DB2 Universal Database for z/OS

Version 8

Installation Guide

GC18-7418-04

DB2® DB2 Universal Database for z/OS

Version 8

Installation Guide

GC18-7418-04

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 545.

Fifth Edition, Softcopy Only (February 2006)

This edition applies to Version 8 of IBM DB2 Universal Database for z/OS (DB2 UDB for z/OS), product number 5625-DB2, and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

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

This and other books in the DB2 for z/OS library are periodically updated with technical changes. These updates are made available to licensees of the product on CD-ROM and on the Web (currently at www.ibm.com/software/data/db2/zos/library.html). Check these resources to ensure that you are using the most current information.

Contents

About this book

This section contains specific information about this book and a general overview of the DB2 Universal Database for z/OS library.

Important

In this version of DB2 UDB for z/OS, the DB2 Utilities Suite is available as an optional product. You must separately order and purchase a license to such utilities, and discussion of those utility functions in this publication is not intended to otherwise imply that you have a license to them. See Appendix D, “DB2 utilities packaging,” on page 539 for packaging details.

Who should read this book

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

msys for Setup DB2 Customization Center is a feature of DB2 that provides a graphical user interface for customizing DB2 UDB for z/OS from the workstation. msys for Setup DB2 Customization Center provides an alternative to the existing ISPF installation panels and CLISTs. msys for Setup DB2 Customization Center lets you install, migrate, and update your DB2 subsystem.

This book assumes that you are familiar with:

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

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

Terminology and citations

In this information, DB2 Universal Database™ for z/OS® is referred to as "DB2 UDB for z/OS." In cases where the context makes the meaning clear, DB2 UDB for z/OS is referred to as "DB2." When this information refers to titles of books in this library, a short title is used. (For example, "See *DB2 SQL Reference*" is a citation to *IBM® DB2 Universal Database for z/OS SQL Reference*.)

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

The following terms are used as indicated:

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

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

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

CICS® Represents CICS Transaction Server for z/OS or CICS Transaction Server for OS/390®.

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

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

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

How to read the syntax diagrams

The following rules apply to the syntax diagrams that are used in this book:

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

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

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

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

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

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

- Optional items appear below the main path.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

  A repeat arrow above a stack indicates that you can repeat the items in the stack.
- Keywords appear in uppercase (for example, `FROM`). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, *column-name*). They represent user-supplied names or values.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products. The major accessibility features in z/OS products, including DB2 UDB for z/OS, enable users to:

- Use assistive technologies such as screen reader and screen magnifier software
- Operate specific or equivalent features by using only a keyboard
- Customize display attributes such as color, contrast, and font size

Assistive technology products, such as screen readers, function with the DB2 UDB for z/OS user interfaces. Consult the documentation for the assistive technology products for specific information when you use assistive technology to access these interfaces.

Online documentation for Version 8 of DB2 UDB for z/OS is available in the Information management software for z/OS solutions information center, which is an accessible format when used with assistive technologies such as screen reader or screen magnifier software. The Information management software for z/OS solutions information center is available at the following Web site: http://publib.boulder.ibm.com/infocenter/dzichelp

How to send your comments

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

- Send your comments by e-mail to db2pubs@vnet.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title, page number, or a help topic title).
- You can also send comments from the Web. Visit the library Web site at:

  www.ibm.com/software/db2zos/library.html

  This Web site has a feedback page that you can use to send comments.
- Print and fill out the reader comment form located at the back of this book. You can give the completed form to your local IBM branch office or IBM representative, or you can send it to the address printed on the reader comment form.

Summary of changes to this book

This section summarizes the changes that have been made to this book.

Section 2. Planning and installing DB2 has the following changes:

- Chapter 1, "Introduction to installation, migration, and conversion" contains updated summaries of installation, migrating to Version 8 compatibility mode, and conversion to Version 8 new-function mode.
- Chapter 2, "Estimating DB2 storage needs" provides updated calculations to help you estimate storage for the DB2 subsystem.
- Chapter 5, "msys for Setup DB2 Customization Center" describes how to use the DB2 Customization Center to install and migrate your DB2 subsystem.
- Chapter 6, "Installing, migrating, and updating system parameters" describes new panel fields to support IRLM, thread management, application programming defaults, and other enhancements.
- Chapter 7, "Installing the DB2 subsystem" provides updated information about the installation process.
- Chapter 8, "Migrating the DB2 subsystem to compatibility mode" describes migration considerations and release incompatibilities for Version 8, and describes the migration process.
- Chapter 10, "Verifying installation with the sample applications" includes information on five new sample jobs.

Section 3. Communicating with other systems has the following changes:

- Chapter 12, "Connecting distributed database systems" has minor changes for Version 8 enhancements.
- Chapter 13, "Connecting systems with VTAM" has minor changes for LU names.
- Chapter 14, "Connecting systems with TCP/IP" has minor changes for LU names and general changes to TCP/IP.

The appendixes have the following changes:

- Appendix A, "Character conversion" contains updated information about converting to the euro symbol.
- Appendix B, "CATENFM and CATMAINT" contains information about the utilities CATENFM and CATMAINT.
- Editing the subsystem parameters and DSNHDECP values contains new parameters.

Part 1. Introduction

Chapter 1. Introduction to installation, migration, and conversion

This chapter introduces you to the process of installing, migrating, and converting to DB2 Version 8. Subsequent chapters of the book provide detailed instructions.

Installation is the process of preparing DB2 to operate as a z/OS subsystem. *Migration* is the process of upgrading from Version 7 to Version 8 compatibility mode. *Conversion* is a process that is comprised of three progressive catalog levels: *compatibility mode*, *enabling-new-function mode*, and *new-function mode*.

Compatibility mode
The state of the catalog after the Version 8 migration process is complete. New Version 8 functions are not available for use in compatibility mode. You can fall back to Version 7 from compatibility mode.

Enabling-new-function mode
The process of converting the catalog to Unicode. New Version 8 functions are not available in enabling-new-function mode. You cannot return to Version 8 compatibility mode. You cannot fall back to Version 7 from enabling-new-function mode.

New-function mode
The state of the catalog after conversion is complete. The catalog has been marked as being in new-function mode. All new Version 8 functions are available in new-function mode. You can return to Version 8 enabling new-function mode, but you cannot return to Version 8 compatibility mode. You cannot fall back to Version 7 from new-function mode.

You can migrate to DB2 UDB for z/OS Version 8 compatibility mode only from DB2 Version 7. The process of migrating to Version 8 and using the new function provided in Version 8 has changed. You should read this chapter, Chapter 8, "Migrating the DB2 subsystem to compatibility mode," and Chapter 9, "Enabling new-function mode" for information on migrating your existing subsystem to Version 8 new-function mode.

You must migrate to a z/OS Version 1 Release 3 or later environment before installing DB2 Version 8. DB2 for z/OS Version 8 operates on any processor that supports 64-bit z/Architecture™, including z800 and z900 or comparable processor. See *DB2 Program Directory* for more information about system requirements.

Before you begin installing or migrating, plan the amount of direct-access storage and virtual storage that you need. Chapter 2, "Estimating DB2 storage needs," on page 17 helps you with your decisions. Planning and coordinating with other DB2 subsystems is essential if you plan to install the distributed data facility (DDF). For more information, see Part 2 (Volume 1) of *DB2 Administration Guide*. Review what values are needed for the parameters on the installation and migration panels. By planning in advance, your task of filling in the parameters becomes easier. See "Running the installation CLIST" on page 80 for help with your decisions.

The following topics provide additional information:
- "Installing other features of the DB2 Universal Database for z/OS" on page 4
- "Installation tools" on page 4

- "Overview of installation and migration steps" on page 5
- "Planning for migration incompatibilities" on page 5
- "Summary of SMP/E steps" on page 6
- "Summary of installation steps" on page 6
- "Summary of migration to compatibility mode steps" on page 7
- "Summary of fallback steps" on page 8
- "Summary of remigration steps" on page 9
- "Summary of conversion to new-function mode steps" on page 9

Installing other features of the DB2 Universal Database for z/OS

For more information about installing the features of DB2 Universal Database for z/OS, see the Program Directories for the individual features.

Installation tools

DB2 includes several tools and capabilities to help you perform the steps that are involved in installing or migrating to Version 8.

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

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

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

  This CLIST is also called the *migration CLIST*, or simply the *CLIST*. It contains the necessary code for tailoring the jobs to suit your needs.
- A series of ISPF panels that you can use to pass information to the CLIST

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

  DB2 provides a set of sample programs and procedures that help you determine if DB2 is operating correctly.
- | mSys for Setup DB2 Customization Center provides a graphical user interface
  | for installing, migrating, converting, and updating DB2.

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

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

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

Ability to update installation and migration options: During the process of installing and migrating, DB2 uses ISPF panels to prompt you for many options. DB2 allows you to update most of these options without requiring you to reinstall or remigrate. You can accept the default values for certain options and, after acquiring experience with DB2, tailor them to suit your needs. A complete list of all parameters is available in Appendix C, "Directory of subsystem parameters," on page 533.

Overview of installation and migration steps

Whether you are installing or migrating, you need to perform the following procedures:

1. Estimate storage needs.
2. Determine which new functions you need.
3. If using distributed data, install VTAM and, optionally, TCP/IP.
4. Set up a Parallel Sysplex® if you plan to use data sharing (see *Parallel Sysplex Configuration Assistant* for information about setting up a Sysplex and *DB2 Data Sharing: Planning and Administration* for information on setting up a data sharing environment).
5. Load the DB2 libraries (do the SMP/E steps).

   If you plan to use the callable SQL interface of DB2, see *DB2 ODBC Guide and Reference* for the additional installation jobs that you need to run.

   If you plan to use DB2 UDB for z/OS Java™ Edition, see *DB2 Application Programming Guide and Reference for Java* for additional installation jobs that you need to run.
6. Install needed service on the prior release (if you are migrating). See *DB2 Program Directory* for information about needed service.
7. Check release incompatibilities, and make the necessary changes in your applications.
8. Tailor the installation or migration jobs.
9. Install or migrate DB2.
10. Connect the DB2 attachment facilities.
11. Prepare DB2 for use.
12. Verify installation or migration.

If you have problems during or after migration to Version 8 compatibility mode, you can perform the following procedures:

1. Fall back to Version 7.
2. Remigrate to DB2 UDB for z/OS Version 8 compatibility mode.

Planning for migration incompatibilities

Be aware that enhancements to this DB2 release might cause unexpected results from your system utilities, applications, or jobs. Proper planning should alleviate any system inconveniences. See "Make adjustments for release incompatibilities" on page 317 for more information.

Summary of SMP/E steps

Before you begin installing or migrating to DB2, you must unload the DB2 tapes or cartridges. Then, you edit and run SMP/E jobs. Table 1 identifies the SMP/E steps that you need to perform.

Before proceeding with these steps, refer to *IBM DB2 Program Directory*, which is shipped with DB2, for keyword specifications for Preventive Service Planning (PSP). Use Information/Access or the ServiceLink facility of IBMLink to check the most current information about DB2 and other products. Contact IBM Software Support if you do not have access to IBMLink.

Table 1. Overview of SMP/E steps

Step	Description	Job
1	Copy and edit the SMP/E jobs.	IEBCOPY
2	Optionally, initialize the SMP/E environment for DB2.	DSNTIJAA
3	Allocate the libraries.	DSNALLOC
4	Run the RECEIVE jobs.	DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4
5	Optionally, run the clean-up job.	DSNTIJUD
6	Run the APPLY jobs.	DSNAPPL1, DSNAPPL2
7	Run the ACCEPT jobs.	DSNACEP1, DSNACEP2
8	Optionally, unload the jobs for the additional FMIDs.	(none)
9	Optionally, run the RECEIVE jobs for the additional FMIDs.	DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4
10	Optionally, run the APPLY jobs for the additional FMIDs.	DSNAPPL1, DSNAPPL2
11	Optionally, run the ACCEPT jobs for the additional FMIDs.	DSNACEP1, DSNACEP2
12	Receive and apply any maintenance that is shipped with the product.	(none)

Summary of installation steps

After you perform the SMP/E steps and follow the steps on page 81 to run the installation CLIST, you can edit and run the jobs that install your DB2 Version 8 subsystem. A new installation starts in new-function mode. Table 2 identifies the steps that you perform to install DB2 Version 8.

Table 2. Overview of steps for installing DB2 Version 8

Step	Description	Job
1	Define DB2 Version 8 to z/OS, and build cataloged procedures.	DSNTIJMV
2	Optionally, define a new integrated catalog facility (ICF) catalog and alias.	DSNTIJCA
3	Define DB2 data sets.	DSNTIJIN
4	Define DB2 initialization parameters.	DSNTIJUZ
5	Initialize DB2 catalog and directory data sets.	DSNTIJID

Table 2. Overview of steps for installing DB2 Version 8 (continued)

Step	Description	Job
6	Optionally, prepare authorization exit routines.	DSNTIJEX
7	Optionally, prepare for SMF recording.	(none)
8	Optionally, establish subsystem security.	(none)
9	Establish the TSO environment for DB2.	DSNTIJVC
10	Optionally, connect IMS to DB2.	(none)
11	Optionally, connect CICS to DB2.	(none)
12	IPL z/OS.	(none)
13	Start DB2 Version 8.	(none)
14	Tailor the DB2 catalog.	DSNTIJTC
15	Define temporary work file table spaces and initial buffer pool sizes.	DSNTIJTM
16	Define and bind DB2 objects and user-maintained databases.	DSNTIJSG
17	Optionally, populate the user-maintained databases, and, if you are using DDF, populate the communications database (within the DB2 catalog).	(none)
18	Bind the packages for DB2 REXX Language Support.	DSNTIJRX
19	Take an image copy of the DB2 catalog and directory.	DSNTIJIC
20	Run the installation verification procedure.	DSNTEJ*xx*

For a detailed description of the installation procedure, see Chapter 7, "Installing the DB2 subsystem," on page 251.

Summary of migration to compatibility mode steps

After you perform the SMP/E steps and follow the steps on page 81 to run the installation CLIST, you can edit and run the jobs that migrate your Version 7 subsystem to a DB2 UDB for z/OS Version 8 subsystem in compatibility mode.

Migration to Version 8 compatibility mode includes the steps in Table 3.

Table 3. Overview of steps for migrating to DB2 Version 8 compatibility mode

Step	Description	Job
1	Perform premigration activities to determine unsupported objects.	DSNTIJPM
2	Optionally, run DSN1COPY with the CHECK option on the catalog table spaces, and invoke DSN1CHKR to check for broken links on your Version 7 subsystem.	(none)
3	Optionally, determine which plans and packages will be invalid after migration.	(none)
4	Optionally, check for consistency between catalog tables.	(none)
5	Take an image copy of your Version 7 catalog.	DSNTIJIC

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

Step	Description	Job
6	Establish the TSO environment for DB2.	DSNTIJVC
7	Optionally, connect IMS to DB2.	(none)
8	Optionally, connect CICS to DB2.	(none)
9	Stop DB2 Version 7.	(none)
10	Back up Version 7 volumes.	(none)
11	Define DB2 initialization parameters.	DSNTIJUZ
12	Optionally, establish subsystem security.	(none)
13	Define DB2 Version 8 to z/OS, and build cataloged procedures.	DSNTIJMV
14	Define system data sets.	DSNTIJIN
15	Define authorization exit routines.	DSNTIJEX
16	IPL z/OS.[1]	(none)
17	Start DB2 Version 8.	(none)
18	Tailor the DB2 Version 8 catalog.	DSNTIJTC
19	Optionally, invoke DSN1CHKR to check for broken links on Version 8. Optionally, run job DSNTIJCX to check indexes in the catalog and directory.	(none)
20	Prepare the dynamic SQL program, and define initial buffer pool sizes.	DSNTIJTM
21	Bind SPUFI and DCLGEN.	DSNTIJSG
22	Bind the packages for DB2 REXX Language Support.	DSNTIJRX
23	Take an image copy of the Version 8 catalog.	DSNTIJIC
24	Run Version 7 verification jobs (optional).	DSNTEJ*xx*
25	Make adjustments for release incompatibilities.	(none)

Note:

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

Summary of fallback steps

Fallback is the process of returning to Version 7 after successfully completing the catalog migration to DB2 UDB for z/OS Version 8. You can fall back if the catalog migration job DSNTIJTC completed successfully. You can fall back if a severe error occurs either during the subsequent migration steps or during operation of DB2 Version 8 in compatibility mode. **This process applies only to those who are migrating to Version 8 compatibility mode. You cannot fall back from Version 8 enabling-new-function or new-function mode.** This process is described in "Falling back" on page 344. To fall back, perform the steps in Table 4.

Table 4. Overview of steps to fall back to Version 7

Step	Description	Job
1	Stop DB2 Version 8.	(none)
2	Rename the cataloged procedures.	DSNTIJFV
3	Reconnect TSO, IMS, and CICS to DB2 Version 7.	(none)

Table 4. Overview of steps to fall back to Version 7 (continued)

Step	Description	Job
4	Start Version 7.	(none)
5	Relink Version 7 DSNTIAR with applications.	(none)
6	Run the Version 7 installation verification jobs.	DSNTEJ*xx*

Summary of remigration steps

Migration to DB2 UDB for z/OS Version 8 after falling back to Version 7 (remigration) is simpler than the initial migration. To remigrate to Version 8, perform the steps that are listed in Table 5.

Table 5. Overview of steps for remigration to DB2 Version 8

Step	Description	Job
1	Run DSN1COPY with the CHECK option on the catalog table spaces. Invoke DSN1CHKR to check for broken links on Version 7. Run the queries in DSNTESQ to check for consistency between catalog tables.	DSN1CHKR
2	Take an image copy of Version 7.	DSNTIJIC
3	Stop Version 7.	(none)
4	Reconnect TSO, IMS, and CICS to DB2 Version 8.	(none)
5	Rename cataloged procedures.	DSNTIJMV
6	Start DB2 Version 8.	(none)
7	Optionally, take an image copy of the Version 8 catalog.	DSNTIJIC
8	Delete steps DSNTIJD, DSNTIJR, DSNTIJC, and DSNTIJG. In step DSNTIRU, delete all statements that are not needed to bind SPUFI and DCLGEN. Run DSNTIJSG.	DSNTIJSG
9	Optionally, run Version 7 verification jobs.	DSNTEJ*xx*

In addition, you must perform some tasks manually. These tasks, as well as the tasks that are performed by the previously listed jobs, are explained in “Remigrating” on page 349.

Summary of conversion to new-function mode steps

After you migrate to Version 8 compatibility mode, you can convert to Version 8 new-function mode.

Migration to Version 8 new-function mode includes the steps in Table 6.

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

Step	Description	Job
1	Run installation CLIST using ENFM option.	(none)
2	Take an image copy of your Version 8 catalog.	DSNTIJIC
3	Convert the DB2 Version 8 catalog and directory.	DSNTIJNE
4	Enter new-function mode.	DSNTIJNF

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

Step	Description	Job
5	Convert stored procedures for JDBC and ODBC support	DSNTIJMS
6	Optionally, check for consistency between catalog tables.	(none)
7	Stop DB2 Version 8.	(none)
8	Create DSNHDECP module for new-function mode	DSNTIJNG
9	Start DB2 Version 8	(none)
10	Optionally, invoke DSN1CHKR to check for broken links on Version 8. Optionally, run job DSNTIJCX to check indexes in the catalog and directory.	(none)
11	Optionally, check indexes in the catalog and directory	DSNTIJCX
12	Optionally, run Version 8 verification jobs.	DSNTEJ*xx*
13	Make adjustments for release incompatibilities.	(none)

Part 2. Planning and installing DB2

Chapter 2. Estimating DB2 storage needs

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

You can use Data Facility Storage Management Subsystem (DFSMS) to manage DB2 data sets. It provides automatic backup and recovery features, which might require disk storage beyond what is estimated below. For more information, see *z/OS DFSMSdfp Storage Administration Reference*.

This chapter explains how to calculate storage requirements for a small site, a medium site, a large site, and an extra-large site. For specific estimates, refer to *DB2 Program Directory*. The following site models are based on several assumptions. You can use these models to help you estimate your storage needs.

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

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

Storage estimates for items that are specific to data sharing are explained in *DB2 Data Sharing: Planning and Administration*.

This chapter describes the following aspects of storage:
- "DB2 subsystem storage requirements"
- "Virtual storage requirements for address spaces" on page 28
- "Virtual storage requirements for storage pools and working storage" on page 32
- "Real storage requirements" on page 43

DB2 subsystem storage requirements

Refer to *DB2 Program Directory* for tables that show estimated space requirements for specific environments.

This topic assumes that, when running the installation CLIST, you accept the default values for the number of databases, tables, and application plans that are expected at your site. You specify these values on installation panel DSNTIPD.

If you do not accept the default values, you can calculate the storage that you need for the DB2 data sets by using the information in "Active log data sets storage requirements" on page 19. For other data sets, you can use the formulas in the CLIST. After calculating the required storage for each data set, you can calculate the total requirements.

To determine the storage requirements based on your storage device model, check the values in Table 7. The space requirements **do not** include space for user databases, image copies, archive logs, or temporary data sets that you create while installing or migrating.

Table 7. Estimated space requirements (in cylinders) for DB2 by site size

Site size	3380	3390
Small	1116	949
Medium	1781	1499
Large	6150	5142
Extra-large	10630	8872

Table 8 provides estimated storage requirements in megabytes (MB) for DB2 data sets. Individual values are rounded and may not add up to the total. Estimated space requirements do not significantly differ by device type. Although the DB2 libraries require a fixed amount of space, disk requirements for active logs and the DB2 catalog increase with the size of a site. You need additional space for archive logs, image copies, user databases, and other working data sets.

Table 8. Estimated space requirements (in megabytes) for DB2 data sets by site size

Site size	DB2 libraries	DB2 catalog	Directory	Active logs	BSDS	Temporary database	Total
Small	316	199	61	102	10	24	712
Medium	316	342	198	204	10	24	1094
Large	316	546	385	2028	10	372	3657
Extra-large	316	747	572	4050	10	580	6275

For information about disk requirements for DB2 libraries and SMP/E data sets, see the *DB2 Program Directory*.

DB2 catalog storage requirements

Storage requirements for the entire set of DB2 catalog data sets and their indexes are shown in Table 9.

Table 9. Estimated space requirements (in cylinders) for the DB2 catalog by site size

Site size	3380	3390
Small	248	224
Medium	504	436
Large	853	727
Extra-large	1204	1017

If you plan to use partitioned table spaces that are created with (or will grow to have a large number of) partitions, you might need to allocate more storage than what is suggested in Table 9. Some catalog objects can grow substantially over time if a large number of partitions are created or added to table spaces.

For information about how to change the size of catalog data sets after you install or migrate DB2, see Part 5 (Volume 2) of *DB2 Administration Guide*.

DB2 directory storage requirements

Directory space depends mainly on the number of user databases, application plans and packages, and tables in the DB2 subsystem. Storage requirements for the DB2 directory are shown in Table 10.

If you plan to use partitioned table spaces that are created with (or will grow to have a large number of) partitions, you might need to allocate more storage than what is suggested in Table 10. Some directory objects can grow substantially over time if a large number of partitions are created or added to table spaces.

Table 10. Estimated space requirements (in cylinders) for the DB2 directory by site size

Site size	3380	3390
Small	99	84
Medium	334	278
Large	652	543
Extra-large	970	808

Active log data sets storage requirements

Active log data sets record significant events and data changes. Active log data sets are periodically offloaded to the archive log. Therefore, the storage requirements for your active log data sets depend on how often DB2 data is changed at your site and how often DB2 offloads those changes to the archive log.

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

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

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

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

1. During the ISPF tailoring session, assume that you specified:
   - An archive period estimate of 24 hours (ARCHIVE LOG FREQUENCY parameter on installation panel DSNTIPL).
   - A data change rate estimate of 3600 changes per hour (UPDATE RATE parameter on installation panel DSNTIPL).
2. DB2 uses 400 bytes as the size of a typical row.
3. Other types of log records are comparatively small in size and are fixed in length. The length of each type depends on the information that it contains.
4. The size of the active log, including disk track overhead, is estimated as:

Data set size

= (data change log record size)
* (data change rate per hour)
* (hours in archive period)
= 400 bytes * 3600 per hour * 24 hours
+ data set allocation overhead
= 34 MB
If you have dual logs, you will need 68 MB.
If you have dual logs with three data sets each, you will need 204 MB.

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

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

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

If you enabled data sharing, you generally need to have more disk space for the active log, and you need to archive the logs more frequently. See Chapter 5 of *DB2 Data Sharing: Planning and Administration* for more information.

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

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

Table 11 shows estimated storage requirements for active log data sets (assuming dual logging).

Table 11. Estimated space requirements (in megabytes) for active log data sets by site size

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

Table 12 on page 21 shows the amount of space that is required for active log data sets on various devices. The estimates in both of these tables include track overhead.

Table 12. Estimated space requirements (in cylinders) for the active log by site size

Site size	3380	3390
Small	174	144
Medium	348	288
Large	3456	2880
Extra-large	6912	5760

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

- Tape utilization

  When you archive to a media type that is listed in Table 13, the planning size numbers are suggested sizes for your active logs. If the size of an active log data set is small compared to the size of a tape, the tape utilization is fairly low.

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

Table 13. Estimated active log planning size

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

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

- Checkpoint frequency

  The CHECKPOINT FREQ field on panel DSNTIPL specifies either the number of consecutive log records that are to be written between DB2 system checkpoints or the number of minutes per checkpoint. When choosing a value, you must consider the trade-offs between the overhead that is needed for frequent subsystem checkpoints and the time that is needed to restart a DB2 subsystem after a termination without a quiesce. If the checkpoint value is more than 1 million, the time that is needed to restart DB2 after a termination without a quiesce can grow to over 15 minutes. The recommended values for the checkpoint frequency are in the range of 500 000 to 1 million in log records or 2 to 5 in minutes.

- Number of tables that are defined for data capture

  When tables are defined with the DATA CAPTURE CHANGES option, the entire before-image of an updated row is captured on the log. This additional information can represent an increase in log data compared to tables that are not defined with the DATA CAPTURE CHANGES option, depending on whether the table contains fixed-length or variable-length rows. For information on what is logged for updated rows in both data-capture and non-data-capture tables, see Appendix C (Volume 2) of *DB2 Administration Guide*.

Bootstrap data sets storage requirements

Each bootstrap data set (BSDS) requires 3.5 MB. If you are installing, DB2 automatically allocates two copies of the BSDS. If you are migrating, Version 8 adopts the BSDS characteristics that you specified for your previous version. That is, if you specified two copies of the BSDSs for your previous version, you will have two copies for Version 8. Keeping two copies of the BSDS is strongly recommended. The total space requirement is about 7 MB for both BSDSs. The BSDSs at any size site require about 12 cylinders of 3380 storage or 10 cylinders of 3390 storage.

If you are migrating and need more than 31 active log data sets and 1000 archive log volumes, you can get more by converting the BSDSs. Each new BSDS requires approximately 3.5 MB. For more information about converting BSDSs, see "Considerations for the BSDS" on page 361.

Work file database storage requirements

The work file database is used as storage for processing SQL statements that require working storage. Table 14 shows the disk requirement estimates for the temporary work files in the work file database. Other work file database storage requirements relate to migration.

Table 14. Estimated space requirements (in cylinders) for the work file database by site size

Site size	3380	3390
Small	35	29
Medium	35	29
Large	547	456
Extra-large	854	712

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

- Data size
- Sort key size

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

- Let MIN be the operation of selecting the lowest value from a set of values.
- Let FLOOR be the operation of discarding the decimal portion of a real number.
- Let CEILING be the operation of rounding a real number up to the next-highest integer.
- Let *data* be the total data length in bytes.
- Let *key* be the total length of the sort key.
- Let *prefix* be the 6-byte header.
- Let *rows* be the total number of rows that are being sorted.

Then calculate as follows:

Records per page = MIN(MAXROWS, FLOOR (4076 / (*data* + *key* + *prefix*)))

Total pages = CEILING (*rows* / *records per page*)

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

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

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

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

To determine the number of tracks that are needed, convert the number of pages into bytes, and divide the result by the number of bytes per unit. Let *r* be the number of 4096-byte records per track, and let *safety_factor* be a number from 1.5 to 2.0. For 3390 devices, *r* is 12. For 3380 and 9340 devices, *r* is 10.

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

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

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

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

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

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

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

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

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

You can use the sort summary trace record, IFCID 0096, to simplify some of the calculations. This record shows the number of records that are sorted, the sort record size (*Data* + *Key*), and an indication of whether a merge phase was required for an individual sort request. For information about the trace facility of DB2, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Default database storage requirements

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

Temporary table space storage requirements

DB2 uses declared temporary tables for processing the following types of scrollable
cursors:

• SENSITIVE STATIC SCROLL
• INSENSITIVE SCROLL
• ASENSITIVE SCROLL, if the effective cursor sensitivity is INSENSITIVE. A
cursor that meets the criteria for a read-only cursor has an effective sensitivity of
INSENSITIVE. See the DECLARE CURSOR statement in *DB2 SQL Reference* for
more information.

Before application programmers can use cursors that require declared temporary
tables, you need to create a temporary database and temporary table spaces for
those declared temporary tables that are large enough to process your cursors. For
example:

```
# CREATE DATABASE DTTDB AS TEMP;
#
# CREATE TABLESPACE DTTTS IN DTTDB
#   SEGSIZE 4;
```

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

The page size of the table space in a temporary database must be large enough to hold the longest row in the declared temporary table. The size of a row in the declared temporary table might be considerably larger then the size of the row in the table for which the static scrollable cursor is used. The size of the row depends on these factors:
- The number of columns that are stored in the declared temporary table
- The size of each column

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

To calculate the size of the longest row in the declared temporary table, follow these steps:
1. Identify the columns in the declared temporary table. See "Identifying the columns in the declared temporary table."
2. Determine the length of each column in the declared temporary table. See "Determining the lengths of the columns in the declared temporary table."
3. Calculate the total length of the longest declared temporary table row. See "Calculating the length of the longest row in a declared temporary table" on page 26.

| **Important:** If you use declared temporary tables, you must define at least one of
| the table spaces in the temporary database to have a page size of 8 KB or greater.

Identifying the columns in the declared temporary table

The columns that are in the declared temporary table for a scrollable cursor depend on whether the result table of the cursor is read-only.

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

For a result table that is not read-only, the following items are columns in the declared temporary table:
- Each column in the select list
- Each expression in the select list that contains more than a single column name
- Three additional columns that DB2 generates

Determining the lengths of the columns in the declared temporary table

After you identify the columns that are in the declared temporary table, you need to determine the length of each column. Use the following method:
1. For columns other than the columns that DB2 generates, determine the data type of each column. See *DB2 SQL Reference* for this information.

   Determine the length of each column, based on the data type, in the following way:
   - For a declared temporary table column that is the result of the concatenation of CHAR, VARCHAR, GRAPHIC, or VARGRAPHIC data types, use the numbers in *DB2 SQL Reference*. Add 1 byte if the column is nullable.
   - For a declared temporary table column that is the result of an expression that contains LOB columns, specify a length of 120 for each column, literal, or host variable that is referenced in the expression. Then add these lengths for the expression.

- For a LOB data type, specify a length of 120.
- For a ROWID data type, specify a length of 42.
- For a declared temporary table column of any other data type, use the information in *DB2 SQL Reference* to determine the column length. Add 1 byte if the column is nullable.

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

Calculating the length of the longest row in a declared temporary table

To determine the length of the longest row in the declared temporary table, find the sum of the column lengths that you calculated in the "Determining the lengths of the columns in the declared temporary table" on page 25 topic.

