

application-embedded SQL. SQL statements coded within an application program.

application plan. The control structure produced during the bind process and used by DB2 to process SQL statements encountered during application execution.

archive log. The portion of the DB2 log that contains log records that have been moved from the active log because they no longer fit.

attachment facility. An interface between DB2 and TSO, IMS/VS, CICS, or batch address spaces. Attachment facility allows application programs to access DB2.

authorization ID. A name identifying a DB2 user.

authorized program facility. A facility that permits the identification of programs that are authorized to use restricted functions.

bind. The process by which the output from the DB2 precompiler is converted to a usable control structure called an application plan. This process is the one during which access paths to the data are selected and some authorization checking is performed.

automatic bind When an application program is being run and the bound application plan has been invalidated, binding takes place automatically, that is, without a user issuing a BIND command.

dynamic bind When SQL statements are entered through a terminal with SPUFI, binding is done dynamically (that is, as the SQL statements are entered).

bootstrap data set (BSDS). A VSAM data set that contains name and status information for DB2, as well as RBA range specifications, for all active and archive log data sets. It also contains passwords for the DB2 directory and catalog, and lists of conditional restart and checkpoint records.

buffer pool. In DB2, main storage reserved to satisfy the buffering requirements for one or more table spaces or indexes.

call attachment facility (CAF). A DB2 attachment facility for programs running in TSO or MVS batch that is an alternative to the DSN command processor and so allows greater control over the execution environment.

checkpoint. A point at which DB2 records internal status information on the DB2 log that would be used in the recovery process if DB2 should abnormally terminate.

CICS attachment facility. A DB2 subcomponent that uses the MVS Subsystem Interface (SSI) and cross storage linkage to process requests from CICS to DB2 and to coordinate resource commitment.

clustering index. An index that determines how rows are physically ordered in a table space.

column. The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer.)

command. In DB2, a DB2 operator command or a DSN subcommand. Distinct from an SQL statement.

command recognition character (CRC). A character that permits an MVS console operator or IMS/VS subsystem user to route DB2 commands to specific DB2 subsystems.

commit point. A point in time when data is considered to be consistent.

coordinator. The entity that determines whether a commit process is to complete or abort. See also *participant.*

correlation ID. An identifier associated with a specific thread. In TSO, it is either your user ID or the job name.

correlation name. An identifier that designates a table, a view, or individual rows of a table or view within a single SQL statement. It can be defined in any FROM clause, or in the first clause of an UPDATE or DELETE statement.

data base. A collection of table spaces and index spaces.

data base administrator (DBA). The person responsible for designing, implementing, and maintaining an operational data base.

data base descriptor (DBD). A DB2 control block that describes the tables and indexes of a data base and their external storage.

data type. An attribute of columns, literals, and host variables. For a list of data types, see *IBM DATABASE 2 SQL Reference.*

date. A three-part value that designates a day, month, and year.

DB2 catalog. DB2-maintained tables that contain descriptions of objects such as tables, views, and indexes.

DB2 command. An instruction to the DB2 subsystem allowing a user to start or stop DB2, to display information on current users, to start or stop data bases, to display information on the status of data bases, etc. DB2 commands are distinguished from other instructions by beginning with a hyphen (-).

DB2 Interactive (DB2I). The DB2 facility that provides for the execution of SQL statements, DB2 (operator) commands, programmer commands, and utility invocation.

deadlock. Unresolved contention for the use of a resource such as a table or index.

default value. A predetermined value, attribute, or option that is assumed when no other is explicitly specified. The default value of a column is the null value or a nonnull value determined by the data type of the column.

directory. The system data base that contains internal objects such as data base descriptors and skeleton cursor tables.

double byte character set (DBCS). A set of characters used by national languages such as Japanese and Chinese that have more symbols than can be represented by the 256 single byte EBCDIC positions. Each character is two bytes in length, and therefore requires special hardware to be displayed or printed.

DSN. (1) The default DB2 subsystem name; (2) the name of DB2's TSO command processor; (3) the first three characters of DB2 module and macro names.

DSN command processor. The DB2 component that processes DB2 subcommands (such as BIND, RUN, etc).

exit routine. An installation-written (or IBM-provided default) program that receives control from DB2 to allow the installation to perform certain specific functions. Exit routines run as extensions of DB2 (for example, userid exit DSN3@ATH, which enables the installation to override userid information).

fall back. The process of returning to a previously installed prior release subsystem after attempting or completing migration to the new release.