Example

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

C1	CLOB(100M)
C2	CHAR(10)
C3	VARCHAR(100) NOT NULL
C4	INTEGER NOT NULL
C5	INTEGER

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

```
DECLARE CUR1 INSENSITIVE SCROLL CURSOR FOR
  SELECT C1, C2 || C3, C4 FROM T1
  WHERE C3 > :HV;
DECLARE CUR2 SENSITIVE STATIC SCROLL
CURSOR FOR
  SELECT C1, C2||C3, C4 FROM T1
  WHERE C3 > :HV;
```

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

Column	Data type	Effective length
C1	CLOB(100M)	120
C2 || C3	VARCHAR(110)	113
C4	INTEGER NOT NULL	4
C5	INTEGER	5
One column added by DB2	(N/A)	22
Total length	(N/A)	265

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

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

Dump data set size storage requirements

Recommendation: Use these guidelines for the dump data sets:
- Have at least two dump data sets.
- Have approximately 250 cylinders of 3390 disk space for each SYS1.DUMPxx data set that you have defined.
- Have 3.25 MB of space for DB2 volatile summary storage data.

Summary data in dumps is usually enough to diagnose most problems. In addition to summary data, DB2 also requests a SDUMP from the operating system to provide these additional storage areas if enough space is available in the dump data set:
- DB2 system services address space
- DB2 database services address space
- DB2 distributed data facility (DDF) address space
- Allied address space of the failing allied task
- IRLM address space for data sharing environments

DB2 passes the following parameters to the SDUMP service aid through the SDATA keyword: SQA, ALLPSA, LSQA, SUMDUMP, and CSA (subpools 231 and 241). Refer to *z/OS MVS Programming: Assembler Services Reference, Volumes 1 and 2* for more information about the SDUMP service aid.

After DB2 SVC dump processing is complete, z/OS issues message IEA911E to indicate whether enough space was available in the dump data set to contain the requested storage areas. If this message indicates that a partial dump was taken, but the 3.25 MB of summary storage is available in the dump, this dump is probably enough for problem diagnosis. Otherwise, IBM Software Support might request that you re-create the problem if storage areas that are required for problem determination are not included in the dump.

System databases storage requirements

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

Archive log data sets storage requirements

If you decide to store the archive log data sets on disk, you need to reserve enough space on the device. The active log data set and the BSDS are both written to the same location. Therefore, you must reserve enough storage for the active log and the BSDS. Use the information in “Active log data sets storage requirements” on page 19 and “Bootstrap data sets storage requirements” on page 22 to determine these sizes. In addition, the total amount of storage that is required for the logs and BSDSs is calculated by the CLIST and displayed in the messages for Volume Serial 5 and Volume Serial 6 on DSNTIPC1. For more information about CLIST storage calculation, see page238

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

Important: Do not specify DFSMS compression for archive logs on disk storage. If you do, you might receive log-read failures when attempting to recover from the archive logs.

Using the installation CLIST to calculate storage

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

Recommendation: Use the model site estimates the first time that you install DB2. Use the model approach that is described in "DB2 subsystem storage requirements" on page 17 to estimate DB2 disk use. After your site has some experience in operating DB2, you can recalculate your disk estimates.

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

To see the algorithms that DB2 uses, print or edit the installation CLISTs and REXX EXEC. The DSNTCALC EXEC contains most of the data set calculations. You can run the CLIST to calculate the sizes. Installation panel DSNTIPC on page 238 displays the storage sizes that the CLIST calculates.

Virtual storage requirements for address spaces

DB2 uses these types of private address spaces:
- DB2 distributed data facility (DDF) address space (DSN1DIST)
- IRLM address space (IRLMPROC)
- DB2 system services address space (DSN1MSTR)
- DB2 database services address space (DSN1DBM1)
- DB2 allied agent address spaces
- DB2 stored procedures address spaces (DSN1SPAS and WLM-named)

DB2 also uses extended common service area (ECSA).

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

DB2 distributed data facility address space (DSN1DIST)

This address space supports network communications with other remote systems and execution of database access requests on behalf of remote users.

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

IRLM address space (IRLMPROC)

| DB2 uses the IRLM to manage locks. When row locking is used, the number of
| locks that DB2 acquires might increase, which might in turn increase the amount of
| storage that IRLM requires. The number of locks that are acquired is dependent on
| your application. You can estimate the IRLM control block structure at 540 bytes
| per lock. IRLM no longer supports placing locks in ECSA. All IRLM locks are now
| placed in the IRLM private address space.

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

| Use the `MODIFY irlmproc,STATUS,STOR` command to view and monitor the amount
| of private storage that IRLM has available. You can adjust the amount of private
| storage dynamically with the MODIFY irlmproc SET,PVT command. The new
| value remains in effect until the next time IRLM is stopped and restarted or until
| the MODIFY command is issued successfully again. This only changes the
| monitoring threshold of private storage for IRLM. It does not change the physical
| amount that the operating system assigned to the address space.

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

For additional information about the IRLM start up procedure parameters, see *DB2 Command Reference*.

DB2 system services address space (DSN1MSTR)

The DB2 system services address space performs a variety of system-related functions. It needs less space than the database services address space.

Recommendation: Specify 0 KB for the system services address space, but plan to use 2 MB below the 16-MB line. The default start up procedure is DSN1MSTR.

DB2 database services address space (DSN1DBM1)

| The DB2 database services address space is the largest DB2 address space. This
| address space uses storage above the 2-GB bar. The default start up procedure is
| DSN1DBM1. First, plan for a minimum of 30 MB in this address space, with 1334
KB below the 16-MB line. The rest of this chapter discusses the various
components of the database services address space.

Most modules, control blocks, and buffers reside in the extended private area. A DB2 subsystem with 200 concurrent users and 2000 open data sets should need less than 2 MB of virtual storage below the 16-MB line.

Allied agent address space

DB2 refers to the user address spaces as the allied agent address space. This can include Resource Recovery Services attachment facility (RRSAF), TSO attach, IMS, CICS, and batch address spaces. The size of the DB2 attachment facility code in the allied agent address space depends on which attachment facilities you use. TSO requires about 130 KB for the DSN command. CAF and IMS each require about 36 KB for the DB2 attachment facility code. For all attachment facilities, except CICS Transaction Server, the DB2 attachment facility code must run below the 16-MB line of virtual storage. Applications can run above the 16-MB line. To calculate space requirements for the CICS attachment facility, see *CICS Transaction Server for z/OS DB2 Guide*.

Common service area

Some of the DB2 load modules and control blocks are in common storage. Most of the space is in the extended common service area (ECSA). With few exceptions, the CSA-resident load modules are link-edited with the residency attribute of RMODE(ANY). Most of the modules reside in ECSA (above the 16-MB line of virtual storage), as do most of the global control blocks. The IRLM control blocks are above the 2-GB bar.

To calculate the approximate residual requirement for CSA (below the 16-MB line), use the following guidelines:
- Start with up to 40 KB for each DB2 subsystem.
- Add 24 KB for each started IRLM.
- Add 1 KB for every 13 latch contentions.
- Add 4 KB for every 4 notify requests.

To estimate storage that is needed for ECSA (above the 16-MB line) for each DB2 subsystem, follow these guidelines:
- Start with 3 MB of ECSA for the base and the first 100 users.
- Start with 0.1 MB for IRLM.
- Add 1.9 MB for IRLM required trace buffers.
- Add 1.9 MB for IRLM optional trace buffers.
- Add 4 KB for each additional user.
- Add 3 KB for each active remote thread.
- Add 4 MB or more for instrumentation facility interface (IFI) buffers as requested by the monitoring programs.

If you use the distributed data functions of DB2, you may find that you need more virtual storage. You can estimate how much your storage needs are likely to increase in the ECSA above the 16-MB line by adding the following amounts:
- 1 KB for each conversation
- 2 KB for each thread that uses distributed processing
- 1 KB for each DB2 site in your network
- 40 KB for code that relates to distributed processing

Under normal conditions, your virtual storage needs in the CSA (below the 16-MB line) will probably not increase. In a data sharing environment, if an IRLM performs member recovery or structure rebuild, additional virtual storage is required. Any other increase in the amount of virtual storage that is needed occurs within the extended private area of the DB2 database address space and the extended private area of the distributed data address space.

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

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

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

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

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

These values are rounded down (CSA) or up (ECSA) to the next 1-MB boundary. For more information, see *z/OS MVS Initialization and Tuning Guide*.

WLM-established stored procedures address spaces

These are WLM-established address spaces that provide multiple isolated environments for stored procedures. The advantages of using WLM-established stored procedures address spaces over a DB2-established stored procedures address space include:
- Stored procedures are isolated, so failures do not affect other stored procedures.
- Demand for storage below the 16-MB line is reduced, thereby removing the limitation on the number of stored procedures that can run concurrently.
- Stored procedures inherit the z/OS dispatching priority of the DB2 thread that issues the CALL statement.
- | The DB2-established stored address space is deprecated. New stored procedures | created or altered in Version 8 must be WLM-managed.

Recommendation: Use partitioned data set extended (PDSE) for load libraries that contain stored procedures. Using PDSEs might eliminate your need to stop and start the stored procedures address space due to growth of the load libraries. If a load library grows from additions or replacements, the library might need to be extended.

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

DB2 UDB for z/OS stored procedures support both main programs and subprograms; this support requires additional storage for each TCB. However, because you can run fewer programs in an address space, you can use less storage below the 16-MB line in each address space. For more information about controlling address space storage, see Part 5 (Volume 2) of *DB2 Administration Guide*.

DB2-established stored procedures address space (DSN1SPAS)

In Version 8, support for DB2-established address spaces is deprecated. Although existing stored procedures can still run in a DB2-established stored procedure address space, you should move your stored procedures to WLM environments as soon as possible. If you CREATE or ALTER an existing stored procedure, it cannot run in a DB2-established stored procedure address space. For more information about moving stored procedures, see Part 5 of *DB2 Application Programming and SQL Guide*.

Recommendation: Use partitioned data set extended (PDSE) for load libraries that contain stored procedures. Using PDSEs might eliminate your need to stop and start the stored procedures address space due to growth of the load libraries. If a load library grows from additions or replacements, the library might need to be extended.

If you use PDSEs for the load libraries, the new extent information is dynamically updated and you do not need to stop and start the address space. If PDSs are used, load failures may occur because the new extent information is not available.

See Part 6 of *DB2 Application Programming and SQL Guide* for more information.

Virtual storage requirements for storage pools and working storage

You specify values during the ISPF tailoring session that the DSNTINST CLIST uses to calculate main storage size.

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

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

The calculations in this topic are planning estimates. The noted values do not provide the exact limits, but they indicate a reasonable range of values. More detailed information is provided in Part 5 (Volume 2) of *DB2 Administration Guide*.

This topic presents information about the following values:
- Buffer pool size
- Sort pool size
- Record identifier (RID) pool size
- Environmental descriptor manager (EDM) pool size
- VSAM data set control block storage size
- Working storage size.

The sum of the following values must fit the region size that DB2 supports:
- Environmental descriptor manager (EDM) pool size
- VSAM data set control block storage size
- Working storage size

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

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

After you specify the values listed above, the CLIST calculates the EDM pool size and the size that is needed for the data set control blocks. The CLIST adds the working storage size and the fixed code size to update the region size that is used in the DB2 startup procedures. The CLIST also displays this information on installation panel DSNTIPC, which is described on page 238.

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

Buffer pool size calculation

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

Virtual buffer pools: For best results, use at least 100 KB of buffer pool space for each concurrent user. A value of 300 KB or more for improved performance is recommended. Very simple SQL statements that access small amounts of data can require less than this amount. Complex SQL statements that access large amounts of data can require more than this amount.

During installation, you can set the buffer pool sizes on the installation panels. Later, you can use the ALTER BUFFERPOOL command to alter the sizes and other attributes of as many as 50 buffer pools for 4-KB page sets, 10 buffer pools for 8-KB page sets, 10 buffer pools for 16-KB page sets, and 10 buffer pools for 32-KB table spaces. The ALTER BUFFERPOOL command can make the changes dynamically while DB2 is running. See Part 5 (Volume 2) of *DB2 Administration Guide* for information on changing the buffer pool sizes.

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

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

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

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

Use Table 15 to calculate the virtual buffer pool sizes for your subsystem.

Table 15. Virtual buffer pool size calculation

Virtual buffer pool calculation			Default
Buffers for BP0	____ x 4 KB = _____	20 000 x 4 KB	= 80 000 KB
Buffers for BP1	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
Buffers for BP2	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
.			
.			
.			
Buffers for BP49	+____ x 4 KB = _____	+ 0 x 4 KB	= 0 KB
Buffers for BP8K0	+____ x 8 KB = _____	+ 1000 x 8 KB	= 8000 KB
Buffers for BP8K1	+____ x 8 KB = _____	+ 0 x 8 KB	= 0 KB
.			
.			
Buffers for BP16K0	+___ x 16 KB = ____	+ 500 x 16 KB	= 8000 KB
Buffers for BP16K1	+___ x 16 KB = ____	+ 0 x 16 KB	= 0 KB
.			
.			
Buffers for BP32K	+___ x 32 KB = ____	+250 x 32 KB	= 8000 KB
.			
.			
Buffers for BP32K9	+___ x 32 KB = ____	+ 0 x 32 KB	= 0 KB
	= ____		= 104 000KB

Sort pool size calculation

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

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

The DB2 sort work area (in-memory) has the following storage boundaries for **each** concurrent sort operation:
  Minimum sort storage = 240 KB
  Maximum sort storage = 128 MB

DB2 initially allocates 240 KB for each sort and gradually adds more storage until the maximum sort work area limit is reached or the maximum number of nodes (32K) are populated at the bottom of the sort tree, whichever occurs first. With 32K nodes at the bottom of the sort tree, the average run size for each sorted string is 64K records.

The default size of the sort pool is 2 MB, but you can override this default value by entering the desired sort pool size on installation panel DSNTIPC. Estimate the required storage for a sort pool with the following formula:

```
32000 * (12 + sort key length + sort data length + 4 (if ESA hardware sort
assist))
```

See Part 5 (Volume 2) of *DB2 Administration Guide* for instructions on choosing sizes for optimal performance.

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

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

DB2 considers the buffers that it uses for work files as sequentially accessed pages. You can adjust the percentage of the buffer pool that is used for work files by using the VPSEQT parameter of the ALTER BUFFERPOOL command. If a buffer pool is used only for work files, you might set VPSEQT to 100%. If you do not have enough allocated storage to complete sort processing, you must allocate more disk space for the work file database. For more information, see "Work file database storage requirements" on page 22.

RID pool size calculation

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

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

Consider the following amounts when calculating RID pool size:
- Startup RID storage = 0 (but acquired in 32-KB blocks as needed)
- Maximum RID storage = 10 000 MB

The default size of the RID pool is 8 MB, but you can override this default value by entering the desired RID pool size on installation panel DSNTIPC. If you do this, estimate the required storage for the RID pool with the following formula:

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

See Part 5 (Volume 2) of *DB2 Administration Guide* for an example of changing a RID pool size to improve performance.

EDM pool size calculation

The environmental descriptor manager (EDM) pool contains active and skeleton application plans and packages. This topic describes how to estimate the space that is needed for plans, packages, and cached prepared statements. These estimations do not account for every factor that affects EDM pool size. Your actual EDM pool size might vary. A general recommendation is that the EDM pool be at least 10 times the size of the largest database descriptor or plan, whichever is greater.

In Version 8, the EDM pool has been separated into three pools: the EDM pool, database descriptor (DBD) pool, and the statement pool. The EDM pool space for database descriptors and dynamic statements has been moved above the 2-GB bar. The EDM pool, which contains the skeleton cursor table and skeleton package table, is below the 2-GB bar.

The CLIST generates the values for these pools. These values are on installation panel DSNTIPC, which is described on page 238.

Calculating the EDM pool space for plans and packages

The first part of the EDM pool calculation involves the space for plans and packages. Plans and packages are very different, but they are treated similarly for storage planning. To estimate the EDM pool space that is needed for plans, use the following variables:

- Let *concplans* be the number of concurrently executing plans. This is the value that is specified as MAX USERS on installation panel DSNTIPE.
- Let *maxplans* be the maximum number of unique plans that you want in the EDM pool at any given time. Estimate this by taking one fourth of the total plans that you have.
- Let *statsize* be the average statement size. To calculate the average statement size, add 1.4 KB for each single table statement and 0.2 KB for each additional table in the statement. For example, if you estimate one table for each statement, multiply the number of SQL statements by 1.4 KB; if you estimate three tables per statement, multiply the number of statements by 1.8 KB (1.4 KB + 0.2 KB + 0.2 KB).
- Let *statexec* be the average number of executed statements. The CLIST uses the value in EXECUTED STMTS on installation panel DSNTIPD.

Then calculate as follows:

```
(concplans + maxplans) * (statsize * statexec)
```

The average plan size changes in proportion to the increase in the number and complexity of SQL statements. The complexity of the access path for SQL statements can also affect plan size.

To calculate your average plan size, you need to allow from 1 KB to 4 KB for control blocks for caching, depending on your BIND option. Let *cntrlblk* be the number of KBs that are allocated for caching. Then, calculate the average plan size with the following formula:

```
(statsize *statexec) + cntrlblk
```

If you use the defaults, the CLIST makes the following calculations. First, the values for the single table and the additional table are added to determine the *statsize*.

```
           1.4  KB (single table)
   +       1.2  KB (additional table)
------------------------------------
statsize = 1.6  KB
```

Then the CLIST multiplies the result, *statsize*, by the default value for *statexec*, and adds 1 KB for caching control blocks:

```
    1.6  KB (statsize)
  x  15  (statexec   )
----------------------
     24  KB
  +   1  KB
-----------
     25  KB
```

The size of a plan during execution is typically 25 KB.

The ACQUIRE and RELEASE options of the BIND command affect the plan size as follows:

- ACQUIRE(ALLOCATE) results in a larger plan size than ACQUIRE(USE).
- RELEASE(DEALLOCATE) can result in a larger plan size than RELEASE(COMMIT). RELEASE(DEALLOCATE) usually results in holding the storage longer.

As a result, the amount of EDM pool storage that the plan consumes is affected. ACQUIRE(ALLOCATE) and RELEASE(DEALLOCATE) cause items to be stored with the plan to enhance performance.

The increase in plan size is determined by the number of database objects (databases, table spaces, and tables) that the plan uses. In CICS and IMS environments, thread reuse tends to increase the number of database objects that the plan uses when the RELEASE(DEALLOCATE) BIND option is used.

General-use Programming Interface

After you bind a plan or package, you can check the size by querying the SYSPLAN and SYSPACKAGE catalog tables. PLSIZE or PKSIZE is the size of the base segment of the plan or package. AVGSIZE is the average size for each section. CACHESIZE is the cache size that you specify for the authorization ID cache for the plan. To find the plan or package sizes and the size of the authorization ID cache, use the appropriate SQL statement:

For a plan:

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

For a package:

```
SELECT NAME, PKSIZE, AVGSIZE
FROM SYSIBM.SYSPACKAGE
ORDER BY NAME;
```

To find the number of sections for each DBRM that is bound with each plan, use the following SQL statement. To find the number of sections in each plan, add the number of sections for each DBRM by plan:

```
SELECT PLNAME, NAME,
CASE WHEN MAX(SECTNOI) <> 0
THEN MAX(SECTNOI)
ELSE MAX(SECTNO)
END AS SECTNUM
FROM SYSIBM.SYSSTMT
GROUP BY PLNAME, NAME
ORDER BY PLNAME, NAME;
```

To find the number of sections for each DBRM bound with each package, use the following SQL statement. Add the number of sections for each DBRM by package to find the number of sections in each package.

```
SELECT COLLID, NAME, VERSION,
CASE WHEN MAX(SECTNOI) <> 0
THEN MAX(SECTNOI)
ELSE MAX(SECTNO)
END AS SECTNUM
FROM SYSIBM.SYSPACKSTMT
GROUP BY COLLID, NAME, VERSION
ORDER BY COLLID, NAME, VERSION;
```

End of General-use Programming Interface

You can also use similar statements with WHERE clauses to specify the plans or packages that you want. The number of sections that are executed per plan or package can be estimated only from the execution of the application that uses the particular plan or package. Consequently, the amount of storage that is needed during execution of the application is the base segment size of the plan or package plus the size for the sections that are being executed.

The storage that is needed for a plan is the sum of the base size and the size of the executed sections. The storage that is needed for a package is the sum of the base size, the base size of the package, and the size of the executed sections.

Calculating the EDM pool space for the prepared-statement cache

| You should consider increasing the size of your EDM pool. This pool space is
| above the 2-GB bar.

When you use the cache, prepared statements are stored in the EDM pool, as are static SQL statements. The number of prepared statements that are stored in the cache depends on the characteristics of the dynamic SQL that your application executes. One type typically benefits from caching prepared statements, while the other type usually does not.

The first type of applications use dynamic SQL that is embedded in an application and is used repeatedly. Applications and queries with this type of SQL benefit most from caching prepared statements because the statement can be used from the cache.

However, applications that contain SQL statements that are infrequently used pay the cost of being added to the cache. For example, queries from DB2 QMF are likely to be prepared and executed only once. Caching prepared statements does not benefit applications that extensively use this kind of SQL.

You should use dynamic statement caching when you have an INSERT statement using host variables.

Estimate the storage that is needed for the prepared-statement cache by using these variables:

- Let *maxdynplans* be the maximum number of unique plans that contain embedded dynamic SQL that you expect to be running in the system at any given time.
- Let *dynstatsize* be the average statement size of a prepared statement in the cache. To calculate the average prepared statement size, add 1.8 KB for each single table statement and 0.2 KB for each additional table in the statement. For example, if you assume an average of three tables per statement, the average statement size is 2.2 KB (1.8 KB + 0.2 + 0.2).
- Let *dynstatexec* be the average number of different dynamic SQL statements that are executed by plans that contain dynamic SQL that are likely to be used repeatedly.
- Let *adhocexec* be the maximum number of unique SQL statements that are likely to be used only once and that are executed by all users at any given time. You can estimate this number by multiplying the maximum number of users who are running ad hoc SQL programs by 5.

Then calculate the size needed for the prepared-statement cache as follows:

```
(maxdynplans * (dynstatsize + dynstatexec) + (adhocexec * dynstatsize))
```

The installation CLIST does not attempt to calculate this value when it calculates the estimated EDM pool size because it does not have all the information that is provided by the variables.

Calculating the EDM pool space for database descriptors

The final part of the EDM pool calculation involves the space for database descriptors (DBDs). This portion of the EDM pool is above the 2-GB bar. To determine this value, multiply the number of concurrently open databases by the average size of the database descriptor.

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

Recommendation: If your tables are small, you should use fewer than 50 tables in a segmented table space. For very large tables that are partitioned and larger than 1-GB, you should use one table per database. You can use additional databases to assist with logging, space management, and concurrency.

Using Table 16, you can estimate the DBD size based on an estimate of columns in each table and tables in each database for your site. Assume:
- The same number of tables as table spaces
- 2 indexes in each table
- 4 partitions in each table space
- 3 keys in each index
- 2 referential relationships in each table

These values are the defaults that are built into the CLIST and are reasonably typical for databases.

Table 16. DBD sizes for ranges of columns and tables

Columns in each table	10 tables in each database	20 tables in each database	30 tables in each database	40 tables in each database	50 tables in each database
10	65 KB	117 KB	168 KB	220 KB	271 KB
20	93 KB	173 KB	253 KB	332 KB	412 KB
30	122 KB	229 KB	337 KB	445 KB	552 KB
40	150 KB	285 KB	421 KB	557 KB	693 KB
50	178 KB	342 KB	506 KB	670 KB	834 KB
75	248 KB	482 KB	717 KB	951 KB	1185 KB
100	318 KB	623 KB	928 KB	1232 KB	1537 KB
200	600 KB	1185 KB	1771 KB	2357 KB	2943 KB
300	881 KB	1748 KB	2615 KB	3482 KB	4349 KB

Total EDM pool space

Use the following variables to calculate EDM pool space for plans, packages, dynamic statements, and DBDs:
- Let *concplans* be the number of concurrently executing plans. This is the sum of the values that are specified for the MAX USERS and MAX REMOTE ACTIVE fields on installation panel DSNTIPE.

- Let *maxplans* be the maximum number of unique plans that you want in the EDM pool at any given time. Estimate this by taking one fourth of the total number of plans.
- Let *plansize* be the average plan size.
- Let *concdb* be the number of concurrent databases, which is specified on installation panel DSNTIPE.
- Let *dbdsize* be the DBD size.

Then, calculate as follows:

```
(concplans + maxplans) * plansize  + (concdb * dbdsize)
```

Then, add 50 KB for overhead.

The default, as calculated by the DSNTINST CLIST, is

```
 ((200 + 200)+ 50) * 25 + (100 * 223) = 33550 KB
```

Then, the CLIST adds 50 KB for overhead to give the result of 33 600 KB.

Data set control block storage calculation

To determine the total number of open data sets (DSMAX):

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

First, calculate the total number of open tables with the following formula:

```
opntab = concdb * tables
```

Then, calculate the number of open data sets for partitioned table spaces with the following formula:

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

Then calculate the number of open data sets with the following formula:

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

You can modify DSMAX by editing job DSNTIJUZ. The maximum number of
concurrently open data sets is 65 041 if you are using z/OS V1R6 or earlier. If you
are using z/OS V1R7 or later, the maximum number of concurrently open data sets
is typically 100 000. See Part 5 (Volume 2) of *DB2 Administration Guide* for
information about tailoring DSMAX for performance.

Important: Your DSMAX calculations might be different. If you use table spaces that are defined with the LARGE parameter, your DSMAX calculations will be larger. Nonpartitioned indexes on a partitioned table space that is defined with the

LARGE parameter can have up to 128 data sets. Additionally, if your average number of open indexes or partitions is higher, your DSMAX will be higher.

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

```
DSMAX * 1.8 KB
```

The default, as calculated by the DSNTINST CLIST, is

```
9960 KB * 1.8 KB = 17928 KB
```

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

Recommendation: Move the scheduler work area (SWA) above the 16-MB line of z/OS virtual storage by using JES initialization statements, JES exit routines, or the SMF exit routine (IEFUJV). This way, you can save approximately 1 KB for each open data set in virtual storage below the 16-MB line and avoid potential storage errors. To determine the amount of storage that is needed below the 16-MB line, use 0.1 KB for the multiplication factor if the SWA is above the line or 1.2 KB for the multiplication factor if the SWA is below the line. The calculations in the CLIST presume that the SWA is above the line. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about improving resource utilization.

Recommendation: If you do not move the SWA above the 16-MB line, you should not use a DSMAX value greater than 5000.

Working storage calculation

Working storage is that portion of main storage, above and below the 16-MB line, that DB2 needs in the database services address space to hold data temporarily. To estimate the amount of working storage that is needed, start with 600 KB. Add 40 KB for each concurrent DB2 user (*concusers*). This value is specified as MAX USERS on installation panel DSNTIPE. Add 40 KB for each remote DB2 user. This value is specified as MAX REMOTE ACTIVE on installation panel DSNTIPE. Follow this formula:

```
600 KB + (MAX USERS + MAX REMOTE ACTIVE) * 40 +
(MAX REMOTE CONNECTED - MAX REMOTE ACTIVE) * 4
```

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

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

If you use dynamic SQL, you need more working storage and less EDM pool space than if you use static SQL. DB2 QMF users have a very small plan in EDM pool, usually 12 KB. Users of static SQL have larger plan sizes as noted above, typically varying from 15 KB to 1 MB. Typical sites would use about 300 KB for each thread of working storage for dynamic SQL users, and 100 KB per thread for static SQL users. A *thread* is a structure that describes an application connection to DB2. For information about thread creation, see Part 5 (Volume 2) of *DB2 Administration Guide*. The CLIST does not include information about open compressed table spaces. Compression dictionaries are stored above the 2-GB bar. Therefore, if you use compressed table spaces, you need additional storage. See Section 1 of *DB2 Administration Guide* for storage information about compressed table spaces.

Virtual storage below the 16-MB line

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

Most of the needed virtual storage is in extended private storage, including the buffer pool, the EDM pool, and almost all of the code and working storage. This is the difference between the total storage and the estimated region size. The region size estimate does not include extended private storage; it includes only the data set control block storage size and some of the code. To estimate the size of storage below the 16-MB line, use the following formula:

```
600 KB + MAX USERS + MAX REMOTE ACTIVE + (DSMAX * 0.01)
```

The default, as calculated by the DSNTINST CLIST, is

```
600 KB + 200 + 200 + (99600 * 0.01 KB), = 1160 KB
```

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

Recommendation: If you use DSMAX of more than 6000, your SWA should be above the 16-MB line.

Table 17 shows total main storage calculation. Place your estimates in the spaces provided in the size column and make the indicated calculations.

Table 17. Main storage size calculation

Category	Your size	Default
EDM pool storage size		33 600 KB
Buffer pool size	+	+ 104 000 KB
Sort pool size	+	+ 2000 KB
RID pool size	+	+ 8000KB
Data set control block storage size	+	+17 928 KB
Code storage size	+ 30 000 KB	+ 30 000 KB
Working storage size	+	+ 55 800 KB
Total main storage size (above 16-MB line)	=	= 252 328 KB
Region size (below 16-MB line) (assume SWA above the line)		1160 KB

The CLIST calculations panel, DSNTIPC, displays storage sizes that are calculated by the DSNTINST CLIST. For more information, see “CLIST calculations panel 1: DSNTIPC” on page 238.

Real storage requirements

DB2 can use real storage to reduce I/O and processor times and to improve response time and throughput. The amount of real storage that DB2 needs varies greatly. Some users find that their organizations need several times the estimates listed below, whereas others need less. The amount of storage is an important parameter in DB2 performance. Performance monitoring programs give you a more accurate estimate of your storage requirements than the formulas in this topic because these programs can take specific environmental characteristics into account.

For the DB2 buffer pools, the EDM pool, and working storage, the amount of real storage must be the same as the amount of virtual storage. Paging activity in the buffers is an indication of a problem. If you do not have enough real storage to hold the buffers, the buffers need to be reduced, which results in fewer concurrent users. You also need space to contain locks, the working set of code in all address spaces, log buffers, and ECSA and CSA space. Because some of the figures that are used in virtual storage calculations are maximums, whereas the real storage figures typically use activity for the peak, changes are needed in the calculations. The virtual storage figures concentrate on the most constrained address space, but real storage work must include them all. For more information about each category, see the information specified in Table 18 and Table 19.

Table 18. Real storage size calculation

Category	**Default**	**Virtual size**	**× Factor = Real size**	**See**
Buffer pools	104 200 KB	_______	1.0 _______	Note [1]
Sort pool	2000 KB	_______	0.5 _______	35
RID pool	8000 KB	_______	0.5 + _______	36
EDM pool	33 600 KB	_______	1.0 + _______	36
Data set size	17 928 KB	_______	0.6 + _______	41
Code size + 1100 (4 address spaces)	30 000 KB	30 000 KB	0.5 + 15 000 KB	
Working storage + (DSCF + DDF)	55 800 KB	_______	1.0 + _______	42
Log buffers	400 KB	_______	0.8 + _______	Note [1]
Lock space	5000 KB	_______	0.4 + _______	Note [1]
CSA/ECSA	2160 KB	_______	0.4 + _______	30
Total real storage size			= _______	

Note: [1] See Part 5 (Volume 2) of *DB2 Administration Guide*

Table 19 shows the default real storage size calculations.

Table 19. Default real storage size calculation

Category	**Virtual size**	**× Factor**	**= Real size**
Buffer pools	104 000 KB	1.0	104 000 KB
Sort pool	2000 KB	0.5	1000 KB
RID pool	8000 KB	0.5	4000 KB
EDM pool	33 600 KB	1.0	33 600 KB
Data set size	17 928 KB	0.6	10 756.8 KB
Code size + 1100 (4 address spaces)	30 000 KB	0.5	15 000 KB
Working storage + (DSCF + DDF)	55 800 KB	1.0	55 800 KB
Log buffers	400 KB	0.8	320 KB
Lock space	5000 KB	0.4	2000 KB
CSA/ECSA	2160 KB	0.4	864 KB

Total real storage size =			324 152 KB

Table 20 uses rough estimates to approximate the amount of additional real storage needed by several kinds of users. If you have more concurrent users, plan to add real storage.

Table 20. Additional real storage for more users

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

Chapter 3. Loading DB2 libraries

IBM distributes DB2 on tapes or cartridges, depending on which feature you order.

If you are installing DB2, your first task is to load the data sets on the distribution tapes or cartridges into DB2 libraries.

If you are migrating to Version 8, you need to check *DB2 Program Directory* to ensure that you are at the proper maintenance level **before** you load the data sets on these tapes or cartridges into DB2 libraries.

To load the DB2 libraries, use System Modification Program Extended (SMP/E). SMP/E processes the installation tapes or cartridges and creates DB2 distribution libraries, DB2 target libraries, and SMP/E control data sets.

DB2 provides several jobs that invoke SMP/E. These jobs are distributed on one of the tapes or cartridges that you receive.

If you ordered a custom-built product delivery offering (CBPDO) or ServerPac, refer to the documentation that is sent with that package for instructions on loading your DB2 libraries. Continue your DB2 installation according to the instructions in Chapter 6, "Installing, migrating, and updating system parameters," on page 79.

The following topics provide additional information:
- "What IBM sends you" on page 48
- "What you produce" on page 49
- "SMP/E step 1: Copy and edit the SMP/E jobs" on page 52
- "SMP/E step 2: Allocate the SMP/E CSI file and SMP/E control data sets: DSNTIJAA (optional)" on page 59
- "SMP/E step 3: Allocate distribution and target libraries: DSNALLOC" on page 60
- "SMP/E step 4: Run the RECEIVE jobs: DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4" on page 61
- "SMP/E step 5: Run the clean-up job for migration: DSNTIJUD (optional)" on page 61
- "SMP/E step 6: Run the apply jobs: DSNAPPL1, DSNAPPL2" on page 62
- "SMP/E step 7: Run the accept jobs: DSNACEP1, DSNACEP2" on page 63
- "SMP/E step 8: Run the receive jobs for the additional FMIDs (optional)" on page 64
- "SMP/E step 9: Run the apply job for the additional FMIDs (optional)" on page 64
- "SMP/E step 10: Run the accept job for the additional FMIDs (optional)" on page 64
- "SMP/E step 11: Ensure installation of proper maintenance" on page 64
- "Post-SMP/E activities" on page 64

What IBM sends you

When you order DB2, you receive 3480 cartridges or 4mm cartridges, depending on the feature you ordered. If you order a custom built product delivery offering (CBPDO), your order may differ.

When you order DB2, it includes: DB2, DB2 REXX Language Support, DB2 Online Help, msys for Setup DB2 Customization Center, IRLM 2.2, JDBC, SQLJ, ODBC, IAV Extenders, Text Extender, and XML Extender.

DB2 Management Clients Package and Net.Data are separately ordered as non-priced features of DB2. DB2 QMF is separately ordered as a priced feature of DB2. DB2 QMF Enterprise Edition includes DB2 QMF for WebSphere®, DB2 QMF for Windows®, DB2 QMF High Performance Option, and DB2 QMF for TSO/CICS.

The DB2 Management Clients Package includes: Visual Explain, DB2 Connect, DB2 Administration Server for z/OS, and DB2 UDB for z/OS Control Center Enablement. The tools are offered to you on a CD-ROM. For more information about these workstation features, see their readme files on the CDs. The DB2 UDB for z/OS Control Center Enablement is shipped on a tape or cartridge. Directions for installing the Control Center Enablement are in the DB2 Management Clients Package Program Directory. For customization information about the Control Center, see *IBM DB2 Connect Quick Beginnings for DB2 Connect Enterprise Edition* and www.ibm.com/software/data/db2/os390/cc390

DB2 REXX Language Support lets you write and run REXX language applications that include SQL statements. For more information about installing DB2 REXX Language Support, see this chapter and *DB2 Program Directory*. For information about using DB2 REXX Language Support, see *DB2 Application Programming and SQL Guide*.

Each tape or cartridge has one or more function modification identifiers (FMIDs) that SMP/E uses to distinguish separate parts of DB2. This arrangement simplifies shipping and service. IRLM, for example, is distributed with both IMS and DB2, and therefore has a separate FMID.

Along with these tapes or cartridges, you receive a set of documents. One of these documents is *DB2 Program Directory*. Read the *DB2 Program Directory* before installing or migrating to a new version of DB2. It identifies and describes the contents of FMIDs for each tape or cartridge. It also describes any additional service that needs to be applied to DB2. You also receive a Program Directory for each feature of the DB2 server that you ordered. Read these directories before installing any of the DB2 features.

If you plan to use the DB2 callable SQL interface (DB2 ODBC), you need to run additional installation jobs. See *DB2 ODBC Guide and Reference* for more information.

If you plan to use DB2 UDB for z/OS Java Edition, see *DB2 Application Programming Guide and Reference for Java* for information about additional installation jobs that you need to run.

Before installing DB2, use Information/Access or the ServiceLink facility of IBMLink to check for PSP updates to the information that is contained in both the

DB2 Program Directory and this book. Refer to *DB2 Program Directory* for PSP keyword specifications. Be sure that you apply all necessary corrective service to your DB2 system before migrating.

Recommendation: Check monthly for PSP updates. This way, you get the most current information about DB2. Contact IBM Software Support if you do not have access to IBMLink.

What you produce

During SMP/E processing, DB2 is loaded into the distribution and target libraries. DB2 uses the distribution libraries to maintain DB2 and contain the master copy of all elements for your DB2 system. The target libraries contain the various DB2 components. DB2 target libraries are updated when you apply corrective service.

Table 21 describes all the DB2 distribution libraries. The distribution libraries contain the master copy of all elements for your DB2 system.

Table 21. DB2 distribution libraries

Distribution libraries	Description
prefix.ADSNBASE	This library contains all jobs that are required to complete SMP/E installation.
prefix.ADSNLOAD	This library contains an individual object module for every DB2 load module. It contains the IRLM load modules if you choose to install IRLM into the same distribution libraries as DB2. This library must be a PDSE data set.
prefix.ADSNLOD2	This library contains a PDSE data set, which contains JDBC and SQLJ DLLs.
prefix.ADSNMACS	This library contains the DB2 macros, sample programs, sample data, initialization data, TSO CLISTs, ISPF panels, and ISPF messages.
prefix.ADSNENU or ADSNDKF	This library contains the DB2 English or Kanji task panels, respectively.
prefix.ADSNIVPD	This library contains the IVP input data and expected output for sample applications.
prefix.ADXRLOAD	This library contains an individual object module for every IRLM load module.
prefix.ADXRSAMP	This library contains the installation procedures for installing IRLM Version 2.
prefix.ADSNHFS	This library contains the data that is to be copied into z/OS UNIX® System Services.
prefix.ADSNBKS *prefix*.ADSNINDX *prefix*.ADSNINST *prefix*.ADSNSHLF	These libraries contain the DB2 Online Help files.

Table 22 describes all DB2 target libraries.

Table 22. DB2 target libraries

Target libraries	Description
prefix.SDSNBASE	This library contains all jobs that are required to complete SMP/E installation.

Table 22. DB2 target libraries (continued)

Target libraries	Description
prefix.SDSNBKS	This library contains the BookManager® books that are used for DB2 Online Help. See Chapter 4, "Setting up and using DB2 Online Help," on page 67 for more information.
prefix.SDSNCLST	This TSO CLIST library contains code that simplifies the process of installing and migrating, aids program preparation and the use of DB2 utilities, and allows the use of DB2 Interactive (DB2I).
prefix.SDSNDBRM	This library contains the system DBRMs for DB2 Version 8.
prefix.SDSNEXIT	This program library is empty when first created. The installation jobs put DSNHDECP, the DSNZP*xxx* subsystem parameters load module, and user exit modules into this library.
prefix.SDSNINDX	This library contains the BookManager index for the books in *prefix*.SDSNINST. See Chapter 4, "Setting up and using DB2 Online Help," on page 67 for more information.
prefix.SDSNINST	This library contains jobs that are used to install DB2 Online Help. See Chapter 4, "Setting up and using DB2 Online Help," on page 67 for more information.
prefix.SDSNLINK	This library contains early code of Version 8.
prefix.SDSNLOAD	This library contains Version 8 load modules. This library must be a PDSE data set.
prefix.SDSNLOD2	This library contains the PDSE data set, which contains JDBC and SQLJ DLLs.
prefix.SDSNMACS	This macro library contains macros that are needed for the CICS and IMS attachment facilities, the initialization parameter macros, and some data-mapping macros that are needed for some applications.
prefix.SDSNSAMP	This initialization library contains the sample applications and data, the jobs for installing and migrating, the default installation and migration parameters, and catalog initialization data for DB2. The JCLIN for each FMID is stored in this library.
prefix.SDSNSHLF	This library contains the BookManager bookshelf for the books in *prefix*.SDSNINST. See Chapter 4, "Setting up and using DB2 Online Help," on page 67 for more information.
prefix.SDSNSPFM	This DB2 ISPF message library contains messages that are issued during install or migrate processing.
prefix.SDSNSPFP	This library is the DB2 ISPF library for installation task and help routing panels.
prefix.SDSNSPFS	This library is the DB2 ISPF skeleton library that is used to produce EDITJCL.
prefix.SDSNSPFT	This library is the DB2 ISPF command table library.
prefix.SDSNPFPE or *prefix*.SDSNPFPK	*prefix*.SDSNPFPE contains the English task and help panels, and *prefix*.SDSNPFPK contains the Kanji task and help panels.

Table 22. DB2 target libraries (continued)

Target libraries	Description
prefix.SDSNC.H	This library contains the header files. The command-line interface (CLI) requires header files. C language application programs can use header files.
prefix.SDSNIVPD	This library contains the IVP input data and the expected output for sample applications.
prefix.SDXRSAMP	The IRLM samples library might be empty if you chose to install IRLM elsewhere.
prefix.SDXRRESL	This library contains the IRLM load modules. This library might be empty if you chose to install IRLM elsewhere.

The remainder of this chapter explains how to edit and run the SMP/E jobs that DB2 provides. These jobs allocate the DB2 libraries and load them with the data from the installation tapes or cartridges.

For a description of each job, see Table 23.

Table 23. List of SMP/E jobs

Job name	Description
DSNTIJAA	This job creates the DB2 target and distribution zones, and defines the SMP/E control data sets in these zones and in the SMP/E global zone.
DSNACEP1	This job invokes SMP/E to accept all the required FMIDs into the DB2 distribution libraries (DLIBs).
DSNACEP2	This job invokes SMP/E to accept all the additional FMIDs into the DB2 DLIBs.
DSNALLOC	This is the SMP/E allocation job. It creates the DB2 target and distribution libraries and defines the libraries in the SMP/E target and distribution zones for DB2 for the required and optional FMIDs.
| DSNMSMKD	This sample job invokes the supplied DSNMMKDR EXEC to allocate HFS paths for msys for Setup DB2 Customization Center
| DSNISMKD	This sample job invokes the supplied DSNMKDIR EXEC to allocate HFS paths for JDBC and SQLJ.
| DMBISMKD	This sample job invokes the supplied DSNMKDIR EXEC to allocate HFS paths for IAV Extenders.
| DESISMKD	This sample job invokes the supplied DSNMKDIR EXEC to allocate HFS paths for Text Extender.
DSNAPPL1	This job invokes SMP/E to apply all the FMIDs to the DB2 target libraries for the required FMIDs.
DSNAPPL2	This job invokes SMP/E to apply all the FMIDs to the DB2 target libraries for the additional FMIDs.
DSNRECV1	This job invokes SMP/E to receive all the required FMIDs (from both tapes or cartridges) from the base tape into the SMP/E control data sets.
DSNRECV2	This job invokes SMP/E to receive all the required FMIDs (from both tapes or cartridges) for IRLM into the SMP/E control data sets.

Table 23. List of SMP/E jobs (continued)

Job name	Description
DSNRECV3	This job invokes SMP/E to receive all the additional FMIDs (from both tapes or cartridges) for ODBC, JDBC, and SQLJ into the SMP/E control data sets.
DSNRECV4	This job invokes SMP/E to receive FMIDs (from both tape or cartridge) for the Kanji DB2I panels into the SMP/E control data sets.
DSNTIJUD	This job invokes SMP/E to delete all Version 7 entries from the SMP/E libraries.

SMP/E step 1: Copy and edit the SMP/E jobs

Before running any of the SMP/E jobs, you must copy them from the tape or cartridge on which they are distributed to a disk that you define. To do this, use the sample JCL that appears in Figure 1. Check *DB2 Program Directory*, which is shipped with the product, for any changes to the contents of the tapes or cartridges.

If you have a CBPDO or ServerPac, refer to the documentation that is sent with the package.

Copying the SMP/E jobs

This JCL invokes the z/OS utility IEBCOPY to copy the jobs to disk. It then invokes the z/OS utility IEBPTPCH to print each job. If you need additional information about these utilities, see *DFSMS/MVS: Utilities*.

```
//STEP1     EXEC PGM=IEBCOPY
//SYSPRINT  DD SYSOUT=*
//IN        DD DSN=IBM.HDB8810.F3,UNIT=tunit, VOL=SER=DB8810,
//          LABEL=(6,SL),DISP=(OLD,KEEP)
//OUT       DD DSNAME=jcl-library-name,
//          DISP=(NEW,CATLG,DELETE),
//          VOL=SER=dasdvol,UNIT=SYSALLDA
//          DCB=*.STEP1.IN,SPACE=(TRK,(20,10,10))
//SYSUT3    DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSIN     DD *
    COPY INDD=IN,OUTDD=OUT
/*
```

Figure 1. Sample JCL to copy SMP/E jobs to disk

Change the lowercase parameters in the sample in Figure 1 to uppercase values that meet your site's requirements before submitting the job. In Figure 1, *tunit* is the unit value that matches the product tape or cartridge, *jcl-library-name* is the name of the data set where the sample jobs are to reside, and *dasdvol* is the volume serial number of the disk device where the data set is to reside.

If this job fails or abends, correct the job and rerun it.

This JCL copies all members that are required to complete SMP/E installation. These jobs exist as members of the partitioned data set IBM.HDB8810.F3 on tape VOLSER=DB8810.

After running the copy job, edit and run jobs DSNALLOC, DSNTIJUD (optional), DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4, DSNAPPL1, DSNACEP1,

DSNAPPL2, DSNACEP2, and DSNMSMKD. Also copy the DSNMMKDR EXEC.
See the header notes within each job for information about how to customize the
job for your particular installation. Do not run DSNRECV2 if your IRLM is at a
higher level. For more information, see *DB2 Program Directory*.

Editing the SMP/E jobs

Before running any of the SMP/E jobs, you must edit them. This topic identifies the items that you might want to modify. The chart below lists each of these items and the location in which a description appears. Read the entire topic before you begin making changes.

Item	Location
JOB statements	53
Link list options	53
DB2 program library	55
DB2 library naming considerations	57
SMP/E data set options	58
IRLM options	59

Creating job statements

The SMP/E jobs do not include JOB statements. Although JOB statements are often built automatically, creating your own JOB statements that are correct for your site is usually easier than editing the provided JOB statements. You have the following choices:

- If you are using ISPF to edit and submit the SMP/E jobs, edit a member that contains the JOB statement. Delete all text except the JOB statement. Then use the ISPF COPY command to copy the member into each job before submitting it.
- If you are using TSO to submit the SMP/E jobs, edit a JOB statement and submit that JOB statement with each job.

  For example, data set JCL.CNTL(J) might contain the following code:

  ```
  //DB2INST   JOB ACCT,NAME,
  //          MSGCLASS=A,MSGLEVEL=(1,1),
  //          TIME=(10),USER=SYSADM,PASSWORD=xxxxxxxx
  /*JOBPARM   ....
  /*ROUTE     PRINT ....
  ```

  When you are ready to submit a job, use a command like the following command:

  ```
  SUBMIT (JCL(J) JCL(DSNTIJxx))
  ```

  In this command, *xx* represents the last two characters of the SMP/E job name. This command submits the JOB statement along with the job.

Choosing link list options

Link list options for the four load module libraries are as follows:

*prefix***.SDSNLINK:** Contains modules that you must place in the link list because they are loaded at subsystem initialization during IPL. For Version 8, the load module library SDSNLINK contains modules that are called early (ERLY) code. If your system is at the prerequisite maintenance level, your Version 7 early code is **upward** compatible with Version 8. The Version 8 early code is **downward** compatible with Version 7.

If you are migrating, be aware that any maintenance to early code or installation of new early code requires that you IPL z/OS to execute the early code. Pointing to SDSNLINK, STEPLIB, LLA REFRESH, or stopping LLA **fails** to update the z/OS subsystem vector table (SSVT). See *DB2 Program Directory* for details.

Schedule a z/OS IPL before or during a migration to a new release of DB2. This IPL is necessary because migration job DSNTIJMV makes changes to SYS1.PARMLIB that are not recognized by z/OS until the next IPL. Changes that DSNTIJMV makes to the SYS1.PARMLIB affect the following libraries:

New subsystem definitions in IEFSSN*xx*
New APF libraries in IEAAPF*xx*
New load module libraries in LNKLST*xx*.

*prefix***.*SDSNLINK:***

Contains early code
can be shared by multiple subsystems and releases of DB2
Is APF-authorized

*prefix***.*SDSNLOAD:***

Contains modules that you can place in the z/OS link list
Is a main load module repository
Can be shared by multiple subsystems at the same release level
Allows only DB2 to modify code
Holds default exit routines
Is APF-authorized
| Must be a PSDE

*prefix***.*SDSNLOD2:*** Contains a PDSE data set, which contains JDBC and SQLJ DLLs.

*prefix***.*SDSNEXIT:***

Contains modules that you can place in the link list
Holds the subsystem parameter module, DSNHDECP, and user-written exit routines
Is modified by user
Is APF-authorized

Libraries *prefix*.SDSNLOAD and *prefix*.SDSNEXIT are separate to allow users who are supporting two levels of DB2 to access modules from either level by using STEPLIB and JOBLIB statements. This also minimizes the number of IPLs that are required by corrective service to DB2 load modules, and it reduces the size of the LNKLST lookaside (LLA) list. When you use both *prefix*.SDSNLOAD and *prefix*.SDSNEXIT, list *prefix*.SDSNEXIT first to override the IBM defaults in *prefix*.SDSNLOAD.

IRLM link list requirement: You must add the IRLM load module DXRRL183 to the link list. This requires that you copy the module into another library if the IRLM load module is not in the link list. After you apply maintenance to IRLM that affects DXRRL183, remember to copy the updated module to the link list.

| ***Integrated Cryptographic Services Facility link list requirement:*** If you plan to use
| encryption with your DB2 subsystem, you must include the Integrated
| Cryptographic Services Facility library that contains the SCSFMOD0 load module
| in the link list. The SQL statement SET ENCRYPTION PASSWORD and the
| following built-in functions require this support:

| • ENCRYPT

- DECRYPT_BIT
- DECRYPT_DB
- DECRYPT_CHAR
- GETHINT

Supporting one DB2 subsystem: You can choose among several methods of maintaining a single DB2 subsystem. The following steps describe what is probably the easiest method for most sites:

1. Change the SMP/E procedure DSNALLOC to assign all load modules to *prefix*.SDSNLOAD. You can do this by changing the data set name for DDDEF (SDSNLINK) from *prefix*.SDSNLINK to *prefix*.SDSNLOAD.
2. Remove the allocation for *prefix*.SDSNLINK from the allocation job DSNALLOC.
3. Include *prefix*.SDSNLOAD (instead of *prefix*.SDSNLINK) in the LNKLST*xx* member of SYS1.PARMLIB.

Supporting multiple DB2 subsystems: Supporting multiple subsystems can mean several things. You can have two or more DB2 subsystems at the same release and service level (for example, two DB2 Version 8 subsystems). Create separate libraries for DSNHDECP and user-written exit routines of each DB2 subsystem. For considerations that affect data sharing environments, see *DB2 Data Sharing: Planning and Administration*.

You can also have two or more DB2 subsystems at the same release level, but at different service levels. For example, you can have a DB2 Version 8 production subsystem and a DB2 Version 8 test subsystem at different service levels. Alternately, you can have two DB2 subsystems at different release levels. For example, you can have a Version 8 subsystem and a Version 7 subsystem.

In either of these cases, you can assign the DB2 modules that must be in the link list libraries to an existing link list data set. To do this, change the data set name for DDDEF (SDSNLINK) in the DSNALLOC procedure to the name of an existing entry in the LNKLST*xx* member of SYS1.PARMLIB. You might still want to have the *prefix*.SDSNLOAD data set listed once in the link list to permit fewer STEPLIB statements. With different SDSNEXIT data sets, you can easily have different subsystem parameter or DSNHDECP members for each subsystem.

The DB2 subsystem must use the appropriate release level of *prefix*.SDSNLOAD, but the application attachment code (for example, CICS, CAF, or TSO Attach) can use code that is either one release level down or one release level up from that of *prefix*.SDSNLOAD. To use application attachment code that is either one level down or one level up from that of *prefix*.SDSNLOAD, place the attachment code in a different STEPLIB data set from the STEPLIB data set that DB2 executes.

Accessing the correct DB2 program library

If you do not place *prefix*.SDSNLOAD in the LNKLST*xx* member of SYS1.PARMLIB, you must provide JOBLIB or STEPLIB statements for it in certain types of programs and procedures.

The installation and migration jobs that are provided with DB2 Version 8 already contain the necessary JOBLIB or STEPLIB statements. In addition, the startup procedures that DB2 provides for Version 7 and Version 8 include STEPLIB statements for their respective program libraries, *prefix*.SDSNLOAD and *prefix*.SDSNEXIT.

Provide STEPLIB or JOBLIB statements for the following types of programs and procedures if you do not place *prefix*.SDSNLOAD in the LNKLST*xx* member of SYS1.PARMLIB.

- **TSO or batch jobs** that access DB2 services require JOBLIB or STEPLIB statements for *prefix*.SDSNLOAD. These jobs include TSO logon procedures and batch jobs that access the DSN command and subcommands, the DB2 precompiler, and DB2 utilities.
- **IMS control, message, and batch processing jobs** also require JOBLIB or STEPLIB statements for *prefix*.SDSNLOAD. You must specify the DB2 load library in the startup procedure for each IMS region (IMS control, message processing program (MPP), batch message processing (BMP), and Fast Path region) that can communicate with DB2. You can do this in two ways:
  – If all the data sets that the JOBLIB or STEPLIB statement refers to for an IMS region are APF-authorized, add the DD statement for *prefix*.SDSNLOAD to the JOBLIB or STEPLIB statement. If you are using the DYNAM option of COBOL, the IMS RESLIB DD statement must precede the reference to *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statement.
  – If any of the data sets that the JOBLIB or STEPLIB statement refers to for the IMS region are not APF-authorized, add the DFSESL DD statement for *prefix*.SDSNLOAD. All libraries that are specified on the DFSESL DD statement must be APF-authorized. The DFSESL DD statement is not required by the DB2 DL/I batch support. IMS requires that an IMS RESLIB DD statement also be referred to by the DFSESL DD statement, as in the following example:
    ```
    //DFSESL     DD      DSN=ims_reslib,DISP=SHR
    //           DD      DSN=prefix.SDSNLOAD,DISP=SHR
    ```
- **CICS procedures**, including the CICS initialization JCL, also need to include DB2 libraries. See *CICS Transaction Server for z/OS DB2 Guide* for more information.

Performance considerations

This topic discusses performance considerations for including modules in the libraries that are included in the link list and presents some suggestions on the strategies you might want to consider. These general suggestions might not match the specific needs of your site.

Adding many modules to the libraries that are included in the link list can reduce system performance. However, adding only a few modules to the libraries requires additional STEPLIB or JOBLIB statements. Because these STEPLIB or JOBLIB statements must be searched before the link list is searched, this approach can also reduce system performance. The approach that produces the best performance for your site depends on the environment in which you use DB2. Regardless of which attachment facilities you use, the modules in *prefix*.SDSNLINK must always be in the link library list.

If you are using DB2 with IMS, you should probably include *prefix*.SDSNLINK, not *prefix*.SDSNLOAD, in the LNKLST*xx* member of SYS1.PARMLIB because both the IMS RESLIB and *prefix*.SDSNLOAD have the DSNHLI alias. Place the needed STEPLIB or JOBLIB statements in the IMS procedures.

If you are using DB2 with IMS and you want *prefix*.SDSNLOAD (in addition to *prefix*.SDSNLINK) in the LNKLST*xx* member of SYS1.PARMLIB, ensure that the library concatenation for *prefix*.SDSNLOAD and the IMS RESLIB are correct for your site because both libraries have the DSNHLI alias.

If you are using DB2 with CICS, you should probably to put *prefix*.SDSNLINK, not *prefix*.SDSNLOAD, in the LNKLST*xx* member of SYS1.PARMLIB. Then place the needed STEPLIB or JOBLIB statements in the CICS procedures.

The approach for using the TSO, RRS attachment, and call attachment facilities involves the following considerations:

- If you use the DSN command and its subcommands infrequently, place **only** *prefix*.SDSNLINK in the LNKLST*xx* member of SYS1.PARMLIB. Provide the necessary STEPLIB or JOBLIB statements in your TSO logon procedures or in your JCL if you are using batch.
- If you use the DSN command and its subcommands frequently, you might also want to move the TSO attachment facility load modules to a library that is defined in the LNKLST*xx*. The TSO attach modules are DSNECP00, DSNECP10, DSNESM00, and DSNELI.
- If you use the call attachment facility (CAF) frequently, move the CAF load modules (DSNACAB, DSNACAF, and DSNALI) to a library that is defined in the LNKLST*xx*.
- If you use the RRS attachment facility (RRSAF) frequently, move the RRSAF load modules (DSNARRS and DSNRLI) to a library that is defined in the LNKLST*xx*.
- If you use the CAF, RRSAF, or the DSN command and its subcommands frequently, you should probably move the eligible load modules to a library that is defined in the link pack area (LPA), in IEALPA*xx* member of SYS1.PARMLIB. The CAF and DSN load modules must reside below the 16-MB line of z/OS virtual storage.
  - The TSO load modules that you can place in the LPA are DSNECP00, DSNECP10, DSNESM00, and DSNELI. If you include these modules in the LPA, remember to include the appropriate aliases for DSNECP00 (DSN) and DSNELI (DSNHLI).
  - The CAF load modules that you can place in the LPA are DSNACAF and DSNALI. If you include these modules in the LPA, remember to include the appropriate alias for DSNALI (DSNHLI2). Do not include DSNACAB in the LPA because it is a data-area-only, non-executable load module.
  - The RRSAF load modules that you can place in the LPA are DSNARRS and DSNRLI. If you include these modules in the LPA, you must include the appropriate alias for DSNRLI (DSNHLIR).

Attention: If modules are moved or copied from one library to another, you must make changes to SMP/E control data to reflect the movement. If you do not make these changes, future service or changes to the modules will not be processed correctly.

DB2 library naming considerations

You need to modify the DB2 library data set names in the SMP/E jobs. These data sets are listed in Table 21 on page 49. Their names are composed of three parts:

- A user-defined prefix
- A fixed base name: for example, SDSNLOAD
- An optional user-defined suffix

The Version 8 default prefix (*prefix*) is used in this book; the default suffix is null. You need to edit each of the DB2 SMP/E jobs and follow the directions in the header notes of each job to specify the names of the SMP/E data sets. If you want to add a suffix, edit the SMP/E procedures and allocation jobs. The prefix cannot exceed 18 characters. The suffix cannot exceed 17 characters, minus the length of the prefix. In addition, any data set names that exceed eight characters must be in groups of no more than eight characters, separated by periods. The qualified data set name cannot exceed 44 characters.

You can also change the base name of these libraries or load them into another data set. If you do this, however, you might need to do additional editing of the installation or migration jobs. The DSNTINST CLIST, which you use later to tailor the installation and migration jobs, uses the following default data set names:

prefix.ADSNLOAD	*prefix*.SDSNMACS
prefix.SDSNCLST	*prefix*.SDSNSAMP
prefix.SDSNEXIT	*prefix*.DBRMLIB.DATA
prefix.SDSNLINK	*prefix*.RUNLIB.LOAD
prefix.SDSNLOAD	*prefix*.SRCLIB.DATA
prefix.SDSNDBRM	*prefix*.SDSNIVPD
prefix.SDXRRESL	*prefix*.SDSNC.H

Recommendation: Use the supplied naming convention.

Document any changes you make to the library names in the SMP/E jobs. You must specify these library names again during the ISPF tailoring session.

SMP/E data set options

You have several options regarding how you establish and use SMP/E data sets. You must decide whether you want DB2 and IMS to share SMP/E data sets. You must also decide whether you need an additional set of SMP/E data sets. An additional set of SMP/E data sets is required if you are supporting more than one release of DB2.

Sharing SMP/E data sets with IMS: If you do not share SMP/E data sets with IMS, skip this topic and continue with 59.

DB2 and z/OS cannot share SMP/E data sets because some module names and macro names are common to both products. Under certain conditions, however, DB2 can share SMP/E data sets with IMS.

The allocation job that you run, DSNALLOC, defines a new set of SMP/E data sets that DB2 and IMS are to share.

You must modify your allocation job for either of the following situations:

Situation 1: You decide to have separate SMP/E data sets for DB2 and IMS. In certain situations, DB2 and IMS cannot share SMP/E data sets. You must have separate SMP/E data sets if you want to have two IRLMs.

Even if you are not **required** to have separate SMP/E data sets, you might want to keep them separate If DB2 and IMS share the SMP/E data sets, you need to accept or reapply DB2 corrective service to these data sets to allow IMS SYSGENs.

To establish separate SMP/E data sets for DB2 and IMS, change the data set prefix that your allocation job uses to a value other than the prefix that you use for your current IMS SMP/E data sets. The allocation jobs use the prefix *IMS*. Changing this prefix prevents the allocation job from replacing your current SMP/E data sets and still allows it to create new SMP/E data sets.

Situation 2: You decide to share SMP/E data sets between DB2 and IMS, but you want to use the SMP/E data sets that already exist for IMS. To do this, remove the data set allocation and initialization statements from your allocation job. When you run the job, no SMP/E data sets are created, and DB2 then shares the existing SMP/E data sets with IMS.

For additional information about sharing data sets, refer to *OS/390 SMP/E User's Guide*.

Establishing SMP/E data sets for two releases: A single set of SMP/E zone structures can record only one release of DB2. Maintaining separate zone structures for both Version 7 and Version 8 is strongly recommended until you are sure that you will not need to fall back. The SMP/E jobs that are provided with DB2 assume that you will allocate a new set of SMP/E data sets for the new release. When you run your allocation job (DSNALLOC), it creates a set of SMP/E data sets. If you choose to reuse your Version 7 zone structure, you can run job DSNTIJUD to delete SMP/E data for Version 7. However, after you run this job, you cannot fall back.