FREE. The DSN subcommand used to delete an application from DB2, as well as all dependencies that plan may have on other structures. The application plan name is then available for use in a BIND subcommand.

function. Same as "Built-in Function": a scalar function or column function.

generalized trace facility (GTF). An MVS service program that records significant system events such as I/O interrupts, SVC interrupts, program interrupts, or external interrupts.

GIMSMP. The load module name for the System Modification Program/Extended, a basic tool for installing, changing, and controlling changes to programming systems.

help panel. A screen of information presenting tutorial text to assist the terminal user.

HMASMP. The load module name for the System Modification Program (SMP), a basic tool for installing, changing, and controlling changes to programming systems.

home address space. The area of storage that MVS currently recognizes as "dispatched."

host program. A program written in a host language that contains embedded SQL statements.

host structure. In an application program, a structure referenced by embedded SQL statements.

host variable. In an application program, a variable referenced by embedded SQL statements.

IDCAMS LISTCAT. A facility for obtaining information contained in the Access Method Services catalog.

identify. A request that an attachment service program in an address space separate from DB2 issues via the MVS subsystem interface to inform DB2 of its existence and initiate the process of becoming connected to DB2.

image copy. An exact reproduction of all or part of a table space. DB2 provides utility programs to make full image copies (to copy the entire table space), or incremental image copies (to copy only those pages that have been modified since the last image copy).

IMS/VS attachment facility. A DB2 subcomponent that uses MVS Subsystem Interface (SSI) protocols and cross-storage linkage to process requests from IMS/VS to DB2 and to coordinate resource commitment.

IMS/VS resource lock manager (IRLM). An MVS subsystem used by DB2 to control communication and data base locking.

index. A set of pointers that are logically ordered by the values of a key. Indexes provide quick access to data and can enforce uniqueness on the rows in a table.

install. The process of preparing a DB2 Release 3 subsystem to operate as an MVS subsystem. It is intended for those who are not DB2 Release 2 users. Release 2 users may choose, however, to install a Release 3 test subsystem prior to migrating their Release 2 production subsystem.

latch. A DB2 internal mechanism for serializing events or the use of system resources.

link-edit. To create a loadable computer program by means of a linkage editor.

link pack area map (LPAMAP). A statement that can be specified on the AMDPRDMP program, which provides a listing of the load modules in the link pack area list. It identifies the entry point address of those load modules and their length.

load module. A program unit that is suitable for loading into main storage for execution; it is the output of a linkage editor.

lock. A means of serializing events or access to data. DB2 locking is performed by the IRLM.

lock attributes. The characteristics of a DB2 lock. Lock attributes include: object, size, mode, and duration.

lock duration. The DB2 lock attribute that specifies how long a table space is locked. Possible lock durations include: allocation to deallocation, use to commit, use to deallocation, or manual.

lock escalation. The promotion of a lock from a page lock to a table space lock because the number of page locks concurrently held on a given resource exceeds a preset limit.

locking. The process by which DB2 ensures integrity of data. Locking prevents concurrent users from accessing inconsistent data. Lock sizes are at either the table space or page level.

lock mode. The DB2 lock attribute that specifies the type of access concurrently running programs can have to a resource that has been locked by a program. A resource can be locked at either the page or table space level, in either shared or exclusive mode. A table space can also be locked in a qualified mode that permits concurrent programs to access the table space to perform specific operations.

lock object. The DB2 lock attribute that refers to the type of resource to be locked. In DB2, the lock objects include: table spaces, pages and index subpages.

lock promotion. The process of changing the size or mode attribute, or both, of a DB2 lock to a higher level. DB2 will promote a lock when: concurrent access requests for the same data resource have conflicting attributes, or the number of page locks concurrently held on a given resource exceeds a preset limit. When a lock is promoted because the number of page locks exceeds preset limits, the process is referred to as "lock escalation."

log. A collection of records that describe the events that occur during DB2 execution and their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

log initialization. The first phase of restart processing, during which DB2 attempts to locate the current end of the log.

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 will be written.

menu. A displayed list of available, logically grouped functions for selection by the operator. Sometimes called a menu panel.

migration. The process of converting a DB2 Release 2 subsystem to a DB2 Release 3 subsystem. It allows you to acquire the functions of Release 3 without losing the data you created on Release 2. You can perform a migration only if you already have Release 2 installed.

must-abort. A state from which a user may only abort.

must-complete. A state during DB2 processing in which the entire operation must be completed to maintain data integrity.

null. A special value that indicates the absence of information.

object. Anything that can be created or manipulated with SQL—that is, data bases, table spaces, tables, views, or indexes.

page. A unit of storage within a table space (4K or 32K bytes) or index space (4K bytes). In a table space, a page contains one or more rows of a table.

page set. One or more data sets used by DB2 for storing data objects such as tables or indexes. See also *simple page set* and *partitioned page set.*

panel. In computer graphics, a predefined display image that defines the locations and characteristics of display fields on a display surface: sometimes called a menu or menu panel.

participant. An entity other than the commit coordinator that takes part in the commit process.

partition. A portion of a page set. Each partition corresponds to a single, independently extendable data set. Partitions can be extended to a maximum size of 1, 2, or 4 gigabytes, depending upon the number of partitions in the partitioned page set. All partitions of a given page set have the same maximum size.

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data. Synonymous with program library.

partitioned page set. A page set that is divided into a number of partitions, where the number of partitions can be from 1 to 64 and is specified by the creator of the page set. The partition of a partitioned page set in which data is stored is determined by the value of the clustering index key associated with the data.

partitioned table space. A table space subdivided into parts (based upon index key range), each of which may be reorganized independently of the others.

plan. See *application plan.*

plan name. The name of an application plan. A DB2 plan is the output from the bind process.

privilege. A capability given to a user by the execution of a GRANT statement.

program temporary fix (PTF). A solution or bypass of a problem diagnosed as a result of a defect in a current unaltered release of a licensed program. An authorized program analysis report (APAR) fix is corrective service for an existing problem; a PTF is preventive service for problems that might be encountered by other users of the product. A PTF is "temporary" because it is usually incorporated in the next release of the product.

rebind. Create a new application plan for a program that has previously been bound. If, for example, you have added an index for a table accessed by your application, you must rebind the application in order for it to be able to take advantage of that index.

record. The storage representation of a row or other data.

recovery. The process of rebuilding data bases after a system failure.

recovery log. A collection of records that describe the events that occur during DB2 execution and their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

redo. One of the possible states of a unit of recovery, in which the changes made are to be reapplied to the DASD media to ensure data integrity.

region control task (RCT). The control program routine that handles quiesce/restore and LOGON/LOGOFF.

remigration. The process of returning to DB2 Release 3 following a fall back to Release 2. This procedure constitutes another migration process.

resource control table (RCT). A construct of the CICS attachment facility, created via installation-provided macro parameters, that defines authorization and access attributes on a per-transaction or per-transaction-group basis.

scalar function. An SQL operation that produces a single value from another value and is expressed in the form of a function name followed by a list of arguments enclosed in parentheses.

sequential data set. A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 data base utilities require sequential data sets.

signon. A request made on behalf of an individual CICS or IMS/VS user by an attach facility to enable DB2 to verify that user's authorization to use DB2 resources.

simple table space. An unpartitioned table space. A simple table space can contain more than one table.

SPUFI. SQL Processor Using File Input. A facility of the TSO attachment subcomponent that enables the DB2I user to execute SQL statements without embedding them in an application program.

SQL. Structured Query Language. A language that can be used within COBOL, PL/I, APL2, FORTRAN, IBM BASIC, and assembler application programs, or interactively to access DB2 data and to control access to DB2 resources.

SQL Communication Area (SQLCA). A collection of variables that are used by DB2 to provide an application program with information about the execution of its SQL statements.