You can create an additional set of SMP/E data sets either by copying them from a prior release of DB2 or by allocating a new set. Allocating a new set is faster because no data must be deleted.

Recommendation: Copy a prior set so that you can then perform service regression checking.

IRLM options

The SMP/E prefix in the SMP/E jobs is the same for the new IRLM as for the old IRLM. Consequently, if you do not change the SMP/E prefix, the jobs overwrite your old IRLM. If you do not want to do this, edit the jobs accordingly.

SMP/E step 2: Allocate the SMP/E CSI file and SMP/E control data sets: DSNTIJAA (optional)

This job creates the SMP/E consolidated software inventory (CSI) file and SMP/E data sets and allocates them to SMP/E. DSNTIJAA also creates the DB2 target and distribution zones. DSNTIJAA is required only if these objects are not created and allocated to the SMP/E global zone. Depending on how you set up your systems, you might need to contact a z/OS system programmer to help you manage some of the SMP/E data sets.

For each group of data sets, DSNTIJAA requires a data set prefix and a volume name on which to allocate the data sets. These names are called the *allocation job parameters*.

Change the allocation parameters according to the decisions that you made regarding the SMP/E data set options. Table 24 lists the allocation job parameters (prefix and volume) for each of the groups of data sets.

Table 24. Allocation job parameters for SMP/E control data sets

Parameter type	Search strings
Prefix	?SMPPRE?
TLIB prefix	?TLIBPRE?
Volume	?SMPVOL?
TLIB volume	?TLIBVOL?
Middle-level qualifier	?SMPMLQ?

If you are using JES3, you must split job DSNALLOC into two jobs. A comment line in the code indicates where to split the job.

Examine the following items in the job you are using, and make any necessary modifications:

Space allocations: The space that is allocated for the SMP/E history log data sets is rather large. The DDNAME for this data set is SMPLOG; its default name is IMSVS.HLDS. If you do not want to retain this log information, remove the data set allocations for DDNAMEs SMPLOG and SMPLOGA from steps ALLOC and INITSMP of job DSNTIJAA. In step INITSMP, you also need to specify DA(NULLFILE) in the DDDEFs for SMPLOG and SMPLOGA.

The ?TLIBVOL? parameter defines the location of the SMPTLIB data sets. The volume on which these data sets reside must have at least 35 MB (1 MB=1048576 bytes) of free space. That is about 37 cylinders on a 3390 and 49 cylinders on a 3380.

The block size is specified as 19069 for unformatted data sets and is determined by the system for other data sets.

Important: AVGREC requires that SMS be active; however the data sets do not need to be allocated on SMS-managed storage devices.

SREL and DSSPACE: The allocation jobs specify an SREL of P115 for the SMP/E data sets. Do **not** change this. They also specify DSSPACE to be (400,400,1200). This is a minimum; change it only if you need to increase it.

SMP/E zone structure: SMP/E zone structures are discussed in *OS/390 SMP/E User's Guide*. You can choose to use a different zone structure from the one that is shown in DSNTIJAA.

SMP/E step 3: Allocate distribution and target libraries: DSNALLOC

This job creates the DB2 target and distribution libraries and defines them in the SMP/E target and distribution zones for DB2. DSNALLOC also creates the target and distribution libraries for DB2 Online Help (FMID HDB881A). If you do not want to install DB2 Online Help, you can delete the JCL in DSNALLOC that creates the ADSNBKS, ADSNINDX, ADSNINST, ADSNSHLF, SDSNBKS, SDSNINDX, SDSNINST, and SDSNSHLF libraries.

Do not modify the data definition statements for SDSNCLST, SDSNLOAD, or SDSNSAMP. The SDSNCLST and SDSNSAMP libraries must be defined as
| partitioned data sets (PDSs). The SDSNLOAD library must be defined as a PDSE.

For each group of data sets, DSNALLOC requires a data set prefix and a volume name on which to allocate the data sets. These names are called the *allocation job parameters*.

You need to change the allocation parameters according to the decisions that you made regarding the LNKLST option and library definition. Table 25 on page 61 lists the allocation job parameters (prefix, volume, and unit name) for each of the groups of data sets.

Table 25. Allocation job parameters for DB2 distribution and target libraries

Data sets	Parameter type	Search strings
DB2 distribution libraries	Prefix volume	?DLIBPRE? ?DLIBVOL?
DB2 target libraries	Prefix volume	?TARGPRE? ?TARGVOL?

Important: After you run job DSNALLOC, run job DSNMSMKD for access to msys
for Setup DB2 Customization Center. See the header notes in job DSNMSMKD for
information about how to customize the job for your particular installation.

SMP/E step 4: Run the RECEIVE jobs: DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4

Before you run the next three jobs, create backups of your DB2 Version 7 distribution and target libraries and your SMP/E data sets. You might need them if you need to fall back. If one of these three jobs fails, you probably need to delete and reallocate data sets or compress them before rerunning the job that failed. When rerunning one of these jobs, delete or comment out the parts that ran successfully, and rerun those parts that failed.

The SMP/E RECEIVE jobs load the DB2 program modules, macros, procedures, IRLM, ODBC, JDBC, SQLJ, and DB2I Kanji into temporary data sets (SMPTLIBs). If this job fails or abends, correct the problem and rerun the job.

Examine the RECEIVE job before you run it. The ?SMPPRE? and ?SMPMLQ? parameters must have the same definition as in the allocation job. The SYSOUT class is defined as the same class as the job's MSGCLASS parameter.

At this point, you might want to run an SMP/E APPLYCHECK job to determine any service and any USERMODs that might be regressed by the following jobs.

| Use the IRLM (FMID HIR2220) that is shipped as part of DB2 Version 8. If the IRLM that is already installed on your system is at a higher maintenance level than the IRLM that is shipped with DB2 Version 8, you can remove the HIR2220 step from job DSNRECV2. If you use the same level of IRLM code for your IMS and DB2 subsystems, use separate SMP/E zones or SMP/E control data sets. Using down-level IRLM code increases your IRLM service activity and is not recommended.

If you do not want to install DB2 Online Help, remove FMID HDB881A from the list of FMIDs that DSNRECV1 is to receive.

SMP/E step 5: Run the clean-up job for migration: DSNTIJUD (optional)

Recommendation: You can avoid running this job by using new SMP/E zones for your migration. If this is not possible (if you are installing Version 8 in the same SMP/E libraries in which you installed Version 7), you must run job DSNTIJUD. Run job DSNTIJUD before the SMP/E APPLY (jobs DSNAPPL1 and DSNAPPL2). Running job DSNTIJUD is not necessary if you are installing DB2 for the first time. If you accidentally run it, it will have no adverse effect.

Run job DSNTIJUD if you are migrating from Version 7 to ensure that delete processing is done properly before installing Version 8. It performs necessary

SMP/E cleanup by deleting all entries from Version 7 in the SMP/E target and distribution libraries. However, this job does not clean up the global zone. Issue the SMP/E REJECT command to remove entries from and clean up the global zone.

| Use the IRLM (FMID HIR2220) that is shipped as part of DB2 Version 8, unless you have a higher maintenance level of IRLM already installed on your system. If you want to use different levels of IRLM for your IMS and DB2 subsystems, you must have different IRLM levels in different SMP/E zones or SMP/E control data sets. Using the down-level IRLM increases your IRLM service activity and is not recommended.

Examine the DSNTIJUD job before you run it. The ?SMPPRE? and ?SMPMLQ? parameters must have the same definition as in the allocation job. The SYSOUT class is defined as the same class as the MSGCLASS parameter of the job.

Attention: If DB2 shares the same CSI with any CICS or ISPF products, delete the following statements from this job before executing:
- DEL MOD(DFHEAI)
- DEL MOD(DFHEAIO)
- DEL MOD(ISPLINK)

SMP/E step 6: Run the apply jobs: DSNAPPL1, DSNAPPL2

The SMP/E APPLY jobs, DSNAPPL1 and DSNAPPL2, copy and link-edit the DB2 program modules, macros, and procedures into the DB2 target libraries for required FMIDs and additional FMIDs, respectively.

Examine the jobs before you run them. The ?SMPPRE? and ?SMPMLQ? parameters must have the same definition as in the allocation job. The SYSOUT class is defined as the same class as the MSGCLASS parameter of the job.

The APPLY statement contains the CHECK parameter, which allows you to verify the APPLY without committing it. When you want to commit the APPLY, remove the CHECK parameter and rerun the APPLY job.

Expect a return code of 4 from this job. You might receive warning messages GIM23913W, GIM61903W, IEW2454W, and IEW2646W during execution of DSNAPPL1. The SQLCA1 and SQLCA2 references are resolved when DSNHFT is included in a Fortran application. If this job fails or abends, correct the problem and rerun the job. If you plan to use the DB2 call level interface (CLI), see *DB2 ODBC Guide and Reference* for information on the CLI FMID.

If you do not apply the FMIDs in a single APPLY statement as DSNAPPLx does, use the following order:
1. HIY8810, HIZ8810, and HBD8810 together
2. HIR2220

If the IRLM (FMID HIR2220) on your system is a more current release or has had maintenance applied so that it is at the same or higher maintenance level than the one that is shipped with DB2, remove FMID HIR2220 from the APPLY step of job DSNAPPL1.

If you do not want an element, remove the FMID of the element from the list of FMIDs that are to be applied by DSNAPPL2. Table 26 identifies the FMIDs for the different products.

Table 26. Product FMIDs

Product name	FMID
Online Help	HDB881A
JDBC/SQLJ	JDB8812
ODBC	JDB8817
IAV Extenders	JDB881B
Text Extender	JDB881C
XML Extender	JDB881X

SMP/E step 7: Run the accept jobs: DSNACEP1, DSNACEP2

The SMP/E ACCEPT jobs, DSNACEP1 and DSNACEP2, copy the program modules, macros, and procedures into the DB2 distribution libraries for the required FMIDs and additional FMIDs, respectively. This action allows you to apply corrective service later. If you do not want the DB2 components to be copied into the distribution libraries, do not run job DSNACEPx.

Examine the ACCEPT job before you run it. The ?SMPPRE? and ?SMPMLQ? parameters must have the same definition as in the allocation job. The SYSOUT class is defined as the same class as the MSGCLASS parameter of the job.

The ACCEPT statement contains the CHECK parameter, which allows you to verify the ACCEPT job without committing it. When you want to commit the ACCEPT job, remove the CHECK parameter and rerun the accept jobs.

If this job fails or abends, correct the problem and rerun the job.

If the IRLM (FMID HIR2220) on your system is a more current release than the one that is shipped with DB2 or has had maintenance applied so that it is at the same or higher level of maintenance, remove FMID HIR2220 from the ACCEPT step of job DSNACEPx.

If you do not want an element, remove the FMID of the element from the list of FMIDs that are to be accepted by DSNACEPx. Table 27 identifies the FMIDs for the different products.

Table 27. Product FMIDs

Product name	FMID
Online Help	HDB881A
JDBC/SQLJ	JDB8812
ODBC	JDB8817
IAV Extenders	JDB881B
Text Extender	JDB881C
XML Extender	JDB881X

SMP/E step 8: Run the receive jobs for the additional FMIDs (optional)

This step is optional unless you plan to use any of the additional FMIDs. To use the additional FMIDs, run the corresponding receive job that is listed in Table 28. Use "SMP/E step 4: Run the RECEIVE jobs: DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4" on page 61 as a guide to help you with this job.

Table 28. Product FMIDs

Product name	FMID	Job name
Online Help books and BookManager bookshelves	HDB881A	DSNRECV1
DB2 Kanji panels	JDB8811	DSNRECV4
JDBC/SQLJ	JDB8812	DSNRECV3
ODBC	JDB8817	DSNRECV3
IAV Extenders	JDB881B	DMBRECEV
Text Extender	JDB881C	DESRECEV
XML Extender	JDB881X	DXXRECEV

SMP/E step 9: Run the apply job for the additional FMIDs (optional)

This step is optional unless you plan to use the additional FMIDs. To use the additional FMIDs, change job DSNAPPL2 to remove any unwanted FMIDs. Use "SMP/E step 6: Run the apply jobs: DSNAPPL1, DSNAPPL2" on page 62 as a guide to help you with this job.

SMP/E step 10: Run the accept job for the additional FMIDs (optional)

This step is optional unless you plan to use the additional FMIDs. To use the additional FMIDs, change job DSNACEP2 to remove any unwanted FMIDs. Use "SMP/E step 7: Run the accept jobs: DSNACEP1, DSNACEP2" on page 63 as a guide to help you with this job.

SMP/E step 11: Ensure installation of proper maintenance

If you are migrating, your Version 7 subsystem must be at the proper maintenance level **before** migrating to Version 8. Refer to *DB2 Program Directory* for information about the proper maintenance level.

Post-SMP/E activities

Each of the display language control techniques that this topic describes is a way to set or change the current allocation of the DDNAME ISPPLIB. If an ISPPALT allocation exists, ISPF uses it instead of an ISPPLIB allocation.

Logon procedures: To switch languages, you need to change only the data set allocation that is currently in effect under the standard ISPF panel library DDNAME. A user's logon procedure can allocate DDNAME ISPPLIB to select the current display language. Here is an example of a logon procedure:

```
//*                            THIS VERSION DISPLAYS ENGLISH PANELS */
//ISPPLIB  DD DSN=prefix.SDSNSPFP,DISP=SHR     ENGLISH TASK
//         DD DSN=prefix.SDSNPFPE,DISP=SHR     ENGLISH DB2I

//*                           THIS VERSION DISPLAYS JAPANESE PANELS */
//ISPPLIB  DD DSN=prefix.SDSNSPFP,DISP=SHR     ENGLISH TASK
//         DD DSN=prefix.SDSNPFPK,DISP=SHR     KANJI
```

Language-switching CLISTs: You can use an ordinary CLIST (outside of ISPF) to free and reallocate ISPPLIB. Here is an example of a CLIST:

```
/* Execute this CLIST outside of ISPF  */
PROC 0 LANGUAGE(E)
FREE DD(ISPPLIB)
WRITE DO YOU WANT ENGLISH OR JAPANESE PANELS:  Enter E or J.
READ &LANGUAGE;
IF &LANGUAGE = E +
  THEN ALLOC DD(ISPPLIB) DS('DSN810.SDSNSPFP' 'DSN810.SDSNPFPE') +
       SHR /*ENGLISH*/
  ELSE ALLOC DD(ISPPLIB) DS('DSN810.SDSNSPFP' 'DSN810.SDSNPFPK') +
        SHR /*JAPANESE*/
END
```

Some users allocate the ISPF panel library from their DEFAULT CLIST. Allocation of DDNAME ISPPLIB controls the current language just as it does for the logon procedure.

If you are falling back to Version 7, change your logon procedures or CLISTs that use the Kanji feature to point to the Version 7 libraries.

Chapter 4. Setting up and using DB2 Online Help

The online help for DB2 UDB for z/OS Version 8 lets you select information in an online DB2 book.

To use DB2 Online Help, complete the set-up instructions in this chapter and then invoke the DSNTINS0 CLIST.

If you choose not to use DB2 Online Help, invoke the DSNTINST CLIST as described in "Invoking the CLIST" on page 81.

Before you install DB2 Online Help, you must first install BookManager READ. See *BookManager READ/MVS V1R3: Installation Planning & Customization* for information about how to do this.

After you install BookManager READ, see the following topics for the information that you need to install, customize, and use online help:
- Copying the bookshelf list, index, and books
- Changing DB2 Online Help book data set names (optional)
- Customizing BookManager READ for DB2 Online Help (optional)
- Verifying DB2 Online Help and setting exit options
- Making DB2 Online Help accessible from installation and DB2I panels
- Using DB2 Online Help

Copying the bookshelf list, index, and books

The bookshelf list, index, and books are shipped as partitioned data sets, but you must copy the contents to sequential data sets before you can use them.
1. Submit the sample JCL DSNUNL1 to copy the bookshelf list to a sequential data set.
2. Submit the sample JCL DSNUNL2 to copy the bookshelf, index, and books to sequential data sets.

DSNUNL1 and DSNUNL2 are in data set DSN810.SDSNINST.

Changing DB2 Online Help book data set names (optional)

Copies of the following DB2 books are shipped with DB2 Online Help:
- *DB2 Application Programming and SQL Guide*
- *DB2 Command Reference*
- *DB2 Installation Guide*
- *DB2 Utility Guide and Reference*

The default names for those books are DSNHELP.*book-ID*.BOOK. You can change the high-order qualifier of the book data set names, but if you do, you must indicate to DB2 what the new data set names are. During installation, change the data set names on installation panel DSNTIPA0. If you use DB2I, change the data set names on panel DSNEIPA0.

To access these panels:

- For installation, start the installation procedure by running CLIST DSNTINS0. See Chapter 6, “Installing, migrating, and updating system parameters,” on page 79 for more information.
- For DB2I, specify YES in field CHANGE HELP BOOK NAMES? in the DB2I Defaults panel. See Part 5 of *DB2 Application Programming and SQL Guide* for more information.

After you enter the data set names in panel DSNTIPA0 or DSNEIPA0, DB2 saves them so that you need to modify the panel again only if you change the data set names. For example, if you want DB2 Online Help to access the latest versions of the DB2 books, you can copy those books from the DB2 UDB for z/OS Version 8 Licensed Online Library CD-ROM to z/OS sequential data sets. Next, access DSNTIPA0 or DSNEIPA0, and enter the new book data set names.

Customizing BookManager READ for DB2 Online Help (optional)

You can perform the following actions to make BookManager READ and DB2 Online Help work together better:

- Add the DB2 Online Help library to the BookManager library list.

  This action lets you select DB2 Online Help books from the BookManager library list, as well as from the installation CLIST and DB2I task panels. To add the DB2 library:

  1. Enter BOOKMGR from TSO option 6. BookManager READ logo information in displayed.
  2. Press Enter. A bookshelf list is displayed.
  3. Select BOOKS on the action bar. The BOOKS pull-down is displayed.
  4. Select PERFORM FILE FUNCTIONS. The PERFORM FILE FUNCTIONS window is displayed.
  5. Select ADD in the FUNCTION TO PERFORM field. The ADD BOOKSHELF window is displayed.
  6. Type the DB2 bookshelf data set name (default= 'DSNSH*nnn*.BKSHELF') in the DATA SET NAME field, and press Enter. The BOOKSHELF LIST DATA SET TO BE MODIFIED window is displayed. The DATA SET NAME field contains the name of your personal bookshelf list.
  7. Replace the bookshelf list data set name in the DATA SET NAME field with the data set name of your site’s system bookshelf list. This data set name is specified in option QLSHELF of the EOXVOPTS member in the SEOYCLIB target library.

     The BOOKSHELF LIST DATA SET TO BE MODIFIED window is removed, and the bookshelf is now in the list.
  8. Press **PF5** to refresh the bookshelf list. The DB2 bookshelf is now on the list.
- Suppress the BookManager logo information so that the logo does not display each time that you enter DB2 Online Help.

  To suppress the logo information, modify the EOXVSTRT member of the SEOYCLIB target library by adding the following line after the trace command at the beginning of the file:

  ```
  ISPEXEC CONTROL NONDISPL ENTER     /* Suppress logo information     */
  ```
- Build a searchable bookshelf.

  The bookshelf shipped on the tape is not enabled for cross-book searches. To build a searchable bookshelf:

  1. From the bookshelf list, open the DSNSH*nnn* bookshelf. A list of DB2 books is displayed.

2. Select SEARCH on the action bar. The SEARCH menu is displayed.
3. Select option 2 ’Set up search ...’. The following error message is displayed:

```
Books  View  Search  Group  Options  Help
----------------------------------------------------------------------
                       Severe Read or Search Error

A serious BookManager error has occurred while reading or searching the
current bookshelf or book. Processing for the current bookshelf or book
is terminated.

Reason . . : Bookshelf does not match bookshelf search index
Data set . : 'DSNHELP.DSNSHnnn.BKSHELF'
Routine  . : EOXMPLSH:plsh_get_string
Code . . . : 9


F1=Help F12=Cancel
----------------------------------------------------------------------
```

4. Press Enter. Choose option 1 on the following dialog:

```
Books  View  Search  Group  Options  Help
----------------------------------------------------------------------
                          Rebuild Bookshelf

  The bookshelf does not match its bookshelf search index. Select
  one of the following actions, and then press ENTER.

      1   1. Rebuild the bookshelf
          2. Do not rebuild the bookshelf


  Data set name  . : 'DSNHELP.DSNSHnnn.BKSHELF'



  F1=Help F12=Cancel
```

5. The following message appears:

```
The bookshelf data set 'DSNHELP.DSNSHnnn.BKSHELF' has been updated to
 correspond with its search index. You must refresh the bookshelf to use it.
```

6. Exit the bookshelf, and select **PF5** to refresh.

Verifying DB2 Online Help and setting exit options

Before installing DB2, you must verify that DB2 Online Help is set up correctly. You can also turn off the confirming messages that appear when you exit an online book.

To verify that DB2 Online Help is installed correctly: From the TSO command option, enter:

```
BOOKMGR
```

The bookshelf list is displayed, as in this example:

```
  Books  View  Options  Help
-------------------------------------------------------------------------
Command ===> ________________________________________________ SCROLL ===>

                                Bookshelf List
                                                        Shelves 1 to 1
   Shelf Name  Description
__ DSNSHnnn    DB2 V8 BOOKS FOR ONLINE HELP
```

If the BOOKMGR command abends, verify that you have concatenated the BookManager data sets correctly in the LINKLIST, in the TSO logon procedure, and in any user-supplied ISPF startup CLISTs.

To set the exit options:

1. Select OPTIONS on the action bar. The OPTIONS pull-down is displayed.
2. Select SET EXIT OPTIONS. The SET EXIT OPTIONS window is displayed.
3. Turn off the exit confirmations and bookmarks, as in this example:

```
                        Set Exit Options

   Exit from book  . . . . . 2   1. Confirm exit
                                 2. Do not confirm exit

   Setting of bookmark . . . 3   1. Keep current closing bookmark
                                 2. Place the closing bookmark
                                 3. Exit without closing bookmark

   ----------------------------------------------------------------

   Exit from bookshelf and
   bookshelf list  . . . . . 2   1. Confirm exit
                                 2. Do not confirm exit

   ----------------------------------------------------------------

   Save the changes as . . . 1   1. Permanent
                                 2. Temporary
```

4. Press Enter to save the changes.
5. Press the exit key (**PF3**) to exit online help.

Making DB2 Online Help accessible from installation and DB2I panels

To make DB2 Online Help accessible from the installation and DB2I task panels, concatenate the DB2 ISPF command table library (*prefix*.SDSNSPFT) to the ISPTLIB DD statements in your TSO logon procedures and in any of the CLISTs where ISPTLIB is allocated.

DB2 Online Help is available in English only. If you want to use DB2 Online Help
for the DB2 installation panels and use the Kanji help panels for DB2I, you need to
delete or rename member DSNECMDS in the prefix.SDSNSPFT library.

Using DB2 Online Help

DB2 Online Help links you directly from DB2 panels to the information that you need in the DB2 online books. Because DB2 Online Help uses BookManager READ to display the help text, you can perform any BookManager function after online help opens a DB2 book. This topic explains how to access DB2 Online Help from installation or DB2I panels and how to navigate within the books.

Accessing DB2 Online Help

To get help for a DB2 installation or DB2I panel, press the Help PF key (usually **PF1**). A selection panel for help topics appears. Type the number of the topic that you want help on and press Enter. DB2 Online Help starts and opens a DB2 online book to the topic.

Navigating through DB2 Online Help

DB2 Online Help lets you navigate through the online book in several ways:

- Press **PF8** to go forward one panel and **PF7** to go backward one panel.
- Text that contains hypertext links is highlighted; move the cursor to the highlighted area, and press Enter to link to related information about the topic.
- Graphics have a PICTURE label with a number. Move the cursor to the label, and press Enter to display the graphic. Press **PF3** to return to the text. If you do not have GDDM installed, you cannot view graphics.
- Figures and tables might be wider than the display screen. Enter `RIGHT` on the command line to scroll to the right of the figure or table; enter `LEFT` to return to the left margin.

Searching for additional information

To search the book that you are currently viewing, select **Search** on the action bar, and then select SET UP SEARCH in the SEARCH pull-down. Type the information that you want to find in the SEARCH FOR area of the SET UP SEARCH window. Your search request can be up to 44 characters long, and it can include any combination of words, phrases, and special characters.

Selecting the search match type: The default search match type is *fuzzy matching*. With fuzzy matching, online help looks for words that share the same language root as words in your search request, such as the plural forms of nouns or different tenses of verbs. You can specify:

- *exact matching, any case* when you want to find the exact words that you type in the SET UP SEARCH window regardless of capitalization
- *exact matching, including case* to find the exact words including capitalization and punctuation

To change the search match type:

1. Select SEARCH on the action bar.
2. Select SET UP SEARCH in the SEARCH pull-down.
3. Press **PF2** in the SET UP SEARCH window.
4. Move the cursor to the type of search matching that you want to use.
5. Take one of these actions:
   - Press Enter to change the search match type temporarily.
   - Press **PF2** to set the search match type as your new default.

Tailoring your search request: To tailor your search request, you can:

- Include a space between words, as in `white house`. A topic matches if it contains both words separated by any number of spaces in the text, but not by punctuation.
- Use a phrase separator. Type the separator character after each word or phrase you want to separate. The default separator character is a comma. A topic matches if it contains either word.
- Use a pattern-matching character. Type the pattern character in place of characters at the end of the word, as in `hous*`. The default pattern character is an asterisk. Words that start with the specified characters match.

Searching for variations and synonyms: You can also search for variations and synonyms of words. To search for spelling variations:

1. Type your search request in the SET UP SEARCH window.
2. Move the cursor to a word in the request, and press **PF4**.
3. After a list of spelling variations for that word is displayed in the WORDCHECK window, you can:
   - Add words to your search request by typing a slash (/) next to each word that you want to include and pressing **PF4**.
   - Replace a word in your search request by typing a slash (/) next to the word that you want to use and pressing **PF5**.
4. Press Enter to search the book.

To search for synonyms:

1. Type your search request in the SET UP SEARCH window.
2. Move the cursor to a word in the request, and press **PF5**.
3. After a list of synonyms for that word is displayed in the SYNONYMS window, you can:
   - Add words to your search request by typing a slash (/) next to each word that you want to include and pressing **PF4**.
   - Replace a word in your search request by typing a slash (/) next to the word that you want to use and press **PF5**.
4. Press Enter to search the book.

Working with your search matches: When you search a book, DB2 Online Help displays a list of the topics that contain information that matches your search request. DB2 Online Help displays the topics in the list by location, frequency, size, exactness, uniqueness, and similarity. The topics that appear first in the list are those with the highest ranking.

- To see the context of the match, press **PF4** to see a line of text beneath each topic entry that shows where the best search match in the topic occurred.
- To see an explanation of why a topic matches your search request, move the cursor to the topic identifier, and press **PF6**.
- To view a topic with a search match, move the cursor to the topic identifier, and press Enter. You go to the first occurrence of the search word or phrase in the selected topic.
- To bring up your search list again, select LIST ALL TOPICS WITH MATCHES from the SEARCH pull-down.

Other options from the SEARCH pull-down include GO TO NEXT MATCH, GO TO NEXT BEST TOPIC, and EMPHASIZE MATCHES (to highlight matching text in the book).

Exiting DB2 Online Help

To exit DB2 Online Help, press the Exit PF key (usually **PF3**). The selection panel for the DB2 task panel that you were working on is displayed. To exit DB2 Online Help and return to the task panel, press the Exit key again.

Chapter 5. msys for Setup DB2 Customization Center

The msys for Setup DB2 Customization Center reduces the complexity of configuring a DB2 subsystem when installing, migrating to compatibility mode, enabling new-function mode, and enabling data sharing.

The following topics provide additional information:
- "Introduction to msys for Setup"
- "Using the DB2 Customization Center"

Introduction to msys for Setup

Managed System Infrastructure for Setup (msys for Setup) is a z/OS tool that addresses the difficulties in maintaining z/OS. It establishes a central repository for product configuration data. It also provides a single interface to the repository. It automates all processes that do not require decisions by the system administrator and defines defaults to minimize the situations in which decisions are necessary. msys for Setup is a base element of z/OS.

The msys for Setup framework consists of three components. These components are:
- The msys for Setup workplace. It runs on a Windows workstation and provides a graphical user interface similar to the Windows Explorer. The msys for Setup workplace is used to manage z/OS products. You interact with the workplace. It can be downloaded to the workstation from any z/OS host.
- The msys for Setup host program. This is based on the Lightweight Directory Access Protocol (LDAP) directory support. The host program usually resides on the z/OS system. It manages all installation and customization tasks.
- The msys for Setup management directory. This stores the configuration data for all systems on which msys for Setup is enabled. It resides on the z/OS system.

z/OS products that provide an msys for Setup plug-in can be managed using msys for Setup. Parameter values for a particular product that is enabled for use with msys for Setup can be specified using the graphical user interface of the product plug-in. These values are stored in the msys for Setup management directory and eventually used by the host code of the product plug-in to customize and install the product.

For information about using msys for Setup with DB2 UDB for z/OS and other products in your z/OS environment, see *z/OS Managed System Infrastructure for Setup User's Guide*.

Using the DB2 Customization Center

After you have prepared your z/OS subsystem and workstation for use with msys for Setup, you can add the DB2 Customization Center to the msys for Setup workplace. After adding the DB2 Customization Center to the workplace, you are ready to use the DB2 Customization Center.

During refresh, the DB2 Customization Center retrieves current DB2 and z/OS settings. These values are stored in the msys for Setup management directory. Then, you provide information to the DB2 Customization Center that is used to set up your DB2 subsystem.

During customization, you review and, if necessary, modify the values of DB2 system parameters. Then, during update, the DB2 Customization Center applies the changes that you made to the DB2 subsystem.

Adding the DB2 Customization Center to msys for Setup

Before you can use the DB2 Customization Center to install DB2, you must add the DB2 Customization Center to msys for Setup. This tells msys for Setup that the DB2 libraries have been loaded on the z/OS system via SMP/E. For information on loading DB2 libraries, see Chapter 3, "Loading DB2 libraries."

DB2 UDB for z/OS includes an XML document that provides msys for Setup with necessary information. This document resides in *prefix*.SDSNXML. You can use the ″Add a product set″ wizard from the msys for Setup workplace to add the DB2 product set to the workplace. When using this wizard, specify that you already have an up-to-date XML document for DB2. Enter the data set and member name of the XML document in the following format:

```
prefix.SDSNXML(DSNMXML)
```

For more information about adding a product set to the msys for Setup workplace, see *z/OS Managed System Infrastructure for Setup User's Guide*.

Refreshing the msys for Setup workplace

After you have added the DB2 Customization Center to the msys for Setup framework, you must perform a refresh. During this step, the msys for Setup workplace retrieves any configuration information about your DB2 subsystem that exists on the z/OS host. The configuration information is stored in the msys for Setup management directory. If you are migrating DB2, customizing an existing DB2 subsystem, or enabling DB2 for data sharing, the refresh step obtains current parameter information from your DB2 subsystem on the z/OS host.

You will enter information such as the DB2 subsystem name, command prefix, and target library prefix. If you have previously used the CLIST to customize a DB2 subsystem, you can clone this DB2 by specifying the name of the output member generated by the CLIST. After you have provided this information, you will be able to perform the refresh.

For more information on initiating the Refresh step from the msys for Setup workplace, see *z/OS Managed System Infrastructure for Setup User's Guide*.