SQL Descriptor Area (SQLDA). A collection of variables that are used in the execution of certain SQL statements. The SQLDA is intended for dynamic SQL programs.

static SQL. SQL statements that are embedded within a program, and are prepared during the program preparation process before the program is executed. After being prepared, the statement itself does not change (although values of host variables specified by the statement might change).

storage group. A named set of DASD volumes on which DB2 data can be stored.

subcomponent. A group of closely related DB2 modules that work together to provide a general function.

subpage. The unit into which a physical index page can be divided. Each index page can be divided into a number of subpages; the larger the number of subpages, the greater the potential for concurrent use of an index by multiple applications because the subpage is the unit on which index locking is performed.

subsystem recognition character (SRC). A character prefixed to DB2 commands and used by MVS to identify DB2 subsystem commands so that they can be routed to the correct DB2 subsystem.

sync point. See *commit point.*

synonym. In SQL, an alternative name for a DB2 table or view.

system administrator. The person having the second highest level of authority within DB2. System administrators make decisions about how DB2 is to be used and implement those decisions by choosing system parameters. They monitor the system and change its characteristics to meet changing requirements and new data processing goals.

system agent. An agent that is created for DB2 internal use. See also *agent.*

system diagnostic work area (SDWA). The data that is recorded in a SYS1.LOGREC entry that describes a program or hardware error.

System Modification Program (SMP). A tool for making software changes in programming systems (such as DB2 or MVS), and for controlling those changes.

SYS1.DUMPxx data set. A data set that contains a system dump.

SYS1.LOGREC. A service aid that contains important information about program and hardware errors.

table. A named data object consisting of a specific number of columns and some number of unordered rows.

table space. A page set used to store the records of one or more tables.

task control block (TCB). A control block used to communicate information about tasks within an address space that are connected to DB2. An address space can support many task connections (as many as one per task), but only one address space connection. See **address space connection.**

thread. The DB2 structure that describes an application's connection existence, traces its progress, provides resource function processing capability, and delimits its accessibility to DB2 resources and services. Most DB2 functions execute under a thread structure.

time. A three-part value that designates a time of day in hours, minutes, and seconds.

timestamp. A seven-part value that consists of a date, a time, and a number of microseconds.

to-do. A state of a unit of recovery that indicates that the unit of recovery's changes to recoverable DB2 resources are indoubt and must either be applied to the DASD media or backed out, as determined by the commit coordinator.

trace. A DB2 tool that allows the user to monitor and collect DB2 performance, accounting, statistics, and serviceability (global) data.

TSO attachment facility. A DB2 facility consisting of the DSN command processor and DB2I. Applications that aren't written for the CICS or IMS/VS environments can run under the TSO attachment facility.

undo. A state of a unit of recovery that indicates that the unit of recovery's changes to recoverable DB2 resources must be backed out.

unique index. An index that assures that no identical key values are stored in a table.

unit of recovery. A sequence of operations within a unit of work between points of consistency.

unlock. To release an object or system resource that was previously locked and return it to general availability within DB2.

value. Smallest unit of data manipulated in SQL.

view. An alternative representation of data from one or more tables. A view can include all or some of the

columns contained in the table or tables on which it is defined.

Acronyms and Abbreviations

ABEND	abnormal end of task
AMODE	address mode
APAR	authorized program analysis report
APF	authorized program facility
auth-ID	authorization ID
BSAM	Basic Sequential Access Method
BSDS	bootstrap data set
CAF	Call Attachment Facility
CICS	Represents (in this publication) CICS/OS/VS and CICS/MVS
CICS/MVS	Customer Information Control System/Multiple Virtual Storage
CICS/OS/VS	Customer Information Control System/Operating System/Virtual Storage
CLIST	command list
CSA	MVS common service area
CSECT	control section
DASD	direct access storage device
DBA	data base administrator
DBCS	double byte character set
DBD	data base descriptor
DBID	data base identifier
DBMS	data base management system
DDL	data definition language
DFHSM	Data Facility Hierarchical Storage Manager
DFP	Data Facility Product
DML	data manipulation language
DXT	Data Extract
EDM	data manager environmental descriptor management function
EDMPOOL	environmental descriptor manager pool
EID	event identifier
EOM	end of memory
EOT	end of task
ERLY	early processing block
ESDS	entry sequenced data set
ESMT	external subsystem module table (IMS/VS)
ESRC	EDM SKCT hash record
EUR	IBM European Standards
FMID	field maintenance identifier

GTF	Generalized Trace Facility
ICF	Integrated Catalog Facility
IMS/VS	Information Management System/Virtual Storage
IPL	initial program load
IRLM	IMS/VS Resource Lock Manager
ISO	International Standards Organization
ISPF	Interactive System Productivity Facility
ISPF/PDF	Interactive System Productivity Facility/Program Development Facility
JCL	job control language
JIS	Japanese Industrial Standard
K (bytes)	1024 (bytes)
MSS	Mass Storage Subsystem
MVS	Multiple Virtual Storage
MVS/XA	Multiple Virtual Storage/Extended Architecture
OBID	data object identifier
PCT	program control table (CICS)
PLT	program list table (CICS)
PTF	program temporary fix
QMF	Query Management Facility
QSAM	Queued Sequential Access Method
RACF	OS/VS2 MVS Resource Access Control Facility
RBA	relative byte address
RCT	region control task resource control table (CICS attachment facility)
RID	record identifier
SMF	System Management Facility
SMP	System Modification Program
SMP/E	System Modification Program/Extended
SPUFI	SQL Processor Using File Input
SRC	subsystem recognition character
TMP	Terminal Monitor Program
TSO	MVS Time Sharing Option
VSAM	Virtual Storage Access Method
VTAM	MVS Virtual Telecommunication Access Method
WTO	write to operator
WTOR	write to operator with reply
XRF	extended recovery facility

Bibliography

- *CICS/MVS Installation Guide*, SC33-0506
- *CICS/MVS Resource Definition (Macro)*, SC33-0509
- *CICS/MVS Resource Definition (Online)*, SC33-0508
- *Customer Information Control System/Operating System/Virtual Storage (CICS/OS/VS): Installation and Operations*, SC33-0071
- *CICS/OS/VS Resource Definition (Online)*, SC33-0186
- *CICS/OS/VS Resource Definition (Macro)*, SC33-0237
- *DATABASE 2 Performance Monitor General Information*, GH20-6856
- *DATABASE 2 Performance Monitor User's Guide*, SH20-6857
- *Data Facility Hierarchical Storage Manager User's Guide*, SH35-0093
- *DFSORT Application Programming: Guide*, SC33-4035
- *IBM DATABASE 2 Application Programming Guide*, SC26-4293
- *IBM DATABASE 2 Advanced Application Programming Guide*, SC26-4292
- *IBM DATABASE 2 Command and Utility Reference*
- *IBM DATABASE 2 Data Base Planning and Administration Guide*, SC26-4077
- *IBM DATABASE 2 Diagnosis Guide*, LY26-3850
- *IBM DATABASE 2 Diagnosis Reference Volume 1*, LY26-3862
- *IBM DATABASE 2 Diagnosis Reference Volume 2: Data Area Descriptions*, LY26-3863
- *IBM DATABASE 2 Diagnosis Reference Volume 3: Advanced Techniques*, LY26-3952
- *IBM DATABASE 2 Diagnosis Reference Volume 4: Service Traces*
- *IBM DATABASE 2 General Information*
- *IBM DATABASE 2 Operation and Recovery Guide*, SC26-4083
- *IBM DATABASE 2 Messages and Codes*, SC26-4113
- *IBM DATABASE 2 Program Directory*
- *IBM DATABASE 2 Publication Guide and Master Index*, GC26-4111
- *IBM DATABASE 2 Reference Summary*, SX26-3740
- *IBM DATABASE 2 Install Guide*, SC26-4084
- *IBM DATABASE 2 SQL Reference*
- *IMS/VS Version 2 Installation Guide*, SC26-4172
- *IMS/VS Version 2 Operations and Recovery Guide*, SC26-4183
- *ISPF Version 2 for MVS Dialog Management Services*, SC34-4021
- *ISPF and ISPF/PDF Version 2 for MVS Installation and Customization*, SC34-4019
- *ISPF/PDF Version 2 for MVS Reference*, SC34-4024
- *MVS/Extended Architecture Data Administration Utilities, Volume 1*, GC26-4018
- *MVS/Extended Architecture Data Administration Utilities, Volume 2*, GC26-4150
- *MVS/Extended Architecture Integrated Catalog Administration: Access Method Services Reference*, GC26-4135