Customizing DB2

After you have refreshed the msys for Setup workplace, you can customize DB2. In this step, the DB2 Customization Center contains several wizards that ask you a series of questions. These questions are used to set values for various DB2 parameters. At the end of each wizard, you will be shown a list of DB2 parameters and their values. You can browse this list and modify any parameter value if necessary.

After you have completed the wizards to customize DB2, an ″Update tasks″ window appears. This window shows a list of all the tasks that need to be performed on the z/OS host before DB2 can be used with these new values. These update tasks will vary depending on whether you have customized DB2 for installation, migration, or data sharing. Some of the update tasks can be performed by msys for Setup, but some will need to be performed by you or an authorized user outside of the msys for Setup framework. A few tasks may be performed by either msys for Setup or an authorized user.

If you have already used the DB2 Customization Center to customize a DB2 subsystem, you can copy the parameter values to another DB2 subsystem that is using msys for Setup. See *z/OS Managed System Infrastructure for Setup User's Guide* for more details.

Important: You can choose to complete these tasks over time, but you cannot use DB2 until all of these tasks have been completed.

For more information on initiating the Customize step from the msys for Setup workplace, see *z/OS Managed System Infrastructure for Setup User's Guide*.

Updating DB2

After you have chosen the parameter values you want to customize, you must update the DB2 subsystem with the values that you have chosen. The DB2 Customization Center performs only those tasks that you specified in the "Update tasks" window in the previous step.

Unlike the installation CLIST, the DB2 Customization Center does not generate JCL jobs. The tasks executed during the update are equivalent to those that are performed by the DB2 installation JCL jobs. If you want to use JCL jobs to configure DB2 on the host, you can use the DB2 Customization Center to generate an output member that can be used as input to the CLIST.

Further information about performing the update step is available in *z/OS Managed System Infrastructure for Setup User's Guide*.

Chapter 6. Installing, migrating, and updating system parameters

The values of system parameters describe the operating characteristics of your DB2 system. You can think of changing those values when **installing** DB2 or when **migrating** from Version 7 to DB2 UDB for z/OS Version 8 compatibility mode. These values can be subsequently **updated** to improve your operations.

When installing or migrating, you can run the installation CLIST to prepare jobs that are needed for later steps. Alternatively, you can use msys for Setup DB2 Customization Center, a GUI tool, to install or migrate your DB2 subsystem.

The installation CLIST displays a series of ISPF panels that prompt you to supply the parameter values or accept the supplied defaults. The CLIST verifies that the values that you enter are within the allowable ranges. The “Running the installation CLIST” on page 80 topic contains instructions for running the CLIST. The instructions identify the parameters and describe their purposes in the order in which they appear on the panels.

The following topics provide additional information:

- "Running the installation CLIST"
- "Data parameters panel: DSNTIPA2" on page 101
- "Define group or member panel: DSNTIPK" on page 106
- "System resource data set names panel: DSNTIPH" on page 109
- "Data set names panel 1: DSNTIPT" on page 113
- "Data set names panel 2: DSNTIPU" on page 118
- "Data set names panel 3: DSNTIPW" on page 127
- "Sizes panel 1: DSNTIPD" on page 131
- "Sizes panel 2: DSNTIP7" on page 137
- "Thread management panel: DSNTIPE" on page 140
- "Buffer pool sizes panel 1: DSNTIP1" on page 146
- "Buffer pool sizes panel 2: DSNTIP2" on page 148
- "Tracing parameters panel: DSNTIPN" on page 149
- "Operator functions panel: DSNTIPO" on page 155
- "Application programming defaults panel 1: DSNTIPF" on page 163
- "Application programming defaults panel 2: DSNTIP4" on page 171
- "Performance and optimization panel: DSNTIP8" on page 176
- "IRLM panel 1: DSNTIPI" on page 182
- "IRLM panel 2: DSNTIPJ" on page 188
- "Protection panel: DSNTIPP" on page 194
- "MVS PARMLIB updates panel: DSNTIPM" on page 199
- "Active log data set parameters: DSNTIPL" on page 204
- "Archive log data set parameters panel: DSNTIPA" on page 210
- "Databases and spaces to start automatically panel: DSNTIPS" on page 217
- "Distributed data facility panel 1: DSNTIPR" on page 219
- "Distributed data facility panel 2: DSNTIP5" on page 225
- "Routine parameters panel: DSNTIPX" on page 229
- "Data definition control support panel: DSNTIPZ" on page 232
- "Job editing panel: DSNTIPY" on page 235
- "CLIST calculations panel 1: DSNTIPC" on page 238
- "CLIST calculations panel 2: DSNTIPC1" on page 243
- "Completing the CLIST processing" on page 243
- "The update process" on page 247

Running the installation CLIST

To use the ISPF panels, you must first make the DB2 ISPF library available to TSO and then invoke the installation CLIST in ISPF mode. You must be aware of the output that the panel session produces and take steps to save it for use later. Instructions for this procedure follow.

To use DB2 Online Help, you must:

1. Set up DB2 Online Help according to the instructions in Chapter 4, "Setting up and using DB2 Online Help," on page 67.
2. Verify online help by entering BOOKMGR from the TSO command option.
3. In your SYSPROC concatenation, ensure that the data set that contains the correct version of the DSNTINST CLIST is concatenated ahead of any other versions of DSNTINST.
4. Invoke the DSNTINS0 CLIST.

If you do not want to use DB2 Online Help:

- Invoke the DSNTINST CLIST.

The installation CLIST allocates several data sets for input/output. From your TSO user ID, you should be able to allocate these data sets to the permanent or temporary unit names that are provided on installation panel DSNTIPA2. These devices can be defined by an esoteric device group. For more information on esoteric device groups, see *z/OS MVS Initialization and Tuning Guide*.

Making the DB2 ISPF libraries available to TSO

Concatenate the DB2 ISPF libraries to your normal allocations by issuing the following commands:

```
PROFILE WTP MSGID
ALLOCATE DDNAME(ISPMLIB) DSN('prefix.SDSNSPFM' +
    'ISP.SISPMENU') SHR REUSE
ALLOCATE DDNAME(ISPPLIB) DSN('prefix.SDSNSPFP' +
    'ISP.SISPPENU') SHR REUSE
ALLOCATE DDNAME(ISPSLIB) DSN('prefix.SDSNSPFS' +
    'ISP.SISPSLIB' 'ISP.SISPSENU') SHR REUSE
```

The PROFILE command provides complete error messages. DB2 does not support using LIBDEFs for the installation CLIST DSNTINS0 and online help.

The ALLOCATE command uses the default names of the libraries that contain the ISPF panels. These ISPF library names might be different at your site. To concatenate or merge existing libraries with them, put the library names in the list of names in parentheses after DSN with the largest block size first. (If two or more libraries have the same block size, you can list either one first.)

Invoking the CLIST

1. Check your TSO logon region size. Usually 2 MB is enough.
2. Invoke ISPF.
3. Select option 6 on the main ISPF panel.
4. Decide and indicate whether you want to use DB2 Online Help.
   - If you want to use DB2 Online Help, enter one of the following specifications:

     ```
     EXEC 'prefix.SDSNCLST(DSNTINS0)'
     EXEC 'prefix.SDSNCLST(DSNTINS0)' 'CONTROL(LIST)'
     ```

     Use the second specification above if you want to receive the messages that trace the progress of the CLIST.
   - If you do not want to use DB2 Online Help, enter one of the following specifications:

     ```
     EXEC 'prefix.SDSNCLST(DSNTINST)'
     EXEC 'prefix.SDSNCLST(DSNTINST)' 'CONTROL(LIST)'
     ```

     Use the second specification above if you want to receive the messages that trace the progress of the CLIST.
5. By default, the CLIST verifies the following information when installation panels DSNTIPT and DSNTIPA1 are displayed:
   - Verifies that the TEMP CLIST LIBRARY and SAMPLE LIBRARY that you specify on DSNTIPT can be allocated and opened for output.
   - If you specify a value for OUTPUT MEMBER NAME on DSNTIPA1, verifies that the data set *prefix*.SDSNAMP can be allocated and opened for output.

   If you do not want the CLIST to verify this information, specify CHECKOUTDS(NO).

   For example, to use DB2 Online Help but skip the verification that the output data sets can be opened, enter:

   ```
   EXEC 'prefix.SDSNCLST(DSNTINS0)' 'CHECKOUTDS(NO)'
   ```

General instructions

The CLIST reads a set of default values and displays them on the panels. The values can be either the original default values that are supplied by IBM or a set of values that you create in a previous CLIST run.

The installation CLIST saves the panel input into your DSNTID*xx* output member just before the CLIST issues this message:

```
DSNT4781 BEGINNING EDITED DATA SET OUTPUT.
```

Output from the panel session

As output, the panel session produces:

- A new data set member, if specified, that contains the resulting parameter values from the session. This member is stored in *prefix*.SDSNSAMP.
- A new data set, *prefix*.NEW.SDSNSAMP, that contains the edited JCL with the values that you entered on the panels.
- A new data set, *prefix*.NEW.SDSNTEMP, that contains tailored CLISTs for input to job DSNTIJVC, which is run during installation or migration.

Figure 2 on page 83 illustrates by examples how the CLIST works during installation. Figure 3 on page 83 illustrates how the CLIST works during migration to compatibility mode. Figure 4 on page 84 illustrates how the CLIST works during conversion to new-function mode.

Use job DSNTIJVC to combine the CLISTs into a common data set. If you are installing DB2, see “Make DB2 CLISTs available to TSO and batch users: DSNTIJVC” on page 264 for more information. If you are migrating, see “Make DB2 CLISTs available to TSO and batch users: DSNTIJVC” on page 326.

DB2 performs validity checking of the values that you enter during the panel sessions. If you receive an ISPF error message, press the HELP key for additional information.

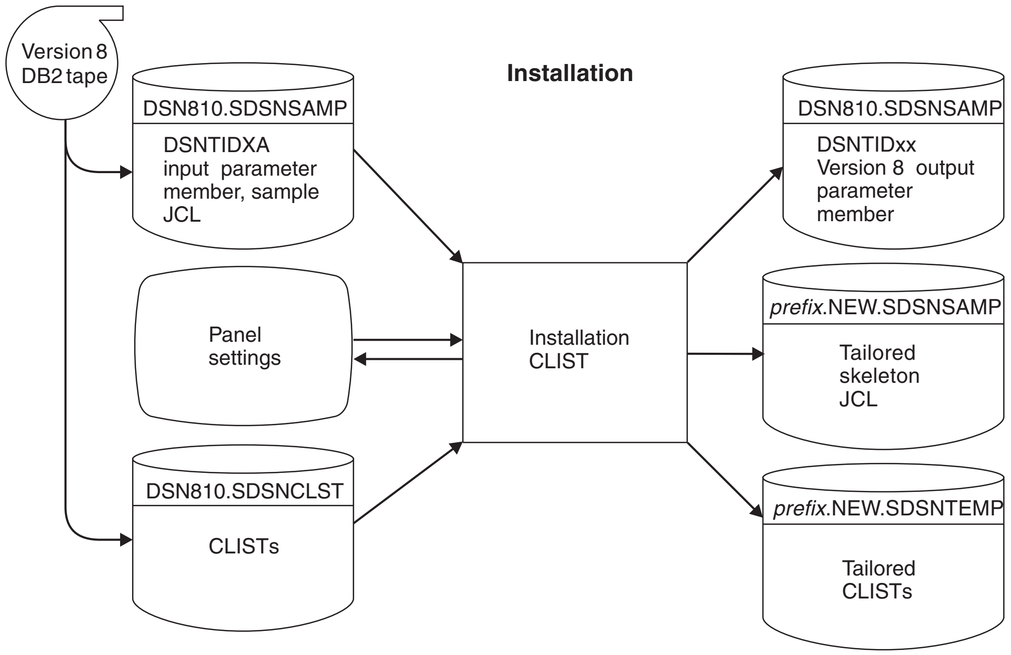


Figure 2. Examples of input to and output from the installation CLIST during installation

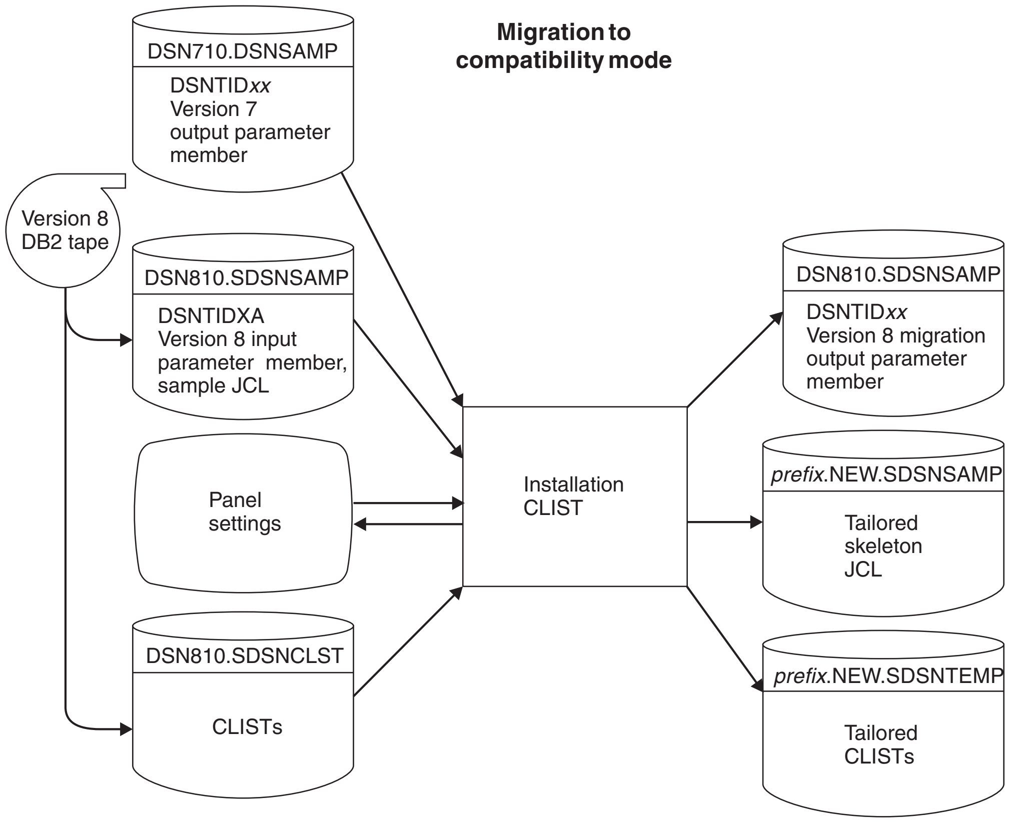


Figure 3. Examples of input to and output from the installation CLIST during migration to compatibility mode

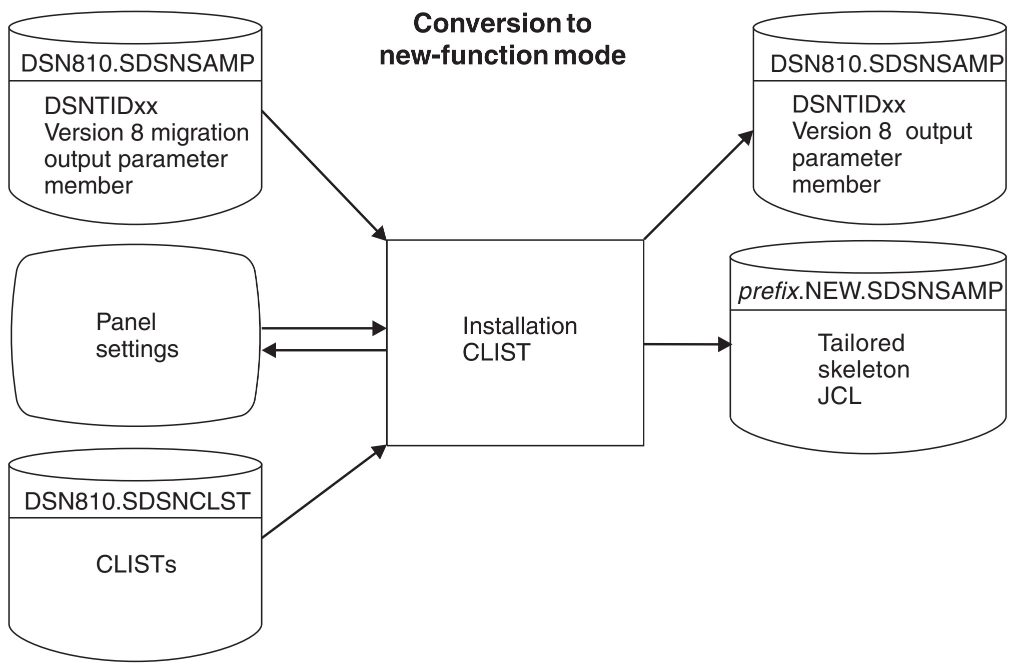

Conversion to new-function mode
DSN810.SDSNSAMP
DSNTIDxx Version 8 migration output parameter member
Panel settings
DSN810.SDSNCLST
CLISTs
Installation CLIST
DSN810.SDSNSAMP
DSNTIDxx Version 8 output parameter member
prefix.NEW.SDSNSAMP
Tailored skeleton JCL

Figure 4. Examples of input to and output from the installation CLIST during conversion to new-function mode

Actions that are allowed on panels

All panel sequences begin with the Main Panel (DSNTIPA1), or with the Online Book Data Set Names panel (DSNTIPA0) if you are using online help.

Preparation: After the description of each parameter, record your choice for a value before you actually use the panels. If, for some reason, you exit the CLIST before you go through all the panels, your values are not saved.

Panels that have fields marked with asterisks show their values are primed on the basis of values from a previous panel. The following message is found on these panels:

```
DSNT444I SCROLLING BACKWARD MAY CHANGE FIELDS MARKED WITH ASTERISKS
```

If you scroll back to the panel which has the original value, the values on the succeeding panels are refreshed **only** if the original value is changed. If the values are changed, the following message is displayed:

```
DSNT443I VALUES MARKED WITH AN ASTERISK HAVE BEEN UPDATED
```

For example, panel DSNTIPH has fields that are marked with asterisks indicating values that are primed on the basis of the CATALOG ALIAS value (field 1) on installation panel DSNTIPA2.

Help: If you have DB2 Online Help installed, press the **HELP** PF key (usually **PF1**) or enter HELP on the command line to get help for the choices you can make on each panel. Pressing the **HELP** key takes you to a help menu panel. Type the number of the item for which you need help and press Enter to link directly to detailed information on that subject in an online book. You can also search or scroll through the book to find additional information on related topics as well. Press the **EXIT** PF key (usually **PF3**) to exit the book and return to the help menu panel.

Press the **END** PF key (usually **PF3**) to return to the installation panels. You cannot make actual entries on help menu panels. Return to the installation panel to make the entry.

Data entry: Enter your choice on a panel in the space that is marked by an arrow (===>). Begin your entry in the second position to the right of the arrow. (The first position is protected; you cannot write in it.)

Panel IDs: If you want the panel IDs to appear on each panel, enter the following command from any panel:

```
PANELID ON
```

Reading the panel descriptions

Scrolling installation panels: The installation panels enable you to scroll back to previous panels to review or change values. The **END** key (usually **PF3** takes you back to the previous panel. Pressing Enter continues to validate entries in the current panel and displays the next panel. If you want to exit completely from the installation process, use the **RETURN** key (usually **PF4**).

Defaults: The defaults shown in this book are the original defaults that are supplied by IBM. If you ran the CLIST before and saved the updated panel values in a DSNTID*xx* data set that you are now using as input, the values you entered now appear as defaults on the panels. Panel values that are modified outside of the installation process are not saved in the DSNTID*xx* data set and are not reflected on the panels.

DSNHDECP names: These are the names of the parameters in the data-only load module DSNHDECP.

Subsystem parameter names: These are the names of the parameters in the data-only load module DSNZP*xxx*.

Acceptable values: This part of the description gives you the range of allowable values or the list of allowable choices for an installation panel field**—not necessarily the value that is associated with that field**. If the maximum allowable value is over 1024, in most cases you can use the equivalent K value. (The CLIST automatically multiplies the K value by 1024.) If the maximum allowable value is over 1 048 576, in most cases you can use the equivalent M value. (The CLIST automatically multiplies the M value by 1 048 576.) If the maximum allowable value is over 1024 MB, in most cases you can use the equivalent G value. If the maximum allowable value is over 1024 GB, in most cases you can use the equivalent T value. The maximum acceptable values might be too large for smaller systems; therefore, ensure that the values that you enter are valid for the size of your system.

Update: This information identifies a corresponding field on the update panel or refers to a panel that gives update instructions. When an option is specified, it refers to the option on the Update Selection Panel (DSNTIPB).

Your installation options are described in the sequence that DB2 presents the panels to you:

Directory of panels

Your installation options are described in the sequence that DB2 presents the panels to you:

Table 29. Panel identifiers

Panel ID	Panel title	See
DSNTIPA0	Online Book Data Set Names	93
DSNTIPA1	Main Panel	94
DSNTIPA2	Data Parameters	101
DSNTIPK	Define Group or Member	106
DSNTIPH	System Resource Data Set Names	109
DSNTIPT	Data Set Names Panel 1	113
DSNTIPU	Data Set Names Panel 2	118
DSNTIPW	Data Set Names Panel 3	127
DSNTIPD	Sizes Panel 1	131
DSNTIP7	Sizes Panel 2	137
DSNTIPE	Thread Management	140
DSNTIP1	Buffer Pool Sizes Panel 1	146
DSNTIP2	Buffer Pool Sizes Panel 2	148
DSNTIPN	Tracing and Checkpoint Parameters	149
DSNTIPO	Operator Functions	155
DSNTIPF	Application Programming Defaults Panel 1	163
DSNTIP4	Application Programming Defaults Panel 2	171
DSNTIP8	Performance and Optimization	176
DSNTIPI	IRLM Panel 1	182
DSNTIPJ	IRLM Panel 2	188
DSNTIPP	Protection	194
DSNTIPM	MVS PARMLIB Updates	199
DSNTIPL	Active Log Data Set Parameters	204
DSNTIPA	Archive Log Data Set Parameters	210
DSNTIPS	Databases and Spaces to Start Automatically	217
DSNTIPR	Distributed Data Facility Panel 1	219
DSNTIP5	Distributed Data Facility Panel 2	225
DSNTIPX	Routine Parameters	229
DSNTIPZ	Data Definition Control Support	232
DSNTIPY	Job Editing	235
DSNTIPC	CLIST Calculations Panel 1	238
DSNTIPC1	CLIST Calculations Panel 2	243
DSNTIPB	Update Selection Menu	248

Directory of panel field names

The following fields are presented on the installation panels:

Table 30. Panel fields

#	Panel field name	Panel	See
#	AGGREGATION FIELDS	DSNTIPN	149
#	ALLOCATION UNITS	DSNTIPA	210
#	APPL PROG AND SQL GUIDE	DSNTIPA0	93
#	APPL REGISTRATION TABLE	DSNTIPZ	232
#	APPLICATION DBRM	DSNTIPT	113
#	APPLICATION ENCODING	DSNTIPF	163
#	APPLICATION LOAD	DSNTIPT	113
#	ARCHIVE LOG FREQ	DSNTIPL	204
#	ARCHIVE LOG RACF	DSNTIPP	194
#	ART/ORT ESCAPE CHARACTER	DSNTIPZ	232
#	ASCII CCSID	DSNTIPF	163
#	ASSISTANT	DSNTIPK	106
#	AUDIT TRACE	DSNTIPN	149
#	AUTH AT HOP SITE	DSNTIP5	225
#	AUTH EXIT LIMIT	DSNTIPP	194

Table 30. Panel fields (continued)

Panel field name	Panel	See
AUTH MEMBER	DSNTIPM	199
AUTH SEQUENCE	DSNTIPM	199
AUTO BIND	DSNTIPO	155
AUTO START	DSNTIPI	182
BACKOUT DURATION	DSNTIPL	204
BIND NEW PACKAGE	DSNTIPP	194
BLOCK SIZE	DSNTIPA	210
BP0 - BP25	DSNTIP1	146
BP26 - BP49	DSNTIP2	148
BP8K0-BP8K9	DSNTIP2	148
BP16K0-BP16K9	DSNTIP2	148
BP32K-BP32K9	DSNTIP2	148
BUFFER POOL SIZE	DSNTIPC	238
C/CPP COMPILER LIBRARY	DSNTIPU	118
C/CPP COMPILER MODULE	DSNTIPU	118
C/CPP HEADER LIBRARY	DSNTIPU	118
C/370 COMPILER MESSAGES	DSNTIPU	118
CACHE DYNAMIC SQL	DSNTIP8	176
CATALOG ALIAS	DSNTIPA2	101
CATALOG DATA	DSNTIPA	210
CHECKPOINT FREQ	DSNTIPL	204
CICS COBOL LIBRARY	DSNTIPW	127
CICS EXCI LIBRARY	DSNTIPW	127
CICS LOAD LIBRARY	DSNTIPW	127
CICS MACRO LIBRARY	DSNTIPW	127
CICS PL/I LIBRARY	DSNTIPW	127
CLIST LIBRARY	DSNTIPT	113
COBOL COMPILER LIBRARY	DSNTIPU	118
CODE STORAGE SIZE	DSNTIPC	238
COLUMNS	DSNTIPD	131
COMMAND PREFIX	DSNTIPM	199
COMMAND REFERENCE	DSNTIPA0	93
COMMAND SCOPE	DSNTIPM	199
COMPACT DATA	DSNTIPA	210
CONTRACT THREAD STG	DSNTIPE	140
CONTROL ALL APPLICATIONS	DSNTIPZ	232
COORDINATOR	DSNTIPK	106
COPY 1 NAME	DSNTIPH	109
COPY 2 NAME	DSNTIPH	109
COPY 1 PREFIX	DSNTIPH	109
COPY 2 PREFIX	DSNTIPH	109
CPP AUTO CALL LIBRARY	DSNTIPU	118
CPP CLASS LIBRARY	DSNTIPU	118
CPP PROCEDURE LIBRARY	DSNTIPU	118
CURRENT DEGREE	DSNTIP8	176
CURRENT MAINT TYPES	DSNTIP8	176
CURRENT REFRESH AGE	DSNTIP8	176
DATASET STATS TIME	DSNTIPN	149
DATA SET(MEMBER) NAME	DSNTIPA1	94
DATA SET STORAGE SIZE	DSNTIPC	238
DATA SHARING	DSNTIPA1	94
DATABASE PROTOCOL	DSNTIP5	225
DATABASES	DSNTIPE	140
DATABASES	DSNTIPD	131
DATE FORMAT	DSNTIP4	171
DBADM CREATE AUTH	DSNTIPP	194

Table 30. Panel fields (continued)

Panel field name	Panel	See
DBRM LIBRARY	DSNTIPT	113
DB2 GENERIC LUNAME	DSNTIPR	219
DB2 LOCATION NAME	DSNTIPR	219
DB2 NETWORK LUNAME	DSNTIPR	219
DB2 NETWORK PASSWORD	DSNTIPR	219
DB2 PROC NAME	DSNTIPX	229
DDF STARTUP OPTION	DSNTIPR	219
DDF THREADS	DSNTIPR	219
DDF/RRSAF ACCUM	DSNTIPN	149
DEADLOCK CYCLE	DSNTIPJ	188
DEADLOCK TIME	DSNTIPJ	188
DEALLOC PERIOD	DSNTIPA	210
DECIMAL ARITHMETIC	DSNTIP4	171
DECIMAL POINT IS	DSNTIPF	163
DECLARATION LIBRARY	DSNTIPT	113
DEF ENCODING SCHEME	DSNTIPF	163
DEFAULT BUFFER POOL FOR USER DATA	DSNTIP1	146
DEFAULT BUFFER POOL FOR USER INDEXES	DSNTIP1	146
DEFINE CATALOG	DSNTIPA2	101
DESCRIBE FOR STATIC	DSNTIP4	171
DEVICE TYPE 1	DSNTIPA	210
DEVICE TYPE 2	DSNTIPA	210
DIST SQL STR DELIMTR	DSNTIPF	163
DISCONNECT IRLM	DSNTIPJ	188
DL/I BATCH TIMEOUT	DSNTIPI	182
DPROP SUPPORT	DSNTIPO	155
DRDA PORT	DSNTIP5	225
DSMAX	DSNTIPC	238
EBCDIC CCSID	DSNTIPF	163
EDM DBD CACHE	DSNTIPC	238
EDM STATEMENT CACHE	DSNTIPC	238
EDMPOOL STORAGE SIZE	DSNTIPC	238
EVALUATE UNCOMMITTED	DSNTIP8	176
EXECUTED STMTS	DSNTIPD	131
EXIT LIBRARY	DSNTIPT	113
EXPLAIN PROCESSING	DSNTIPO	155
EXTENDED SECURITY	DSNTIPR	219
EXTRA BLOCKS REQ	DSNTIP5	225
EXTRA BLOCKS SRV	DSNTIP5	225
FORTRAN COMPILER LIBRARY	DSNTIPU	118
FORTRAN LINK EDIT LIB	DSNTIPU	118
FREQUENCY TYPE	DSNTIPL	204
FROM RELEASE	DSNTIPA1	94
GDDM LOAD MODULES	DSNTIPW	127
GDDM MACLIB	DSNTIPW	127
GROUP ATTACH	DSNTIPK	106
GROUP NAME	DSNTIPK	106
HIGH LEVEL ASSEMBLER LIB	DSNTIPU	118
IBM LE RUNTIME LIBRARY	DSNTIPU	118
IBM LE LINK EDIT LIB	DSNTIPU	118
IBM LE PRELINK MSG LIB	DSNTIPU	118
IDLE THREAD TIMEOUT	DSNTIPR	219
IMMEDIATE WRITE	DSNTIP8	176
IMS BMP TIMEOUT	DSNTIPI	182
IMS RESLIB	DSNTIPW	127
INCLUDE LIBRARY	DSNTIPT	113

Table 30. Panel fields (continued)

Panel field name	Panel	See
INDEX SPACE DEFAULT SIZE	DSNTIP7	137
INPUT MEMBER NAME	DSNTIPA1	94
INSTALL DD CONTROL SUPT.	DSNTIPZ	232
INSTALL IRLM	DSNTIPI	182
INSTALL TYPE	DSNTIPA1	94
INSTALLATION GUIDE	DSNTIPA0	93
IRLM LOAD LIBRARY	DSNTIPT	113
IRLM LOCK MAXIMUM SPACE	DSNTIPC	238
IRLM XCF GROUP NAME	DSNTIPJ	188
ISPF ISPLINK MODULE	DSNTIPW	127
IVP DATA LIBRARY	DSNTIPT	113
LANGUAGE DEFAULT	DSNTIPF	163
LARGE EDM BETTER FIT	DSNTIP8	176
LEVELID UPDATE FREQ	DSNTIPL	204
LIMIT BACKOUT	DSNTIPL	204
LINK LIST ENTRY	DSNTIPM	199
LINK LIST LIBRARY	DSNTIPT	113
LINK LIST SEQUENCE	DSNTIPM	199
LOAD DISTRIBUTION	DSNTIPT	113
LOAD LIBRARY	DSNTIPT	113
LOCAL DATE LENGTH	DSNTIP4	171
LOCALE LC_CTYPE	DSNTIPF	163
LOCAL TIME LENGTH	DSNTIP4	171
LOCK ENTRY SIZE	DSNTIPJ	188
LOCKS PER TABLE(SPACE)	DSNTIPJ	188
LOCKS PER USER	DSNTIPJ	188
LOG APPLY STORAGE	DSNTIPL	204
LONG-RUNNING READER	DSNTIPE	140
MACRO LIBRARY	DSNTIPT	113
MANAGE THREAD STORAGE	DSNTIPE	140
MAX ABEND COUNT	DSNTIPX	229
MAX BATCH CONNECT	DSNTIPE	140
MAX DEGREE	DSNTIP8	176
MAX KEPT DYN STMTS	DSNTIPE	140
MAX OPEN CURSOR	DSNTIPX	229
MAX REMOTE ACTIVE	DSNTIPE	140
MAX REMOTE CONNECTED	DSNTIPE	140
MAX STORAGE FOR LOCKS	DSNTIPJ	188
MAX STORED PROCS	DSNTIPX	229
MAX TSO CONNECT	DSNTIPE	140
MAX INACTIVE DBATS	DSNTIPR	219
MAX USERS	DSNTIPE	140
MAXIMUM LE TOKENS	DSNTIP7	137
MEMBER IDENTIFIER	DSNTIPJ	188
MEMBER NAME	DSNTIPK	106
MINIMUM DIVIDE SCALE	DSNTIPF	163
MIXED DATA	DSNTIPF	163
MONITOR SIZE	DSNTIPN	149
MONITOR TRACE	DSNTIPN	149
NUMBER OF COPIES	DSNTIPH	109
NUMBER OF LOCK ENTRIES	DSNTIPJ	188
NUMBER OF LOGS	DSNTIPL	204
NUMBER OF TCBS	DSNTIPX	229
OBJT REGISTRATION TABLE	DSNTIPZ	232
OPTIMIZATION HINTS	DSNTIP8	176
OPTIMIZE EXTENT SIZING	DSNTIP7	137