- *MVS/Extended Architecture JCL User's Guide*, GC28-1351
- *MVS/Extended Architecture System Programming Library JES2 Initialization and Tuning Guide*, SC23-0065
- *MVS/Extended Architecture System Programming Library JES3 Initialization and Tuning Guide*, SC23-0059
- *MVS/Extended Architecture Message Library: Routing and Descriptor Codes*, GC28-1194
- *MVS/Extended Architecture Message Library: System Messages*, GC28-1376 through GC28-1377
- *MVS/Extended Architecture Data Administration Guide*, GC26-4140
- *MVS/Extended Architecture System Programming Library: Initialization and Tuning*, GC28-1149
- *MVS/Extended Architecture System Programming Library: Supervisor Services and Macro Instructions*, GC28-1154
- *MVS/370 System Programming Library: System Macros and Facilities Volume 2*, GC28-1153
- *MVS/Extended Architecture System Programming Library: System Modifications*, GC28-1152
- *MVS/Extended Architecture System Programming Library: TSO*, GC28-1173
- *MVS/Extended Architecture VSAM Catalog Administration: Access Method Services Reference*, GC26-4136
- *MVS/Extended Architecture Message Library: Routing and Descriptor Codes*, GC28-1194
- *MVS/Extended Architecture System Programming Library: System Macros and Facilities, Volume 2*, GC28-1151
- *MVS/370 System Programming Library: System Macros and Facilities Volume 2*, GC28-1153
- *MVS/Extended Architecture System Programming Library: System Modifications*, GC28-1152
- *MVS/370 JCL*, GC28-1300
- *MVS/370 Routing and Descriptor Codes*, GC38-1102
- *Initialization and Tuning Guide OS/VS2 SPL*, GC28-1029
- *MVS/370 System Management Facilities (SMF)*, GC28-1030
- *MVS/Extended Architecture Integrated Catalog Administration: Access Method Services Reference*, GC26-4135
- *OS/VS2 MVS Supervisor Services and Macros*, GC28-0683
- *MVS/Extended Architecture System-Data Administration*, GC26-4149
- *Resource Access Control Facility (RACF) Auditor's Guide*, SC28-1342
- *Resource Access Control Facility (RACF) Command Language Reference*, SC28-0733
- *Resource Access Control Facility (RACF) Security Administrator's Guide*, SC28-1340
- *Resource Access Control Facility (RACF) General Information Manual*, SC28-0722
- *System Modification Program (SMP) Messages and Codes*, GC38-1047
- *System Modification Program (SMP) System Programmers Guide*, GC28-0673
- *System Modification Program Extended (SMP/E) General Information Manual*, GC28-1106
- *System Modification Program Extended (SMP/E) User's Guide*, SC28-1302
- *System Programming Library: Resource Access Control Facility (RACF)*, SC28-1343
- *TSO Extensions Command Language Reference*, SC28-1370

Index

Special Characters

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

X

Reader's Comment Form

IBM DATABASE 2
System Planning
and Administration Guide
SC26-4085-3

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Do not use this form to request IBM publications. If you do, your order will be delayed because publications are not stocked at the address printed on the reverse side. Instead, you should direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

If you have applied any technical newsletters (TNLs) to this book, please list them here: _____

Chapter/Section _____

_____ Page No. _____

Comments:

If you want a reply, please complete the following information.

Name _____ Phone No. (____)_____

Company _____

Address _____

_____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC26-4085-3

Reader's Comment Form

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and tape | Please do not staple | Fold and tape

IBM®

Reader's Comment Form

IBM DATABASE 2
System Planning
and Administration Guide
SC26-4085-3

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Do not use this form to request IBM publications. If you do, your order will be delayed because publications are not stocked at the address printed on the reverse side. Instead, you should direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

If you have applied any technical newsletters (TNLs) to this book, please list them here: _____

Chapter/Section _____

_____ Page No. _____

Comments:

If you want a reply, please complete the following information.

Name _____ Phone No. (_____)_____

Company _____

Address _____

_____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Reader's Comment Form

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and tape | Please do not staple | Fold and tape

IBM®

Reader's Comment Form

IBM DATABASE 2
System Planning
and Administration Guide
SC26-4085-3

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Do not use this form to request IBM publications. If you do, your order will be delayed because publications are not stocked at the address printed on the reverse side. Instead, you should direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

If you have applied any technical newsletters (TNLs) to this book, please list them here: _____

Chapter/Section _____

_____ Page No. _____

Comments:

If you want a reply, please complete the following information.

Name _____ Phone No. (____)_____

Company _____

Address _____

_____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC26-4085-3

Reader's Comment Form

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and tape | Please do not staple | Fold and tape

IBM®