Table 30. Panel fields (continued)

Panel field name	Panel	See
OUTPUT BUFFER	DSNTIPL	204
OUTPUT MEMBER NAME	DSNTIPA1	94
PACKAGE AUTH CACHE	DSNTIPP	194
PACKAGE LISTS	DSNTIPD	131
PACKAGE STATEMENTS	DSNTIPD	131
PACKAGES	DSNTIPD	131
PAD INDEXES BY DEFAULT	DSNTIPE	140
PAD NUL-TERMINATED	DSNTIP4	171
PROTECT	DSNTIPJ	188
PARAMETER MODULE	DSNTIPO	155
PERMANENT UNIT NAME	DSNTIPA2	101
PLAN AUTH CACHE	DSNTIPP	194
PLAN STATEMENTS	DSNTIPD	131
PLANS	DSNTIPD	131
PL/I COMPILER LIBRARY	DSNTIPU	
PL/I COMPILER MODULE	DSNTIPU	
POOL THREAD TIMEOUT	DSNTIP5	225
PREFIX	DSNTIPA1	94
PRIMARY QUANTITY	DSNTIPA	210
PROC NAME	DSNTIPI	182
QUIESCE PERIOD	DSNTIPA	210
READ COPY2 ARCHIVE	DSNTIPO	155
READ TAPE UNITS	DSNTIPA	210
REAL TIME STATS	DSNTIPO	155
RECALL DATABASE	DSNTIPO	155
RECALL DELAY	DSNTIPO	155
RECORDING MAX	DSNTIPA	210
REGISTRATION DATABASE	DSNTIPZ	232
REGISTRATION OWNER	DSNTIPZ	232
RELEASE LOCKS	DSNTIP8	176
REMOTE LOCATION	DSNTIPY	235
REQUIRE FULL NAMES	DSNTIPZ	232
RESOURCE AUTHID	DSNTIPP	194
RESOURCE TIMEOUT	DSNTIPI	182
RESYNC INTERVAL	DSNTIPR	219
RESYNC PORT	DSNTIP5	225
RETAINED LOCK TIMEOUT	DSNTIPI	182
RETENTION PERIOD	DSNTIPA	210
RID POOL SIZE	DSNTIPC	238
RLF AUTO START	DSNTIPO	155
RLST ACCESS ERROR	DSNTIPO	155
RLST ACCESS ERROR	DSNTIPR	219
RLST NAME SUFFIX	DSNTIPO	155
RO SWITCH CHKPTS	DSNTIPL	204
RO SWITCH TIME	DSNTIPL	204
ROUTINE AUTH CACHE	DSNTIPP	194
SAMPLE LIBRARY	DSNTIPT	113
SECONDARY QTY	DSNTIPA	210
SEQUENTIAL CACHE	DSNTIPE	140
SINGLE VOLUME	DSNTIPA	210
SITE TYPE	DSNTIPO	155
SKIP UNCOMM INSERTS	DSNTIP8	176
SMF ACCOUNTING	DSNTIPN	149
SMF STATISTICS	DSNTIPN	149
SORT LIBRARY	DSNTIPW	127
SORT POOL SIZE	DSNTIPC	238

#	*Table 30. Panel fields (continued)*

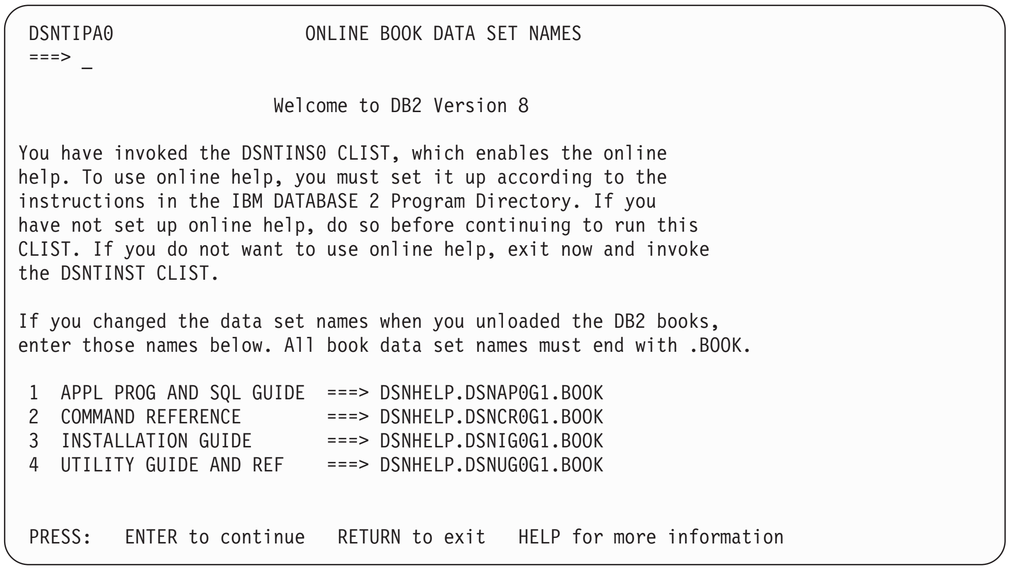
#	Panel field name	Panel	See
#	SQL STRING DELIMITER	DSNTIPF	163
#	STAR JOIN MAX POOL	DSNTIP8	176
#	STAR JOIN QUERIES	DSNTIP8	176
#	START IRLM CTRACE	DSNTIPI	182
#	STATISTICS HISTORY	DSNTIPO	155
#	STATISTICS ROLLUP	DSNTIPO	155
#	STATISTICS SYNC	DSNTIPN	149
#	STATISTICS TIME	DSNTIPN	149
#	STD SQL LANGUAGE	DSNTIP4	171
#	STRING DELIMITER	DSNTIPF	163
#	SUBSYSTEM MEMBER	DSNTIPM	199
#	SUBSYSTEM NAME	DSNTIPI	182
#	SUBSYSTEM NAME	DSNTIPM	199
#	SUBSYSTEM SEQUENCE	DSNTIPM	199
#	SUFFIX	DSNTIPA1	94
#	SUPPRESS SOFT ERRORS	DSNTIPM	199
#	SYSTEM ADMIN 1	DSNTIPP	194
#	SYSTEM ADMIN 2	DSNTIPP	194
#	SYSTEM LOB VALUE STORAGE	DSNTIP7	137
#	SYSTEM MACLIB	DSNTIPW	127
#	SYSTEM OPERATOR 1	DSNTIPP	194
#	SYSTEM OPERATOR 2	DSNTIPP	194
#	SYSTEM PROCEDURES	DSNTIPW	127
#	TABLES	DSNTIPD	131
#	TABLES IN STMT	DSNTIPD	131
#	TABLE SPACES	DSNTIPD	131
#	TABLE SPACE DEFAULT SIZE	DSNTIP7	199
#	TCP/IP ALREADY VERIFIED	DSNTIP5	225
#	TCP/IP KEEPALIVE	DSNTIP5	225
#	TEMP CLIST LIBRARY	DSNTIPT	113
#	TEMP 4K DATA SETS	DSNTIPD	131
#	TEMP 4K SPACE	DSNTIPD	131
#	TEMP 32K DATA SETS	DSNTIPD	131
#	TEMP 32K SPACE	DSNTIPD	131
#	TEMPORARY UNIT NAME	DSNTIPA2	101
#	TIME FORMAT	DSNTIP4	171
#	TIME TO AUTOSTART	DSNTIPI	182
#	TIMEOUT VALUE	DSNTIPX	229
#	TIMESTAMP ARCHIVES	DSNTIPH	109
#	TOTAL MAIN STORAGE	DSNTIPC	238
#	TOTAL STORAGE BELOW 16MB	DSNTIPC	238
#	TRACE AUTO START	DSNTIPN	149
#	TRACE SIZE	DSNTIPN	149
#	TRACKER SITE	DSNTIPO	155
#	U LOCK FOR RR/RS	DSNTIPI	182
#	UNICODE CCSID	DSNTIPF	163
#	UNICODE IFCIDS	DSNTIPN	149
#	UNKNOWN AUTHID	DSNTIPP	194
#	UNREGISTERED DDL DEFAULT	DSNTIPZ	232
#	UPDATE PART KEY COLS	DSNTIP8	176
#	UPDATE RATE	DSNTIPL	204
#	UR CHECK FREQ	DSNTIPL	204
#	UR LOG WRITE CHECK	DSNTIPL	204
#	USE FOR DYNAMICRULES	DSNTIP4	171
#	USE PROTECTION	DSNTIPP	194
#	USER LOB VALUE STORAGE	DSNTIP7	137

Table 30. Panel fields (continued)

Panel field name	Panel	See
UTILITY GUIDE AND REFERENCE	DSNTIPA0	93
UTILITY CACHE OPTION	DSNTIPE	140
UTILITY TIMEOUT	DSNTIPI	182
VARCHAR FROM INDEX	DSNTIP8	176
VARY DS CONTROL	DSNTIP7	137
VIEWS	DSNTIPD	131
VOLUME SERIAL 1	DSNTIPA2	101
VOLUME SERIAL 2	DSNTIPA2	101
VOLUME SERIAL 3	DSNTIPA2	101
VOLUME SERIAL 4	DSNTIPA2	101
VOLUME SERIAL 5	DSNTIPA2	101
VOLUME SERIAL 6	DSNTIPA2	101
VOLUME SERIAL 7	DSNTIPA2	101
WLM ENVIRONMENT	DSNTIPX	229
WLM PROC NAME	DSNTIPX	229
WORK FILE DB	DSNTIPK	106
WORKING STORAGE SIZE	DSNTIPC	238
WRITE TO OPER	DSNTIPA	210
WTO ROUTE CODES	DSNTIPO	155
WTOR ROUTE CODE	DSNTIPA	210
X LOCK FOR SEARCHED U/D	DSNTIPI	182

Online book data set names panel: DSNTIPA0

The entries on this panel enable DB2 Online Help.



```
 DSNTIPA0                  ONLINE BOOK DATA SET NAMES
 ===> _

                        Welcome to DB2 Version 8

You have invoked the DSNTINS0 CLIST, which enables the online
help. To use online help, you must set it up according to the
instructions in the IBM DATABASE 2 Program Directory. If you
have not set up online help, do so before continuing to run this
CLIST. If you do not want to use online help, exit now and invoke
the DSNTINST CLIST.

If you changed the data set names when you unloaded the DB2 books,
enter those names below. All book data set names must end with .BOOK.

 1  APPL PROG AND SQL GUIDE  ===> DSNHELP.DSNAP0G1.BOOK
 2  COMMAND REFERENCE        ===> DSNHELP.DSNCR0G1.BOOK
 3  INSTALLATION GUIDE       ===> DSNHELP.DSNIG0G1.BOOK
 4  UTILITY GUIDE AND REF    ===> DSNHELP.DSNUG0G1.BOOK


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

Figure 5. Online book data set names panel: DSNTIPA0

1. **APPL PROG AND SQL GUIDE**

Acceptable values:	Valid data set name, which ends in .BOOK. See 97 for information about valid data set names.
Default:	DSNHELP.DSNAP0G1.BOOK
DSNZPxxx:	none

2. **COMMAND REFERENCE**

Acceptable values:	Valid data set name, which ends in .BOOK. See 97 for information about valid data set names.
Default:	DSNHELP.DSNCR0G1.BOOK
DSNZPxxx:	none

3. **INSTALLATION GUIDE**

Acceptable values:	Valid data set name, which ends in .BOOK. See 97 for information about valid data set names.
Default:	DSNHELP.DSNIG0G1.BOOK
DSNZPxxx:	none

4. **UTILITY GUIDE AND REF**

Acceptable values:	Valid data set name, which ends in .BOOK. See 97 for information about valid data set names.
Default:	DSNHELP.DSNUG0G1.BOOK
DSNZPxxx:	none

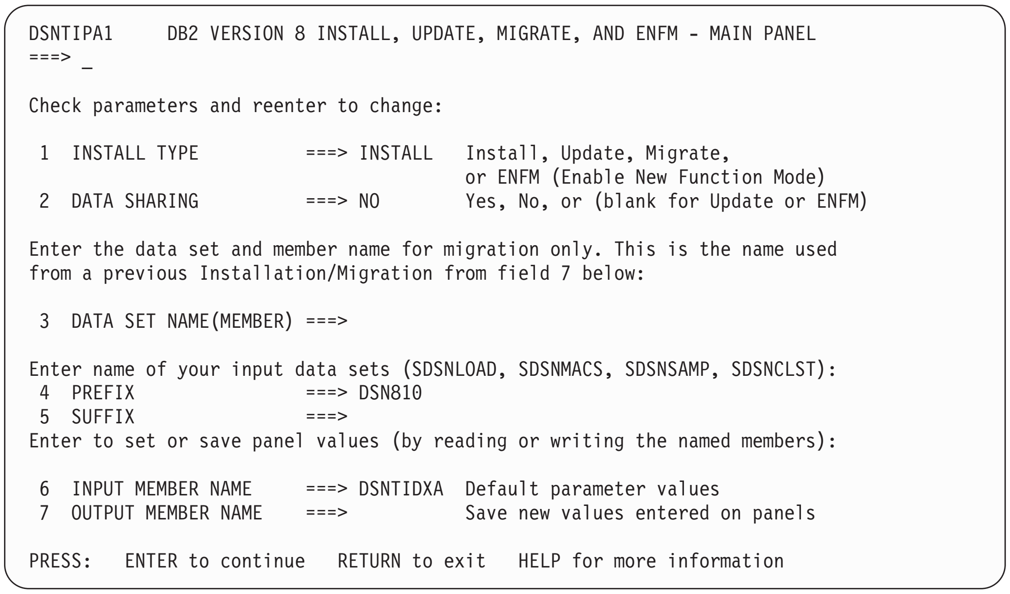
Main panel: DSNTIPA1

The entries on the Main Panel control input to and output from the installation CLIST. When processing is complete, this panel is displayed again. The values you enter are saved in the ISPF profile for your authorization ID and are displayed each time you run the CLIST.

To save your panel input, you must specify an output member name in OUTPUT MEMBER NAME.

The DSNTINST CLIST saves the panel input into your DSNTID*xx* output member just before the CLIST issues this message:

```
DSNT4781 BEGINNING EDITED DATA SET OUTPUT
```



```
DSNTIPA1     DB2 VERSION 8 INSTALL, UPDATE, MIGRATE, AND ENFM - MAIN PANEL
===> _

Check parameters and reenter to change:

 1  INSTALL TYPE          ===> INSTALL   Install, Update, Migrate,
                                         or ENFM (Enable New Function Mode)
 2  DATA SHARING          ===> NO        Yes, No, or (blank for Update or ENFM)

Enter the data set and member name for migration only. This is the name used
from a previous Installation/Migration from field 7 below:

 3  DATA SET NAME(MEMBER) ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DSN810
 5  SUFFIX                ===>
Enter to set or save panel values (by reading or writing the named members):

 6  INPUT MEMBER NAME     ===> DSNTIDXA  Default parameter values
 7  OUTPUT MEMBER NAME    ===>           Save new values entered on panels

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

Figure 6. Main panel: DSNTIPA1

1. **INSTALL TYPE**

Acceptable values:	Install, Update, Migrate, ENFM
Default:	INSTALL
DSNHDECP:	NEWFUN

Specify whether you are installing, updating, migrating, or converting to new-function mode.

- Use INSTALL to install DB2 for the first time. This is the default for the first run of the CLIST. After you have completed the installation, DB2 is in new-function mode.
- Use UPDATE to update parameters for an existing DB2 subsystem.
- Use MIGRATE to migrate from Version 7 to DB2 UDB for z/OS Version 8 compatibility mode. When you are migrating, DATA SET NAME(MEMBER) is required.
- Use ENFM to convert the DB2 catalog to new-function mode. You must run the CLIST in MIGRATE mode before you can choose this option. You should be stabilized in Version 8 compatibility mode before using this option to convert to

new-function mode. In a data sharing environment, the enabling-new-function mode process is done once for the data sharing group.

If you are updating or migrating, you use the same set of panels you use for installation. Each panel displays all fields; however, the fields that cannot be changed in update or migrate mode are protected. This way, you can see the values that are related to ones that you want to change.

If you are converting to new-function mode, see Chapter 9, "Enabling new-function mode" for more information.

You can also choose either INSTALL or UPDATE to recheck values you that chose before.

Certain fields cannot be changed during a migration. See Figure 11 on page 109, Figure 15 on page 132, Figure 16 on page 137, and Figure 27 on page 194 for more information. Ensure that those fields are correct in the data set member you provide.

2. **DATA SHARING**

Acceptable values:	YES, NO, or blank for update and ENFM
Default:	NO
DSNZP*xxx*:	DSN6GRP DSHARE

Specify whether you want to use the data sharing function. Choose NO if you are not using data sharing. If you choose YES, you will continue to panel DSNTIPK after completing panel DSNTIPA2.

If you specify YES during installation, this window is displayed:

```
.-------------------------------.
| DSNTIPP1                      |
|                               |
|  DATA SHARING FUNCTION:       |
|                               |
|  Select one.                  |
|          _ 1. Group           |
|            2. Member          |
|            3. Enable          |
|                               |
|  PRESS:  ENTER to continue    |
|          RETURN to exit       |
|                               |
|                               |
|                               |
'-------------------------------'
```

Figure 7. DSNTIPP1

DATA SHARING FUNCTION:

Acceptable values:	Group, Member, Enable
Default:	none
DSNZP*xxx*:	none

Specify a data sharing function. A value is required. After entering a value, you proceed to panels DSNTIPA2 and DSNTIPK. See *DB2 Data Sharing: Planning and Administration* for more information about the Group, Member, and Enable functions.

If you specify YES in the DATA SHARING field during migration, a window is displayed asking if the current member is the first to migrate.

```
.-------------------------------------.
| DSNTIPP2                            |
|                                     |
|  FIRST MEMBER OF GROUP TO MIGRATE?  |
|                                     |
|  Select one.                        |
|          _ 1. Yes                   |
|            2. No                    |
|                                     |
|  PRESS:  ENTER to continue          |
|          RETURN to exit             |
|                                     |
|                                     |
'-------------------------------------'
```

Figure 8. DSNTIPP2

FIRST MEMBER OF GROUP TO MIGRATE?

Acceptable values:	Yes, No
Default:	none
DSNZP*xxx*:	none

Specify Yes if this is the first member of a data sharing group to migrate. A value is required.

After entering a value, you proceed to panels DSNTIPA2 and DSNTIPK.

3. **DATA SET NAME(MEMBER)**

Acceptable values:	1 to 44 alphanumeric characters
Default:	NULL
DSNZP*xxx*:	none

Specify the name of the input data set to use for migrating from DB2 UDB for z/OS Version 7. The named member contains the output parameters that are produced when you last installed, updated, or migrated DB2. Give the fully qualified data set name in the following form:

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

This is an example of an actual data set name:

```
DSN810.SDSNSAMP(DSNTID71)
```

If you no longer have this data set member, or if the one you have is incorrect, use the installation or update process from your previous release to re-create or correct the member. Enter the correct values on the panels, and save them under a new output member name. Discard the JCL that is created by this process; use the newly created member for migration.

If you are installing, converting to new-function mode, or updating, the field must remain blank.

Valid data set name: Valid data set names can be unqualified or qualified:

Unqualified name
One to eight alphanumeric or national characters, a hyphen, or the character X'C0'. The first character must be alphabetic or national. Do not use hyphens in data set names for RACF-protected data sets. For example, ALPHA is an unqualified data set name.

Qualified name
Multiple names joined by periods. Each name is coded like an unqualified name. Therefore, the name must contain a period within every eight characters. For example, ALPHA.PGM is a qualified data set name. The maximum length of a qualified data set name is:
- 44 characters if you use the TSO PROFILE setting NOPREFIX.
- 42 characters if you use the TSO PROFILE setting PREFIX.
- For an output tape data set, 17 characters, including periods. If the name is longer than 17 characters, only the right-most 17 characters are written to the tape header label.

4. **PREFIX**

Acceptable values:	1 to 18 characters
Default:	DSN810
DSNZP*xxx*:	none

Specify the input prefix for the SDSNLOAD, SDSNMACS, SDSNSAMP,
SDSNDBRM, and SDSNCLST libraries. The prefix must be the same as the name
that you specified for the symbolic parameter TARGPRE in SMP/E job
DSNALLOC.

This is also the prefix for several partitioned data sets, which are deleted, if they exist, and are created or re-created during the tailoring session. If you use the default DSNTIDXA in field 7 (INPUT MEMBER NAME), the prefix for fields 1, 2, 3, and 15, and the prefix and suffix for fields 4 through 14 on installation panel DSNTIPT on 113 are set. If all these data sets do not have the same prefix and suffix, you can change them on installation panel DSNTIPT.

5. **SUFFIX**

Acceptable values:	0 to 17 characters
Default:	NULL
DSNZP*xxx*:	none

Specify a suffix to the names listed below. The fully qualified data set name cannot exceed 44 characters. Names that exceed eight characters must be in groups of no more than eight characters, separated by periods.

Use a suffix only if you have added a common suffix to the following libraries
when you created them in job DSNALLOC:

prefix.ADSNLOAD	*prefix*.SDSNLINK	*prefix*.SDSNMACS
prefix.SDSNLOAD	*prefix*.SDSNEXIT	*prefix*.ADSNMACS
prefix.SDSNCLST	*prefix*.SDSNSAMP	*prefix*.SDSNDBRM

prefix.SDXRRESL	*prefix*.SDSNIVPD

If you did not add a common suffix to these libraries, enter their correct data set names on panel DSNTIPT.

To use the default DB2 data set names, specify DSN810 in field 4, and leave field 5 blank.

6. **INPUT MEMBER NAME**

Acceptable values:	1 to 8 characters
Default:	DSNTIDXA
DSNZP*xxx*:	none

Specify the input member name of the data set that contains the default parameter values for installing and migrating, as in *prefix*.SDSNSAMP.*suffix*. To install DB2 for the first time, use the IBM-supplied defaults in member DSNTIDXA. If you process the panels several times within a single run of the CLIST, all the previous values that are entered, except edited output data sets, remain the same.

For migration to compatibility mode, give two member names for input values: one in the INPUT MEMBER NAME field, and one in the DATA SET NAME(MEMBER). The INPUT MEMBER NAME must specify a member that contains the default parameter values for the new release (usually DSNTIDXA) and is applied first to establish the CLIST parameters. However, if you are migrating a second or subsequent data sharing member to Version 8 then the INPUT MEMBER NAME is the OUTPUT MEMBER NAME used when migrating the first member of the data sharing group to Version 8. The DATA SET NAME(MEMBER) field must specify a member containing the DB2 UDB for z/OS Version 7 values at your site. This member is applied last and overrides the CLIST values established by the member specified in INPUT MEMBER NAME.

For conversion to new-function mode, give a member name for input values. The INPUT MEMBER NAME that you specify for conversion should be the same as the OUTPUT MEMBER NAME that you specified during migration to compatibility mode. In a data sharing environment, the INPUT MEMBER NAME that you specify for conversion should be the same as the OUTPUT MEMBER NAME used when migrating the first member of the data sharing group to Version 8.

To install DB2 by using parameters from a previous run as defaults, you must supply the member that contains the output from the previous run. It was the OUTPUT MEMBER NAME during the last run.

Table 31 lists the data set names that are generated with the *prefix* and *suffix* values from PREFIX and SUFFIX only when the INPUT MEMBER NAME is DSNTIDXA.

Table 31. Resulting data set names when using prefix and suffix parameters

Default library name	CLIST edited library name
prefix.DBRMLIB.DATA	*prefix*.DBRMLIB.DATA.*suffix*
prefix.RUNLIB.LOAD	*prefix*.RUNLIB.LOAD.*suffix*
prefix.SRCLIB.DATA	*prefix*.SRCLIB.DATA.*suffix*
prefix.SDSNDBRM	*prefix*.SDSNDBRM.*suffix*
prefix.SDSNLINK	*prefix*.SDSNLINK.*suffix*
prefix.SDSNLOAD	*prefix*.SDSNLOAD.*suffix*
prefix.SDSNMACS	*prefix*.SDSNMACS.*suffix*
prefix.ADSNLOAD	*prefix*.ADSNLOAD.*suffix*
prefix.ADSNMACS	*prefix*.SDSNMACS.*suffix*
prefix.SDSNSAMP	*prefix*.SDSNSAMP.*suffix*
prefix.SDSNCLST	*prefix*.SDSNCLST.*suffix*
prefix.SDSNIVPD	*prefix*.SDSNIVPD.*suffix*
prefix.SDSNC.H	*prefix*.SDSNC.H
prefix.SDXRRESL.	*prefix*.SDXRRESL.*suffix*

7. **OUTPUT MEMBER NAME**

Acceptable values:	1 to 8 characters
Default:	NULL
DSNZP*xxx*:	none

Specify the member name of the output data set in which to save the values that you enter on the panels. If you do not specify a name, the values are lost when you leave the installation CLIST, and you no longer have the values available for future updates. This member is stored in *prefix*.SDSNSAMP (not the one created by the DSNTINST CLIST). To avoid replacing any members of *prefix*.SDSNSAMP that were shipped with the product, specify DSNTID*xx* as the value of OUTPUT MEMBER NAME, where *xx* is any alphanumeric value except XA or VB.

Always give a new value in the OUTPUT MEMBER NAME field for a new panel session. You supply the name from your current session in the INPUT MEMBER NAME field for your next session. You should not use the same member name for output as for input.

Recommendation: Write down the output member name that you entered below for reference during future sessions:

Recommended approach for a new installer

If you are installing for the first time, try the following suggestions:

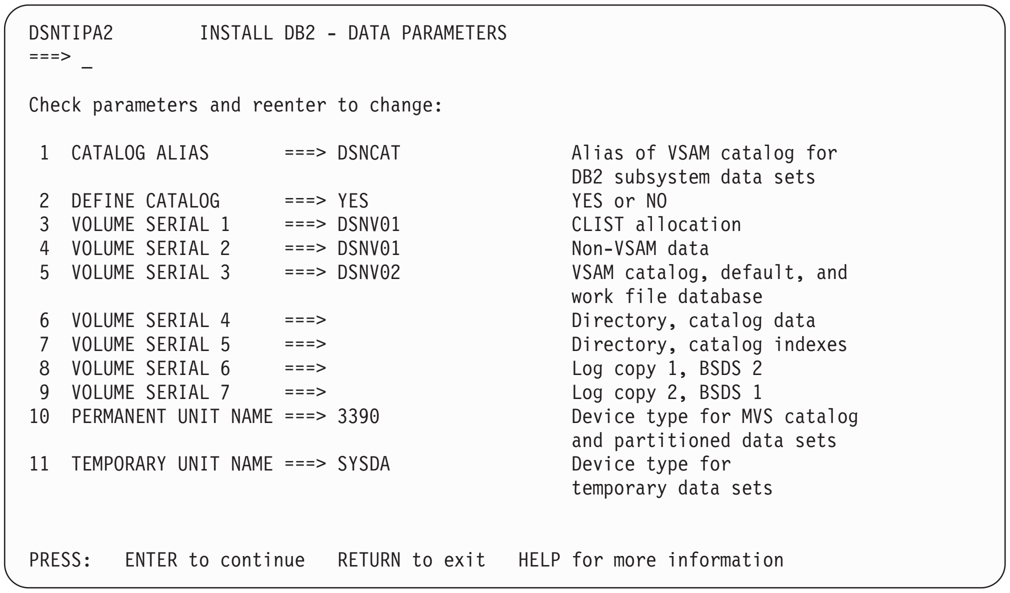
- For field 1, INSTALL TYPE, enter INSTALL.
- Set the PREFIX and SUFFIX fields to the values that you used when you allocated the DB2 libraries by using job DSNALLOC.
- In the INPUT MEMBER NAME field, use DSNTIDXA (the default) for the first run. For any later runs, the CLIST sets the default input name to the prior output name.
- Specify a value in OUTPUT MEMBER NAME to save your options. Specify values in TEMP CLIST LIBRARY, CLIST LIBRARY, and SAMPLE LIBRARY on installation panel DSNTIPT on “Data set names panel 1: DSNTIPT” on page 113 when you want output data sets tailored.

Do not run the installation jobs before tailoring them with the CLIST. If you want to update the parameters later, the CLIST sets the default input name to the prior output name.

Data parameters panel: DSNTIPA2

The entries on this panel define the disk volumes for the subsystem databases and data sets. The values that you enter on this and each of the following panels are saved in the data set member that you named in the OUTPUT MEMBER NAME field on the Main Panel.

For information on updating parameters with changes that cannot be made by using the panels, see "The update process" on page 247.



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

Check parameters and reenter to change:

 1  CATALOG ALIAS       ===> DSNCAT               Alias of VSAM catalog for
                                                  DB2 subsystem data sets
 2  DEFINE CATALOG      ===> YES                  YES or NO
 3  VOLUME SERIAL 1     ===> DSNV01               CLIST allocation
 4  VOLUME SERIAL 2     ===> DSNV01               Non-VSAM data
 5  VOLUME SERIAL 3     ===> DSNV02               VSAM catalog, default, and
                                                  work file database
 6  VOLUME SERIAL 4     ===>                      Directory, catalog data
 7  VOLUME SERIAL 5     ===>                      Directory, catalog indexes
 8  VOLUME SERIAL 6     ===>                      Log copy 1, BSDS 2
 9  VOLUME SERIAL 7     ===>                      Log copy 2, BSDS 1
10  PERMANENT UNIT NAME ===> 3390                 Device type for MVS catalog
                                                  and partitioned data sets
11  TEMPORARY UNIT NAME ===> SYSDA                Device type for
                                                  temporary data sets


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

Figure 9. Data parameters panel: DSNTIPA2

1. **CATALOG ALIAS**

Acceptable values:	1 to 8 characters
Default:	DSNCAT
Update:	see Part 2 (Volume 1) of *DB2 Administration Guide*
DSNZP*xxx*:	DSN6SPRM CATALOG

Specify the alias of the VSAM ICF user catalog or the name of the VSAM ICF master catalog in which to catalog the DB2 VSAM data sets that are created during installation.

VSAM data set cataloging options: The installation jobs catalog the DB2 VSAM data sets. This includes recovery log, subsystem, and user data sets. You must create the catalog that defines these data sets through the VSAM ICF.

Recommendation: Use an ICF user catalog to catalog all DB2 objects because you can use aliases for user catalogs. When you use the CREATE STOGROUP statement, you might need to use an alias for the VCAT option, which must be a single-level one- to eight-character name. You can use a master catalog, but only if the name of the master catalog is a single-level name of one to eight characters.

Ensure that your alias conforms to your local naming conventions. To change this parameter for a previously installed DB2 subsystem, see Part 2 (Volume 1) of *DB2 Administration Guide*. Using the same ICF catalog alias for DB2 UDB for z/OS

Version 8 that you used for Version 7 is recommended because Version 8 uses many of your Version 7 data sets that are already cataloged.

Whether you are installing or migrating, DB2 does not require you to catalog all DB2 VSAM data sets in the same ICF catalog. In this chapter, the catalog that you create when installing is called the *primary* ICF catalog. You must catalog some data sets in the primary catalog. You can catalog other data sets elsewhere, and some data sets need not be cataloged at all. See Table 32 for a list of available options. The BSDS is VSAM KSDS. The archive logs are sequential. All other data sets are VSAM linear data sets.

Table 32. DB2 data sets ICF catalog options

DB2 data sets	Options
DB2 directory (DSNDB01) DB2 catalog (DSNDB06) Default database (DSNDB04) Work file database	You must catalog these data sets in the primary ICF catalog.
Active logs Bootstrap data set	You can catalog these in a different ICF catalog and give them a prefix different from those in the primary catalog.
Archive logs	If the archive log data set is allocated on disk, the data set must be cataloged. If the archive log data set is allocated on a tape device, you have the option to catalog the data set.
User table spaces User index spaces	You do not need to put these in the primary catalog. You can put different user spaces in different ICF catalogs.

You must provide any catalog connections for log and bootstrap data sets that you do not catalog on the primary DB2 ICF catalog.

Recommendation: Add an alias for the proper catalog.

Although you can catalog the two DB2 subsystems on the same ICF catalog, they must not share the same ICF catalog alias, because the alias is the only parameter that makes the data set names unique.

Data set naming conventions: The value that you specify as the z/OS catalog alias is also used as the high-level qualifier for DB2 VSAM data sets. The data sets for the DB2 directory and catalog databases, the default database, and the temporary database are all VSAM linear data sets (LDSs). Their data set names have the following format:

```
dddddddd.DSNDBn.bbbbbbbb.xxxxxxxx.y0001.Accc
```

In this format:

dddddddd	Is the high-level qualifier, the value that you supply for this field.
DSNDB*n*	Is a constant identifying this as a DB2 data set; *n* is C for a cluster name or D for a data component name.
bbbbbbbb	Is the database name. The system database names are: **DSNDB01** The DB2 directory database

	DSNDB04 The default database
	DSNDB06 The DB2 catalog database
	DSNDB07 The work file database
xxxxxxxx	Names the individual table space or index space.
*y***0001.A***ccc*	Identifies the data set. For table spaces and index spaces that can be reorganized with SHRLEVEL CHANGE or SHRLEVEL REFERENCE, *y* can be I or J, depending on whether REORG has been run. For more information about REORG, see Part 2 of *DB2 Utility Guide and Reference*. For other table spaces and index spaces, *y* is I. *ccc* is the partition number of a partitioned table space or index space, or the relative data set number of a simple or segmented table space or index space.

For example, if the catalog alias is DSNCAT, one of the DB2 directory data sets is named:

```
DSNCAT.DSNDBD.DSNDB01.DBD01.I0001.A001
```

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

```
DSNCAT.DSNDBD.DSNDB06.SYSDBASE.I0001.A001
```

2. **DEFINE CATALOG**

Acceptable values:	YES, NO
Default:	YES
Update:	see Part 2 (Volume 1) of *DB2 Administration Guide*
DSNZP*xxx*:	none

Specify whether you want to create a new ICF catalog. YES builds a new ICF catalog by using the alias you specified in the CATALOG ALIAS field. NO signals that the catalog that named in the CATALOG ALIAS field already exists; the CLIST does not create a new one.

If you specify YES, DB2 edits job DSNTIJCA which, when run, creates a user catalog and an alias for that catalog. DB2 creates the high-level qualifier of the catalog name by adding a number to the end of the alias that you defined in the CATALOG ALIAS field. If the alias has fewer than eight characters, DB2 appends a 1 to the end. For example, if you accepted the default of DSNCAT for field 1, the catalog that is created is named DSNCAT1.USER.CATALOG. If the alias has eight characters, DB2 changes the last character into a 1. If the last character is already a 1, DB2 changes the 1 into a 2.

3-9. **VOLUME SERIAL 1– VOLUME SERIAL 7**

Acceptable values:	any valid z/OS volume serial name
Default:	DSNV01–DSNV02
Update:	see 247 ("The update process")
DSNZP*xxx*:	none

Specify the volume serial numbers for the data sets defined by installation or migration. You cannot use an asterisk (*) as a pattern-matching character in this field; you must use a volume serial number.

Specifying more than one volume for VOLUME SERIAL 1 through VOLUME SERIAL 7 helps recovery and performance. A series of messages are produced that estimate space distribution on the specified volumes. See "CLIST calculations panel 2: DSNTIPC1" on page 243 for examples. If you specify fewer than six volumes, data is combined on them. If you specify six volumes, data is distributed as follows:

Field	**Is used for ...**
(VOLUME SERIAL 1) (Field 3 on Panel DSNTIPA2)	CLIST allocation. Only this volume is required for the tailoring session. VOLUME SERIAL 1 through VOLUME SERIAL 7 are not required until you begin running the tailored jobs. The default is DSNV01. This volume is used for *prefix*.NEW.SDSNTEMP and *prefix*.NEW.SDSNSAMP.
(VOLUME SERIAL 2) (Field 4 on Panel DSNTIPA2)	Non-VSAM data. It is used for *prefix*.DBRMLIB.DATA, *prefix*.RUNLIB.LOAD, and *prefix*.SRCLIB.DATA. The default is DSNV01.
(VOLUME SERIAL 3) (Field 5 on Panel DSNTIPA2)	The temporary data sets, the default and sample storage group, and the VSAM catalog (if a new one is created). The default value is DSNV02.
(VOLUME SERIAL 4) (Field 6 on Panel DSNTIPA2)	DB2 catalog and directory. If you leave this field blank, it is set to the value you that specified for field 4.
(VOLUME SERIAL 5) (Field 7 on Panel DSNTIPA2)	DB2 catalog indexes and directory indexes. If you leave this field blank, it is set to the value that you specified for field 5.
(VOLUME SERIAL 6) (Field 8 on Panel DSNTIPA2)	The first copy of the active log and the second copy of the bootstrap data set (BSDS). If you leave this field blank, it is set to the value that you specified for field 6.
(VOLUME SERIAL 7) (Field 9 on Panel DSNTIPA2)	The second copy of the active log and the first copy of the BSDS. If you leave this field blank, it is set to the value that you specified for field 7 on this panel.

If you accept the default for dual active logging, specify different volume serials for field 8 and field 9. Using different serial numbers causes DB2 to place the active logs on different disk devices.

During migration you can change volume serial 1, but you cannot change volume serials 2 through 7.

10. **PERMANENT UNIT NAME**

Acceptable values:	valid device type or unit name
Default:	3390
Update:	see 247 ("The update process")
DSNZP*xxx*:	none

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

- ICF catalog
- *prefix*.DBRMLIB.DATA.*suffix*
- *prefix*.RUNLIB.LOAD.*suffix*
- *prefix*.SRCLIB.DATA.*suffix*
- The two data sets that the DSNTINST CLIST generates:
  - *prefix*.NEW.SDSNTEMP
  - *prefix*.NEW.SDSNSAMP

The value identifies a direct access unit name for partitioned data sets and the ICF catalog. If you want to use different device types for different data sets, edit the installation or migration jobs after you complete the tailoring session. Some common device types are 3380, 3390, and 9340.

The value is sometimes used during IVP processing to place output (from COPY TABLESPACE, for example) on the device type that is specified here.

A change to this parameter during migration does not affect the ICF catalog, DB2 catalog, directory, or logs. The new value is used for data sets created during migration.

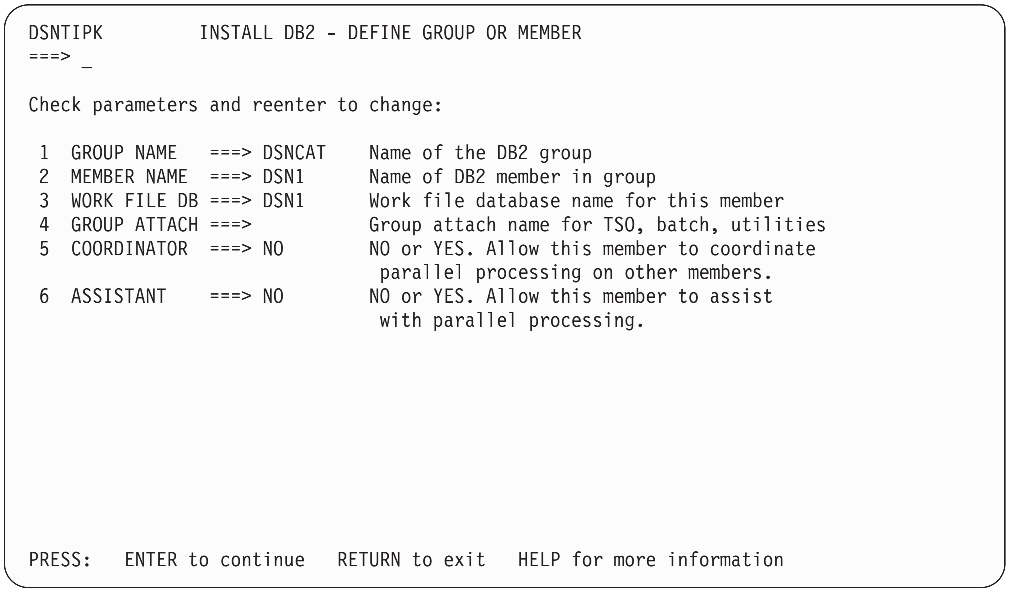
11. **TEMPORARY UNIT NAME**

Acceptable values:	valid device type or unit name
Default:	SYSDA
Update:	see 247 ("The update process")
DSNZP*xxx*:	DSN6SPRM VOLTDEVT

Specify the device type or unit name for allocating temporary data sets. It is the direct access or disk unit name that is used for the precompiler, compiler, assembler, sort, linkage editor, and utility work files in the tailored jobs and
| CLISTs. DB2 utilities that dynamically allocate temporary data sets also use this
| parameter.

Define group or member panel: DSNTIPK

This panel follows panel DSNTIPA2 when you select a data sharing function (GROUP, MEMBER, or ENABLE). You must start DB2 and IRLM group names with an alphabetic character. You should carefully consider the naming convention for a data sharing system. See *DB2 Data Sharing: Planning and Administration* for guidance on planning a naming convention before you choose names for the fields on panel DSNTIPK.



```
DSNTIPK         INSTALL DB2 - DEFINE GROUP OR MEMBER
===> _

Check parameters and reenter to change:

 1  GROUP NAME   ===> DSNCAT    Name of the DB2 group
 2  MEMBER NAME  ===> DSN1      Name of DB2 member in group
 3  WORK FILE DB ===> DSN1      Work file database name for this member
 4  GROUP ATTACH ===>           Group attach name for TSO, batch, utilities
 5  COORDINATOR  ===> NO        NO or YES. Allow this member to coordinate
                                 parallel processing on other members.
 6  ASSISTANT    ===> NO        NO or YES. Allow this member to assist
                                 with parallel processing.











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

Figure 10. Define group or member panel: DSNTIPK

1. **GROUP NAME**

Acceptable values:	1 to 8 characters consisting of A-Z, 0-9, $, #, @
Default:	DSNCAT
DSNZP*xxx*:	DSN6GRP GRPNAME

Specify the name of a new or existing DB2 data sharing group. The group name encompasses the entire data sharing group and forms the basis for the coupling facility structure names.

To avoid names that IBM uses for its z/OS cross-system coupling facility (XCF) groups, the first character must be an uppercase letter J-Z unless the name begins with DSN. Do not use SYS as the first three characters, and do not use UNDESIG as the group name.

2. **MEMBER NAME**

Acceptable values:	1 to 8 characters
Default:	DSN1
DSNZP*xxx*:	DSN6GRP MEMBNAME

Specify the name of a new or existing DB2 data sharing member.

Recommendation: Use the z/OS subsystem name. DB2 uses this name as its XCF member name. An example of a member name is DB1G. The member name can consist of the characters A-Z, 0-9, $, #, and @.

3. **WORK FILE DB**

Acceptable values:	1 to 8 characters
Default:	DSN1
DSNZP*xxx*:	none

Specify the name of the work file database for the DB2 member. Each DB2 member has its own work file database (called DSNDB07 in a non-data-sharing environment). One member of the data sharing group can have the name DSNDB07, but the recommendation is that you use a more meaningful name, such as WRKDSN1. You cannot specify a name that begins with DSNDB unless the name is DSNDB07.

4. **GROUP ATTACH**

Acceptable values:	1 to 4 characters
Default:	none
DSNHDECP:	SSID

Specify a generic group attachment name for batch programs, the call attachment facility (CAF), the RRS attachment facility (RRSAF), IMS, CICS Transaction Server, and utilities. An example of a group attachment name is DB0G. See *DB2 Data Sharing: Planning and Administration* for information about using the group
# attachment name. The value you specify here is also used in the IEFSSN*xx* member
# of SYS1.PARMLIB.

If you leave this field blank, DSNHDECP SSID contains the value that you
specified in the SUBSYSTEM NAME field on panel DSNTIPM.

5. **COORDINATOR**

Acceptable values:	YES, NO
Default:	NO
DSNZP*xxx*:	DSN6GRP COORDNTR

Specify whether this DB2 member can coordinate parallel processing on other members of the group. If you specify NO, only this DB2 member can process a query. If you specify YES, a read-only query on this DB2 member can be processed in part on other members of the group.

6. **ASSISTANT**

Acceptable values:	YES, NO
Default:	NO
DSNZP*xxx*:	DSN6GRP ASSIST

Specify whether this DB2 member can assist a parallelism coordinator with parallel processing. If you specify NO, this member is not considered as an assistant at either bind time or run time. If you specify YES, this member is considered as an assistant at both bind time and run time. To qualify as an assistant at run time, the

VPPSEQT and VPXPSEQT buffer pool thresholds of this member must each be greater than zero.

System resource data set names panel: DSNTIPH

The entries on this panel name the bootstrap data sets, active logs, and archive logs. They also specify the number of copies (one for single logging or two for dual logging) for the active and archive logs.

Fields 1, 2, 4, 5, 7, and 8 on the DSNTIPH panel contain the prefix that was entered in the CATALOG ALIAS field on installation panel DSNTIPA2. If you scroll back to panel DSNTIPA2 and change the CATALOG ALIAS value, the values on DSNTIPH for fields 1, 2, 4, 5, 7, and 8 change. When you scroll from panel DSNTIPA2 to panel DSNTIPH, check these values and enter them again if necessary. In MIGRATE or UPDATE modes, the CATALOG ALIAS value cannot be changed, so the fields on DSNTIPH are not affected.

Dual logging improves reliability of recovering and, for active log reads, eases device contention.

Recommendations:

- Specify dual logging for both active and archive logs. If you specify dual logging, and an error occurs during offload to the archive logs, DB2 restarts the archive process using the second copy of the active log.
- If you use dual active logging, place the two active logs on different disk volumes and, ideally, on different channels and control units. To do that, specify different volume serial numbers for VOLUME SERIAL 6 and VOLUME SERIAL 7 (fields 8 and 9 on panel DSNTIPA2).

| **If you are migrating,** DB2 UDB for z/OS Version 8 adopts your Version 7 BSDS and active logs. Therefore, you cannot change the parameters that affect the characteristics of these objects during migration. After your DB2 subsystem is in new-function mode, you can convert the BSDS. However, after you have entered new-function mode, you can update the parameters that affect the BSDS and active logs.

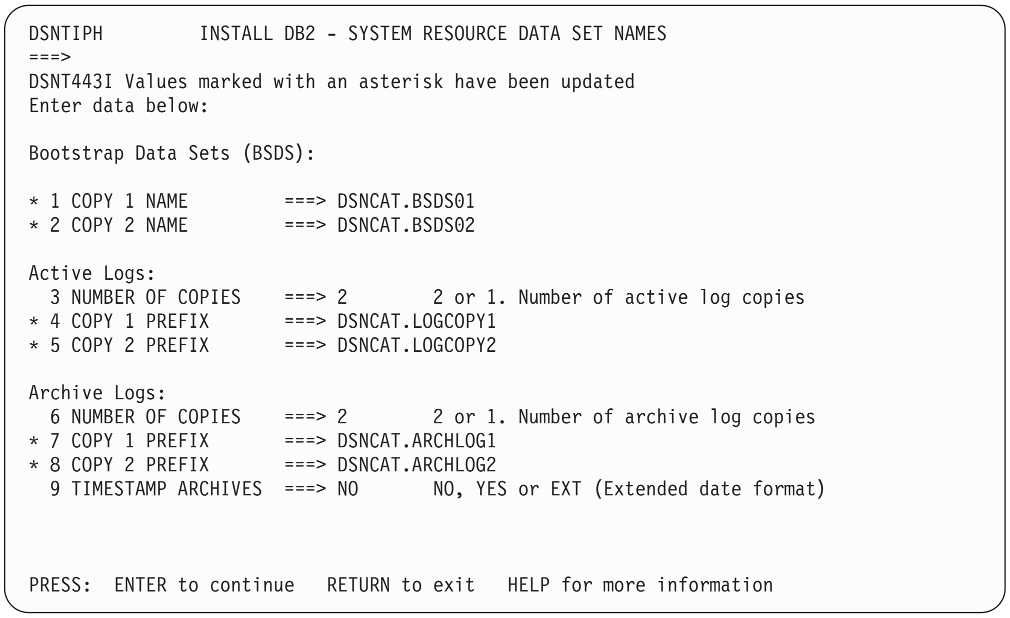

```
DSNTIPH         INSTALL DB2 - SYSTEM RESOURCE DATA SET NAMES
===>
DSNT443I Values marked with an asterisk have been updated
Enter data below:

Bootstrap Data Sets (BSDS):

* 1 COPY 1 NAME         ===> DSNCAT.BSDS01
* 2 COPY 2 NAME         ===> DSNCAT.BSDS02

Active Logs:
  3 NUMBER OF COPIES    ===> 2       2 or 1. Number of active log copies
* 4 COPY 1 PREFIX       ===> DSNCAT.LOGCOPY1
* 5 COPY 2 PREFIX       ===> DSNCAT.LOGCOPY2

Archive Logs:
  6 NUMBER OF COPIES    ===> 2       2 or 1. Number of archive log copies
* 7 COPY 1 PREFIX       ===> DSNCAT.ARCHLOG1
* 8 COPY 2 PREFIX       ===> DSNCAT.ARCHLOG2
  9 TIMESTAMP ARCHIVES  ===> NO      NO, YES or EXT (Extended date format)



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

Figure 11. System resource data set names: DSNTIPH

1. **COPY 1 NAME**

Acceptable values:	valid data set name; 1 to 33 characters
Default:	DSNCAT.BSDS01 or DSNCAT.DSN1.BSDS01
Update:	see 247 ("The update process"); not during migration
DSNZP*xxx*:	none

Specify the fully qualified name of the first copy of the bootstrap data set. For non-data-sharing environments, the default prefix is DSNCAT.BSDS*xx*. For data sharing environments, the default prefix is DSNCAT.DSN1.BSDS*xx*. The resulting data set name is DSNCAT.BSDS*xx*.A*nnnnnnn* or DSNCAT.DSN1.BSDS*x*.A*nnnnnnn* where:

- DSNCAT is the value that you specified for CATALOG ALIAS (on installation panel DSNTIPA2). You can change this portion of the data set prefix on this panel. If you change it, you need to supply another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the value that you specified for MEMBER NAME on panel DSNTIPK.
- *xx* is **01** for the first copy of the logs and **02** for the second copy.
- A*nnnnnnn* is generated by DB2.

For the definition of a valid data set name, see 97.

2. **COPY 2 NAME**

Acceptable values:	valid data set name; 1 to 33 characters
Default:	DSNCAT.BSDS02 or DSNCAT.DSN1.BSDS02
Update:	see 247 ("The update process); not during migration
DSNZP*xxx*:	none

Specify the fully qualified name of the second copy of the bootstrap data set. For the definition of a valid data set name, see 97.

3. **NUMBER OF COPIES**

Acceptable values:	1, 2
Default:	2
Update:	see 247 ("The update process"); not during migration
DSNZP*xxx*:	DSN6LOGP TWOACTV

Specify the number of copies of the active log that DB2 is be maintain: 2 (dual logging) or 1 (single logging). Dual logging increases reliability of recovery. If your DB2 subsystem creates copies of the archive log on tape, two tape drives must be available during the offload process.

4. **COPY 1 PREFIX**

Acceptable values:	valid data set name prefix; 1 to 30 characters
Default:	DSNCAT.LOGCOPY1 or DSNCAT.DSN1.LOGCOPY1
Update:	see 247 ("The update process"); not during migration
DSNZP*xxx*:	none

Specify the prefix for the first copy of the active log data sets. For non-data-sharing environments, the default prefix is DSNCAT.LOGCOPY*x*. For data sharing environments, the default prefix is DSNCAT.DSN1.LOGCOPY*x*. The resulting data set name is DSNCAT.LOGCOPY*x*.A*nnnnnnn* or DSNCAT.DSN1.LOGCOPY*x*.A*nnnnnnn*, where:

- DSNCAT is the value that you specified for CATALOG ALIAS (field 1 on installation panel DSNTIPA2). You can change this portion of the data set prefix on this panel. If you change it, you need to specify another catalog alias. This additional catalog alias is not automatically defined by the installation process.
- DSN1 is the value you specified for MEMBER NAME on panel DSNTIPK.
- LOGCOPY is part of the data set prefix that you can change on this panel.
- *x* is **1** for the first copy of the logs and **2** for the second copy.
- *nn* is the data set number.

For the definition of a valid data set name, see 97.

5. **COPY 2 PREFIX**

Acceptable values:	valid data set name prefix; 1 to 30 characters
Default:	DSNCAT.LOGCOPY2 or DSNCAT.DSN1.LOGCOPY2
Update:	see 247("The update process"); not during migration
DSNZP*xxx*:	none

Specify the prefix for the second copy of the active log data sets. See the description in field 4. If you are using single logging, accept the default value. Do not leave the entry blank.

6. **NUMBER OF COPIES**

Acceptable values:	1, 2
Default:	2
Update:	option 3 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP TWOARCH

Specify the number of copies of the archive log that DB2 is to produce during offloading: 2 (dual logging) or 1 (single logging). Dual logging increases reliability of recovery.

7. **COPY 1 PREFIX**

Acceptable values:	valid data set name prefix; 1 to 35 characters
Default:	DSNCAT.ARCHLOG1 or DSNCAT.DSN1.ARCLG1
Update:	option 3 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCPFX1

Specify the prefix of the first copy of the archive log data set. For definitions of valid data set names, see 97.

8. **COPY 2 PREFIX**

Acceptable values:	valid data set name prefix; 1 to 35 characters
Default:	DSNCAT.ARCHLOG2 or DSNCAT.DSN1.ARCLG2
Update:	option 3 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCPFX2

Specify the prefix of the second copy of the archive log data set. If you are using single logging, accept the default value. Do not leave the entry blank.

9. **TIMESTAMP ARCHIVES**

Acceptable values:	NO, YES, EXT
Default:	NO
Update:	option 3 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP TSTAMP

Specify whether the date and time of creation of the DB2 archive log data set is to be placed in the archive log data set name.

If you specify NO, the archive data set name does not contain a timestamp.

If you specify YES, the maximum allowable length of the user-controlled portion of the archive log prefix is reduced from 35 characters to 19 characters. This reduction in size permits the 16-character date and time qualifiers (timestamp) to be added to the archive log data set prefix. The timestamp format is as follows:

```
.Dyyddd.Thhmmsst,
```

where:

D	is the letter D.
yy	is the last two digits of the year.
ddd	is the day of the year.
T	is the letter T.
hh	is the hour.
mm	is the minutes.
ss	is the seconds.
t	is tenths of a second.

If you specify EXT, the archive data set name contains a timestamp with an **extended** date component in the format:

```
.Dyyyyddd.
```

A value of EXT in this field causes the lengths of the values that are entered for field COPY 1 PREFIX and field COPY 2 PREFIX to be audited to ensure that neither exceeds 17 bytes (19 bytes for other settings of TIMESTAMP ARCHIVES).

Data set names panel 1: DSNTIPT

The entries on this panel establish data set names for the DB2 libraries that are used in the DB2 CLIST and JCL that DB2 provides. The values that you enter on this panel are edited into all pertinent sample and installation jobs.

You can fill in these values in one of three ways: same data set name prefix, no data set name prefix, or a new data set name prefix. Table 33 summarizes these selections.

Table 33. Summary of values

If you use ...	Then
Same data set name prefix or data set names	Current data sets are deleted and reallocated for installation and migration.
No data set names	No new output is created. Previous output remains intact.
New prefix	Output is saved in new data set. Previous output remains intact.

The following warning message is displayed for any output data set that already exists:

```
DSNT434I WARNING, DATA SETS MARKED WITH ASTERISKS EXIST AND WILL BE OVERWRITTEN
```

To avoid deleting these data sets, take one of the following actions:
- Press Enter to leave the installation process.
- Change the data set names.

Press Enter again if you want to continue; this overwrites your data sets.

| When you are in update or ENFM mode, this panel is displayed immediately after
| panel DSNTIPA1. This allows you to check the SDSNSAMP data set name to see if
| it is the one you want to use for the DSNTIJUZ job. Data sets are not deleted or
| reallocated if you use the same name. Instead, the data set is compressed, and only
| the DSNTIJUZ member is replaced within the data set. Other members in the data
| set are left unchanged.

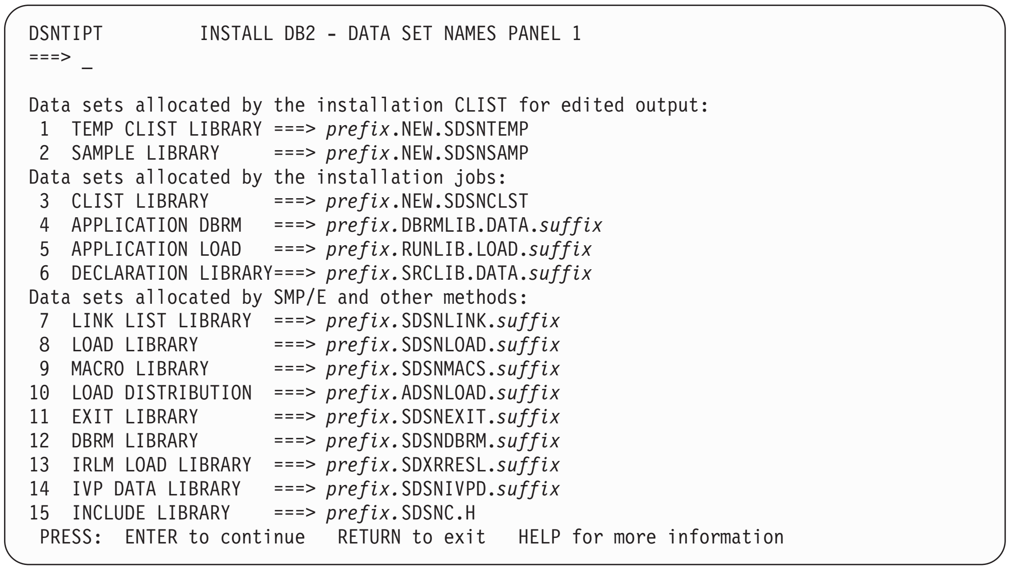

```
DSNTIPT         INSTALL DB2 - DATA SET NAMES PANEL 1
===> _

Data sets allocated by the installation CLIST for edited output:
 1  TEMP CLIST LIBRARY ===> prefix.NEW.SDSNTEMP
 2  SAMPLE LIBRARY     ===> prefix.NEW.SDSNSAMP
Data sets allocated by the installation jobs:
 3  CLIST LIBRARY      ===> prefix.NEW.SDSNCLST
 4  APPLICATION DBRM   ===> prefix.DBRMLIB.DATA.suffix
 5  APPLICATION LOAD   ===> prefix.RUNLIB.LOAD.suffix
 6  DECLARATION LIBRARY===> prefix.SRCLIB.DATA.suffix
Data sets allocated by SMP/E and other methods:
 7  LINK LIST LIBRARY  ===> prefix.SDSNLINK.suffix
 8  LOAD LIBRARY       ===> prefix.SDSNLOAD.suffix
 9  MACRO LIBRARY      ===> prefix.SDSNMACS.suffix
10  LOAD DISTRIBUTION  ===> prefix.ADSNLOAD.suffix
11  EXIT LIBRARY       ===> prefix.SDSNEXIT.suffix
12  DBRM LIBRARY       ===> prefix.SDSNDBRM.suffix
13  IRLM LOAD LIBRARY  ===> prefix.SDXRRESL.suffix
14  IVP DATA LIBRARY   ===> prefix.SDSNIVPD.suffix
15  INCLUDE LIBRARY    ===> prefix.SDSNC.H
 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 12. Data set names panel 1: DSNTIPT

TEMP CLIST LIBRARY and SAMPLE LIBRARY are data sets that are allocated by the installation CLIST for edited output. CLIST LIBRARY is allocated by DSNTIJVC. If the input member is DSNTIDXA (field 6 on installation panel DSNTIPA1 on 94), the three data sets default to *prefix.*NEW.SDSNTEMP, *prefix.*NEW.SDSNCLST, and *prefix.*NEW.SDSNSAMP respectively, where *prefix* is the value that is entered for field 4 on installation panel DSNTIPA1. Table 34 shows the job-tailoring fields.

Table 34. Job-tailoring fields

Mode	Tailored output	No tailored output
Installing	All three fields entered	All three fields blank
Migrating	All three fields entered	All three fields blank
Updating	SAMPLE LIBRARY entered	SAMPLE LIBRARY blank

DB2 adds blanks to these fields after a successful tailoring session to avoid writing over the tailored output.

1. **TEMP CLIST LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix.*NEW.SDSNTEMP
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the data set name where edited CLISTs are to be placed. This field must not be blank if you are tailoring output.

2. **SAMPLE LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix.*NEW.SDSNSAMP
Update:	option 4 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the name of the edited JCL library. In update mode, the new sample library data set is not reallocated. It is compressed and member DSNTIJUZ is overwritten. This field must not be blank if you are tailoring output.

3. **CLIST LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.NEW.SDSNCLST
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the data set name into which job DSNTIJVC loads all CLISTs. This field must not be blank if you are tailoring output.

Fields 4, 5, and 6 are for DB2-provided sample applications. The names of your own development libraries most likely are different from the defaults that are shown here. Job DSNTIJMV references another set of DBRMLIB, RUNLIB, and SRCLIB data sets for SYS1.PROCLIB. See "Installation step 1: Define DB2 to z/OS: DSNTIJMV" on page 252 for more information. These fields must not be blank.

4. **APPLICATION DBRM**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.DBRMLIB.DATA.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the library for DB2 sample application DBRMs.

5. **APPLICATION LOAD**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.RUNLIB.LOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the DB2 sample application load module library.

6. **DECLARATION LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SRCLIB.DATA.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the DB2 declaration library for sample application include files.

Fields 7 through 13 specify the names of data sets that are allocated during SMP processing. These fields must not be blank.

7. **LINK LIST LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNLINK.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the APF-authorized DB2 early code library.

8. **LOAD LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNLOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the main APF-authorized DB2 load module library.

9. **MACRO LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNMACS.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the library that contains the CICS and IMS attachment facility macros, the initialization parameter macros, and some data-mapping macros that are needed for some applications.

10. **LOAD DISTRIBUTION**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.ADSNLOAD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the distribution load module library.

11. **EXIT LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNEXIT.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the library where your DSNZP*xxx* module, DSNHDECP module, and exit routines are to be placed. When you use *prefix*.SDSNLOAD and *prefix*.SDSNEXIT together, list *prefix*.SDSNEXIT first to override the IBM defaults in *prefix*.SDSNLOAD.

12. **DBRM LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNDBRM.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the library where the DBRMs that are shipped with DB2 are to be placed.

13. **IRLM LOAD LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDXRRESL.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the IRLM load library data set to use in the IRLM procedure.

14. **IVP DATA LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNIVPD.*suffix*
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the data set name of SDSNIVPD. SDSNIVPD is the SMP/E target library for the DB2 installation verification procedure (IVP) input data and for the expected output from the sample applications.

15. **INCLUDE LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	*prefix*.SDSNC.H
Update:	cannot change during update or ENFM
DSNZP*xxx*:	none

Specify the name of the include library data set. This library is used in the DB2 language PROCS for C and C++.

Data set names panel 2: DSNTIPU

The entries on this panel and DSNTIPW establish data set names for other product libraries. The values that you enter on these panels are edited into sample and installation jobs. If you do not have the product, accept the default. Jobs for those particular products should not be run.

DB2 makes assumptions about which one of the possible C, C++, and PL/I compilers that you are using, depending on the values you supply or leave as default in the C, C++, and PL/I fields.

Many data set names for other products appear in the jobs. You can enter most of these data sets on this panel and on installation panel DSNTIPW. These names are shown in Table 35 as they appear in the jobs that are shipped with DB2. Change the names of the data sets if they are different at your site.

Table 35. Data set names that are used in jobs for related products

Job	Data set name	Function
DSNTEJ1	SYS1.MACLIB	Assembler macro library
	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ1L	CEE.SCEELKED	Language Environment® linkage editor library
	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ1P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ1U	CEE.SCEERUN	Language Environment dynamic runtime library
	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ2A	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ2C	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2D	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2E	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ2F	SYS1.MACLIB	Assembler macro library
	SYS1.VSF2FORT	VS Fortran runtime library
DSNTEJ2P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ3C	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ3P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ4C	IMSVS.RESLIB	IMS linkage editor library
	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ4P	IMSVS.RESLIB	IMS linkage editor library
	CEE.SCEELKED	PL/I linkage editor base library
	CEE.SCEERUN	PL/I dynamic runtime base library
DSNTEJ5A	CICS410.SDFHLOAD	CICS command translator and linkage editor
	CICS410.SDFHMAC	CICS macro library
	SYS1.MACLIB	Assembler macro library
DSNTEJ5C	CICS410.SDFHLOAD	CICS command translator and linkage editor library
	IGY.SIGYCOMP	Enterprise COBOL for z/OS
		See also the list of libraries that are used by DSNH CLIST in *DB2 Command Reference*
DSNTEJ5P	CICS410.SDFHLOAD	CICS command translator and linkage editor library
	CICS410.SDFHPLI	CICS PL/I linkage editor library
	CEE.SCEELKED	Language Environment link editor library
DSNTEJ6D	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6P	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6R	CEE.SCEERUN	Language Environment dynamic runtime library
	CEE.SCEEH.H	C library headers
	CEE.SCEELKED	Language Environment linkage editor library
DSNTEJ6S	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6T	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6U	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6V	CEE.SCEERUN	Language Environment dynamic runtime library

Table 35. Data set names that are used in jobs for related products (continued)

Job	Data set name	Function
DSNTEJ6W	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ6Z	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ61	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ62	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ63	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ64	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ65	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ7	SYS1.SORTLIB	DFSORT load modules (can be deleted if DFSORT is in link list)
DSNTEJ71	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ73	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ75	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ76	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ77	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJ78	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTEJXP	CEE.SCEERUN	Language Environment dynamic runtime library
DSNTIJMV	SYS1.MACLIB	Assembler macro library
	CEE.SCEERUN	Language Environment dynamic runtime library
	CEE.SCEELKED	Language Environment linkage editor library
	EDCPRLK	Language Environment pre-link editor library
	CEE.SCEEMSGP	Language Environment pre-link message file
	CBC.SCCNCMP	C/C++ for z/OS compiler library
	CCNDRVR	C/C++ compiler load module
	CEE.SCEEH.H	C library headers
	CEE.SCLBH.HPP	C++ library headers
	CEE.SCEECPP	C++ autolink library
	CBC.SCLBCPP	C++ class library
	CICS.SCLBCPP	CICS library for COBOL
	CICS.SDFHLOAD	CICS command translator and linkage editor
	CICS.SDFLPLI	CICS library for PL/I
	prefix.SDSNLOAD(DSNHPC)	DB2 precompiler
	prefix.SDSNLOAD	DB2 linkage editor library
	DSNHPC	DB2 precompiler module
	GDDM.SADMSAM	GDDM macro library
	GDDM.SADMMOD	GDDM load module library
	IGY.SIGYCOMP	Enterprise COBOL for z/OS compiler library
	IGYCRCTL	Enterprise COBOL for z/OS compiler load module
	IMSVS.RESLIB	IMS linkage editor library
	ISP.SISPLOAD	ISPF ISPLINK module
	IBM.SIBMCZMP	Enterprise PL/I for z/OS compiler library
	IBMZPLI	Enterprise PL/I compiler load module
	SYS1.VSF2FORT	VS Fortran runtime library

When the compiler fields are left blank, the DSNH CLIST and the provided JCL procedures operate differently. The DSNH CLIST issues a specific call statement, using the default load module data set name as the argument of the call. The JCL procedures use the z/OS link list to find the data set in which the load module resides.

Use this panel to define the data set names of your IBM Language Environment, C/370™ or C/C++, COBOL, FORTRAN, and PL/I program product libraries. For more information about these libraries, consult the appropriate program product documentation.

Data sets that are specified on this panel are used by the DB2 installation process to tailor the DB2 language procedures that are generated by installation job DSNTIJMV:

- DSNHASM can be used to prepare DB2 programs using assembly language
- DSNHC can be used to prepare DB2 programs that use C.
- DSNHCPP can be used to prepare DB2 programs that use C++.
- DSNHCPP2 can be used to prepare a class and a client for a DB2 object-oriented program that use C++.
- DSNHCPPS contains the header file search path that is to be used by DSNHCPP and DSNHCPP2.
- DSNHFOR can be used to prepare DB2 programs using FORTRAN
- DSNHICOB can be used to prepare DB2 programs using COBOL
- DSNHPLI can be used to prepare DB2 programs using PL/I
- DSNHSQL can be used to prepare DB2 SQL procedures

Data sets that you specify on this panel are also used by the DB2 installation process to tailor the DB2 Interactive (DB2I) program preparation CLIST, DSNH.

Use fields 1 to 3 to specify the IBM Language Environment runtime environment, link editor, and pre-link editor message libraries. The CLIST assumes that these libraries are used by all language products except FORTRAN.

Use fields 5 through 12 to define C/370 Version 2 Release 1 (C only), AD/Cycle®, C/370 Version 1 Release 2 (C only), C/C++ for MVS/ESA™ Version 3 Release 2, C/C++ for OS/390, or C/C++ for z/OS. If you need to define C++, you must define C/C++ for C as well.

If C is not installed on your system:

- Accept the default values for fields 5-12.
- Do not run IVP jobs DSNTEJ2D, DSNTEJ2U, DSNTEJ6D, DSNTEJ6R, DSNTEJ6T, DSNTEJ63, DSNTEJ6W, DSNTEJ6Z, DSNTEJ64, DSNTEJ65, DSNTEJ71, DSNTEJ73, and DSNTEJ75.

If C++ is not installed on your system:

- Accept the default values for fields 5-12.
- Do not run jobs DSNTEJ2E or DSNTEJ6V. Skip steps PH02US08 and PH02US09 of IVP job DSNTEJ2U.
- Remove all statements that refer to DAYNAME and MONTHNAME from part DSNTESU in the *prefix*.SDSNSAMP library if C++ is not available.

If COBOL is not installed on your system:

- Accept the default value for field 13.
- Do not run IVP jobs DSNTEJ2C, DSNTEJ3C, DSNTEJ4C, DSNTEJ5C, DSNTEJ61, DSNTEJ62, DSNTEJ76, DSNTEJ77, and DSNTEJ78.

If FORTRAN is not installed on your system:

- Accept the default values for fields 14 and 15.
- Do not run IVP job DSNTEJ2F.

If PL/I is not installed on your system:

- Accept the default values for fields 16 and 17.
- Do not run IVP jobs DSNTEJ1P, DSNTEJ2P, DSNTEJ3P, DSNTEJ4P, DSNTEJ5P, DSNTEJ6P, DSNTEJ6S, and DSNTEJ6U.

```
DSNTIPU         INSTALL DB2 - DATA SET NAMES PANEL 2
===>

Enter data set names below:
 1  IBM LE RUNTIME LIBRARY   ===> CEE.SCEERUN
 2  IBM LE LINK EDIT LIB     ===> CEE.SCEELKED
 3  IBM LE PRELINK MSG LIB   ===> CEE.SCEEMSGP

 4  HIGH LEVEL ASSEMBLER LIB ===>
 5  C/CPP COMPILER MODULE    ===> CCNDRVR
 6  C/CPP COMPILER LIBRARY   ===> CBC.SCCNCMP
 7  C/CPP HEADER LIBRARY     ===> CEE.SCEEH.H
 8  C/370 COMPILER MESSAGES  ===>
 9  CPP CLASS LIB HEADERS    ===> CBC.SCLBH.HPP
10  CPP AUTO CALL LIBRARY    ===> CEE.SCEECPP
11  CPP CLASS LIBRARY        ===> CBC.SCLBCPP
12  CPP PROCEDURE LIBRARY    ===> CBC.SCBCUTL
13  COBOL COMPILER LIBRARY   ===>
14  FORTRAN COMPILER LIBRARY ===>
15  FORTRAN LINK EDIT LIB    ===> SYS1.VSF2FORT
16  PL/I COMPILER MODULE     ===> IBMZPLI
17  COMPILER LIBRARY         ===>
PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 13. Data set names panel 2: DSNTIPU

1. **IBM LE RUNTIME LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	CEE.SCEERUN
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the IBM Language Environment dynamic runtime library. The data set name typically includes the qualifier SCEERUN. Leave this field blank if SCEERUN is in the link list. If you enter a value in this field, it will be used in the STEPLIB concatenation of the JCL procedures generated by installation job DSNTIJMV, including the DSNDBM1, DSNDIST, and DSNSPAS address spaces for DB2, the DB2–supplied WLM procedures DSNWLM and DSNCICS, the DB2 language procedures, and the JOBLIB concatenation of many IVP jobs.

If you plan to use DB2 to run XPLINK or AMODE 64 applications in z/OS Version 1 Release 6, provide the SCEERUN and SCEERUN2 libraries for IBM Language Environment in the z/OS program search order. See *z/OS Language Environment Customization* for more information.

2. **IBM LE LINK EDIT LIB**

Acceptable values:	valid data set name: see 97
Default:	CEE.SCEELKED
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the IBM Language Environment linkage editor library. The data set name typically includes the qualifier SCEEKLED. SCEELKED is required to link-edit load modules for C, C++, COBOL, and PL/I application programs. The value that you enter here is used to customize the DB2 language procedures and the DSNH CLIST.

3. **IBM LE PRELINK MSG LIB**

Acceptable values:	valid data set name: see 97
Default:	CEE.SCEEMSGP
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for messages that are issued by the IBM Language Environment pre-linkage editor (EDCPRLK). The SMP/E target data set name typically includes the qualifier SCEEMSGP. SCEEMSGP is required to pre-link edit load modules for C, C++, COBOL, and PL/I application programs. The value that you enter here is used to customize DB2 language procedures and the DSNH CLIST.

4. **HIGH LEVEL ASSEMBLER LIB**

Acceptable values:	blank, or valid data set name: see 97
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the assembler load module library. The data set name typically includes the qualifier SASMMOD1. The value that you specify for this field is used to customize the DSNHASM language procedure and the DSNH CLIST. It is also added to the STEPLIB concatenation of each DB2-provided job that uses the assembler. You can leave this field blank if the library is in the link list.

5. **C/CPP COMPILER MODULE**

Acceptable values:	blank, or valid data set name: see 97
Default:	CCNDRVR
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the load module name of the C/370 or C/C++ compiler used on your system. The default value is CCNDRVR for C/C++ for z/OS. The value that you enter here is used to determine the configuration for the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, the DSNH CLIST, and the IVP jobs that use C and C++. The value is used as follows:

- If the entry begins with the string ’EDC’, the CLIST configures your system to use C/370.

  Important: You cannot use C/370 and C++.
- If the entry begins with the string ’CBC’, the CLIST configures your system to use C/C++ for OS/390 and C/C++ for MVS/ESA Version 3 Release 2.
- If the entry begins with any other value, including blanks, the CLIST configures your system to use C/C++ for z/OS.

6. **C/CPP COMPILER LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	CEE.SCCNCMP
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the compiler library for C/C++ or C/370.

- For C/C++ for z/OS, the SMP/E target data set name typically includes the qualifier SCCNCMP.
- For C/C++ for OS/390, the SMP/E target data set name typically includes the qualifier SCBCCMP.
- For C/C++ for MVS/ESA Version 3 Release 2, the SMP/E target data set name typically includes the qualifier SCBC3CMP.
- For C/370, the SMP/E target data set name typically includes the qualifier SEDCDCMP or SEDCCOMP.

This value is used to customize the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, and the DSHN CLIST. This field can be left blank if the compiler library is in the link list.

7. **C/CPP HEADER LIBRARY**

Acceptable values:	blank, or valid load module name: see 97
Default:	CEE.SCEEH.H
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the header include library for C/C++ or C/370.

- For C/C++, the SMP/E target data set name typically includes the qualifier SCEEH.H.
- For C/370, the SMP/E target data set name typically includes the qualifier SEDCDCMP or SEDCCOMP.

This value is used to customize the DSNHC, DSNHCPP, DSNHCPP2, and DSNHSQL language procedures, and the DSHN CLIST. This field can be left blank if the compiler library is in the link list.

8. **C/370 COMPILER MESSAGES**

Acceptable values:	blank, or valid data set name: see 97
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the message library for the C/370 compiler. The data set name typically includes the qualifier SEDCDMSG or SEDCMSGS. If you are using C/C++, leave this field blank. This value is used the customize the DSNHC and DSHNSQL language procedures and the DSNH CLIST..

You can specify C/C++ for MVS/ESA V3R1, or C/C++ for MVS/ESA V3R2. These are shipped as single products, but you need to define them separately on this line and on line 1, depending on your need for C or C++. If you specify a name in this field, a STEPLIB is added to the compile step of the DSNHCPP and DSNHCPP2 procedures in job DSNTIJMV, and to the C++ portion of the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

9. **CPP CLASS LIB HEADERS**

Acceptable values:	blank, or valid data set name: see 97
Default:	CBC.SCLBH.HPP
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the C++ class header files. Accept the default value if C++ is not available on your system.

- For C/C++ for z/OS and C/C++ for OS/390, the data set name typically includes the qualifier SCLBH.HPP.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCLB3H.HPP.

If you specify a name in this field, it is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

10. **CPP AUTO CALL LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	CEE.SCEECPP
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the C++ auto call library. This data set name typically includes the qualifier SCEECPP. Accept the default value if C++ is not available on your system. The value that you enter here is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

11. **CPP CLASS LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	CEE.SCLBCPP
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the C++ class library. Accept the default value if C++ is not available on your system.

- For C/C++ for z/OS and C/C++ for OS/390, the data set name typically includes the qualifier SCLBCPP.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCLB3CPP.

The value that you enter here is used to customize the DSNHCPP and DSNHCPP2 language procedures and the DSNH CLIST.

12. **CPP PROCEDURE LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CEE.SCCNUTL
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the C++ procedure library.

- For C/C++ for z/OS, the data set name typically includes the qualifier SCCNUTL.
- For C/C++ for OS/390, the data set name typically includes the qualifier SCBCUTL.
- For C/C++ for MVS/ESA V3R2, the data set name typically includes the qualifier SCBC3UTL.

13. **COBOL COMPILER LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the compiler library for COBOL. The data set name typically includes the qualifier SIGYCOMP. The value that you enter here is used to customize the DSNHICOB language procedure and the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

14. **FORTRAN COMPILER LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the FORTRAN compiler library. If you specify a value here, it is used to customize the DSNHFOR language procedure and the DSNH CLIST. You can leave this field blank if the compiler library is in the link list.

15. **FORTRAN LINK EDIT LIB**

Acceptable values:	blank, or valid data set name: see 97
Default:	SYS1.VSF2FORT
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the FORTRAN linkage editor library. The value that you enter here is used to customize the DSNHFOR language procedure and the DSNH CLIST. Accept the default value if FORTRAN is not available on your system, and do not run IVP job DSNTIJ2F.

16. **PL/I COMPILER MODULE**

Acceptable values:	blank, or valid data set name: see 97
Default:	IBMZPLI
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the load module name of the PL/I used on your system. The value that you enter here is used to configure the DSNHPLI language procedure, the DSNH CLIST, and the IVP jobs that use PL/I. The default value is IBMZPLI for Enterprise PL/I. If the value that you enter here begins with the string 'IEL', the CLIST configures your system for IBM PL/I for OS/390 & VM.

17. **PL/I COMPILER LIBRARY**

Acceptable values:	blank, or valid data set name: see 97
Default:	NONE
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the PL/I compiler library.

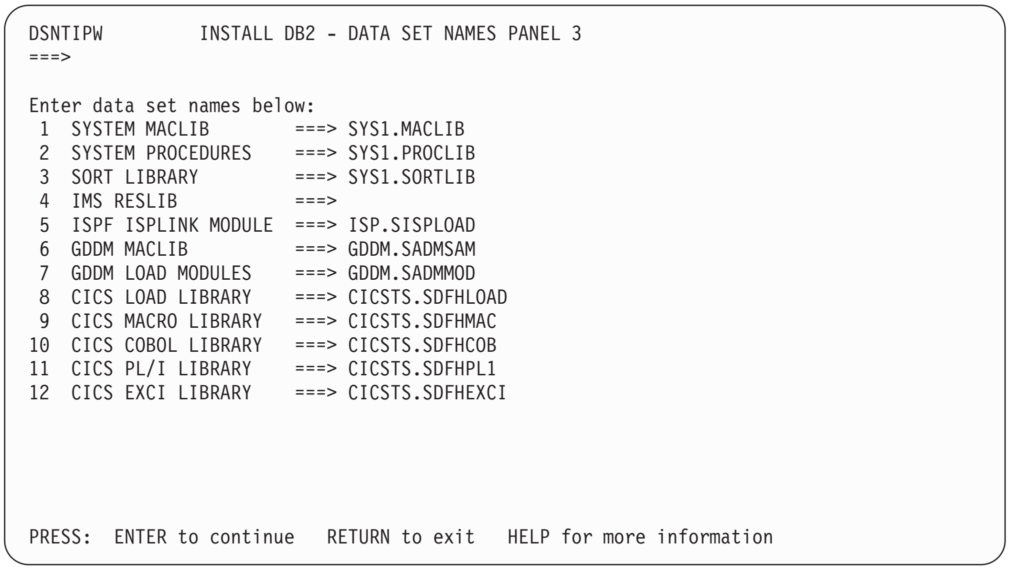
- For Enterprise PL/I, the data set name typically includes the qualifier SIBMZCMP.
- For IBM PL/I for OS/390 & VM, the data set name typically includes the qualifier SIELCOMP.

The value that you enter here is used to customize the DSNHPLI language procedure and the DSNH CLIST. You can leave the field blank if the PL/I compiler library is in the system link list.

Data set names panel 3: DSNTIPW

The entries on this panel establish data set names for the libraries of other products in your system. The values entered on this panel are edited into all pertinent sample and installation jobs. If you do not have the product, accept the default. The default cannot be blank.

Many data set names for other products appear in the jobs. You can enter most of these data set names on this panel and on installation panel DSNTIPU. These names are shown in Table 35 on page 118 as they appear in the jobs that are shipped with DB2. Change the names of the data sets if they are different at your site.



```
DSNTIPW          INSTALL DB2 - DATA SET NAMES PANEL 3
===>

Enter data set names below:
 1  SYSTEM MACLIB        ===> SYS1.MACLIB
 2  SYSTEM PROCEDURES    ===> SYS1.PROCLIB
 3  SORT LIBRARY         ===> SYS1.SORTLIB
 4  IMS RESLIB           ===>
 5  ISPF ISPLINK MODULE  ===> ISP.SISPLOAD
 6  GDDM MACLIB          ===> GDDM.SADMSAM
 7  GDDM LOAD MODULES    ===> GDDM.SADMMOD
 8  CICS LOAD LIBRARY    ===> CICSTS.SDFHLOAD
 9  CICS MACRO LIBRARY   ===> CICSTS.SDFHMAC
10  CICS COBOL LIBRARY   ===> CICSTS.SDFHCOB
11  CICS PL/I LIBRARY    ===> CICSTS.SDFHPL1
12  CICS EXCI LIBRARY    ===> CICSTS.SDFHEXCI





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

Figure 14. Data set names panel 3: DSNTIPW

1. **SYSTEM MACLIB**

Acceptable values:	valid data set name: see 97
Default:	SYS1.MACLIB
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the assembler macro library.

2. **SYSTEM PROCEDURES**

Acceptable values:	valid data set name: see 97
Default:	SYS1.PROCLIB
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the system procedures library.

3. **SORT LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	SYS1.SORTLIB
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the name of the data set where the DFSORT load module resides. DFSORT is required. If your load library is not in the link list, you can change the DSNUPROC JCL procedure in job DSNTIJMV.

If DFSORT is installed as your primary z/OS sort product, you do not need to take any action to make DFSORT available to DB2. If multiple releases of DFSORT are installed, ensure that DFSORT R14 is found first in the system search order.

If DFSORT is not installed as your primary z/OS sort product, use one of the following methods to enable DB2 to use DFSORT:

- Add the DFSORT SORTLPA library to the link pack area list; then add the DFSORT SICELINK library to the link list.
- Add the DFSORT SICELINK library to the link list; then add the DFSORT SORTLPA library to the link list.

  Important: If any non-IBM primary sort product is installed in the link list, install DFSORT in the link list, in system search order, after the non-IBM primary sort product

  Recommendation: If any non-IBM primary sort product is installed in the link pack area, add the DFSORT libraries to the link list.
- Add the DFSORT SICELINK library to the JOBLIB statement; then add the DFSORT SORTLPA library to the JOBLIB statement.
- Add the DFSORT SICELINK library to the STEPLIB DD statement; then add the DFSORT SORTLPA library to the STEPLIB DD statement.
- Add the DFSORT modules to a private library that is equivalent to one of the above configurations.

  Important: If your non-IBM primary sort product is run from a private library, you must use DFSORT in the same way.

If you install DFSORT in the link pack area or link library, you must install it after you install the non-IBM primary sort product.

DB2 uses only the SORT and MERGE functions in DFSORT. If you want to use DFSORT for any other uses outside of this limited DB2 support, you must separately order and license DFSORT.

More information about installing DFSORT is available in *DFSORT Installation and Customization*.

4. **IMS RESLIB**

Acceptable values:	valid data set name: see 97
Default:	none
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the IMS linkage editor library.

If you do not have IMS, you do not need to connect DB2 to IMS, and you can skip the phase 4 sample application jobs DSNTEJ4C and DSNTEJ4P.

5. **ISPF ISPLINK MODULE**

Acceptable values:	valid data set name: see 97
Default:	ISP.SISPLOAD
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the ISPF load module library.

6. **GDDM MACLIB**

Acceptable values:	valid data set name: see 97
Default:	GDDM.SADMSAM
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the GDDM macro library. This field and GDDM LOAD MODULES must both have a valid data set name or both be blank. The data set name that you specify in this field is included in the compile step SYSLIB DD concatenations of DSNHASM, DSNHC, DSNHICOB, and DSNHPLI. The installation CLIST only generates sample jobs DSNTEJ75 and DSNTEJ78 if you specify a GDDM MACLIB name.

7. **GDDM LOAD MODULES**

Acceptable values:	valid data set name: see 97
Default:	GDDM.SADMMOD
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name of the GDDM load module library. This field and GDDM MACLIB must both have a valid data set name or both be blank. The data set name that you specify in this field is included in the link-edit SYSLIB concatenations of DSNHASM, DSNHC, DSNHICOB, and DSNHPLI.

8. **CICS LOAD LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CICSTS.SDFHLOAD
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the CICS load module library. If you do not use CICS, put a blank in the CICS LOAD LIBRARY field.

9. **CICS MACRO LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CICSTS.SDFHMAC
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the CICS macro library.

10. **CICS COBOL LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CICSTS.SDFHCOB
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the CICS library that the COBOL programs are to use.

11. **CICS PL/I LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CICSTS.SDFHPLI
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the CICS library that PL/I programs are to use.

12. **CICS EXCI LIBRARY**

Acceptable values:	valid data set name: see 97
Default:	CICS.SDFHEXCI
Update:	cannot change during update
DSNZP*xxx*:	none

Specify the data set name for the CICS library that contains the CICS EXCI load modules.

Sizes panel 1: DSNTIPD

The entries on this panel establish the size of the DB2 catalog, directory, work file database, and log data sets.

The values that you supply on this panel are estimates that are used in calculating sizes for main storage and data sets. The values do not reduce any system limits and do not preclude an application or user from exceeding these estimates, within reasonable limits. For example, if you specify 500 databases, you could create 600. However, if you exceed the values by a large margin, you might encounter a shortage of main storage or use many secondary extents for some data sets. You can usually change the main storage values using the next panel. If you cannot change the values with the update process, see "The update process" on page 247 for the appropriate method.

The installation CLIST contains formulas that calculate the space for each catalog data set and the indexes that DB2 requires for each data set. Data that you enter on this panel is used in these formulas. Use integers; do not enter fractions. You can use K (as in 32 K) for multiples of 1024 bytes and M (as in 16 M) for multiples of 1 048 576 bytes in most fields, but do not exceed the maximum value that is accepted by the field. For example, for field 10, which has a maximum of 32 000, you can enter 31 K, meaning 31 744 bytes. Values of 32 K and above exceed the maximum acceptable value for this field.

| Many of the fields on this panel affect the values of the EDMPOOL, EDMSTATC,
| and EDMDBDC parameters in macro DSN6SPRM.

The defaults for most of the parameters on this panel correspond to the medium storage sizes for the site models that are shown in Table 8 on page 18. One exception is the value for the work file database. To obtain this value, see "Work file database storage requirements" on page 22. If you have a large site, increase the values according to your needs.

If you are migrating, DB2 UDB for z/OS Version 8 adopts your Version 7 DB2 catalog, directory, work file databases, BSDS, and active logs. Therefore, during migration, you cannot change any of the fields on this panel that affect those data sets.

Updating the parameters: You can alter the characteristics of the DB2 catalog, directory, work file databases, BSDS, and active and archive logs by using the methods described on 247. You cannot actually change the values of these parameters.

```
DSNTIPD         INSTALL DB2 - SIZES PANEL 1
===> _

Check numbers and reenter to change:

 1  DATABASES          ===> 200      In this subsystem
 2  TABLES             ===> 20       Per database (average)
 3  COLUMNS            ===> 10       Per table (average)
 4  VIEWS              ===> 3        Per table (average)
 5  TABLE SPACES       ===> 20       Per database (average)
 6  PLANS              ===> 200      In this subsystem
 7  PLAN STATEMENTS    ===> 30       SQL statements per plan (average)
 8  PACKAGES           ===> 300      In this subsystem
 9  PACKAGE STATEMENTS ===> 10       SQL statements per package (average)
10  PACKAGE LISTS      ===> 2        Package lists per plan (average)
11  EXECUTED STMTS     ===> 15       SQL statements executed (average)
12  TABLES IN STMT     ===> 2        Tables per SQL statement (average)
13  TEMP 4K SPACE      ===> 16       Size of 4K- work space (megabytes)
14  TEMP 4K DATA SETS  ===> 1        Number of data sets for 4K data
15  TEMP 32K SPACE     ===> 4        Size of 32K- work space (megabytes)
16  TEMP 32K DATA SETS ===> 1        Number of data sets for 32K data

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

Figure 15. Sizes panel 1: DSNTIPD

1. **DATABASES**

Acceptable values:	1 to 64000
Default:	200
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

Estimate the number of user databases in your subsystem.

2. **TABLES**

Acceptable values:	1 to 400
Default:	20
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

Estimate the average number of tables per database in your subsystem.

3. **COLUMNS**

Acceptable values:	1 to 750
Default:	10
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

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

4. **VIEWS**

Acceptable values:	1 to 200
Default:	3
Update:	see 247 (″The update process″)
DSNZP*xxx*:	none

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

5. **TABLE SPACES**

	Acceptable values:	1 to 400
#	Default:	20
	Update:	see 247 (″The update process″); cannot update during migration
	DSNZP*xxx*:	none

Estimate the average number of table spaces per database in your subsystem.

6. **PLANS**

Acceptable values:	1 to 32000
Default:	200
Update:	see 247 (″The update process″); cannot update during migration
DSNZP*xxx*:	none

Estimate the number of application plans in your subsystem. Each program requires a separate application plan.

7. **PLAN STATEMENTS**

Acceptable values:	1 to 32000
Default:	30
Update:	see 247 (″The update process″); cannot update during migration
DSNZP*xxx*:	none

Estimate the average number of SQL statements per application plan.

8. **PACKAGES**

Acceptable values:	1 to 256000
Default:	300
Update:	see 247 (″The update process″); cannot update during migration
DSNZP*xxx*:	none

Estimate the total number of packages in the system.

9. **PACKAGE STATEMENTS**

Acceptable values:	1 to 32000
Default:	10
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

Estimate the number of individual SQL statements per package.

10. **PACKAGE LISTS**

Acceptable values:	1 to 32000
Default:	2
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

Estimate the average number of packages in a package list per plan.

11. **EXECUTED STMTS**

Acceptable values:	1 to 32000
Default:	15
Update:	see 247 ("The update process"); not during migration
DSNZP*xxx*:	none

Estimate the average number of SQL statements that are executed per plan. The number of SQL statements that are executed can be less than the number written.

12. **TABLES IN STMT**

Acceptable values:	1 to 16
Default:	2
Update:	see 247 ("The update process"); cannot update during migration
DSNZP*xxx*:	none

Estimate the average number of tables that are used per SQL statement. Some SQL statements use more than one table (for example, those using joins, unions, or subselect clauses). Consider how often you expect to use such statements when choosing a value for this parameter.

13. **TEMP 4K SPACE**

Acceptable values:	1 to 2000000
Default:	16
Update:	see below; cannot update during migration
DSNZP*xxx*:	none

Specify in megabytes the total size of the 4-KB table spaces in the work file database.

The work file database is used as temporary space for SQL statements and triggers that require working storage. In particular, this includes statements that use:

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

Fields 13 and 15 allow you to create two kinds of table spaces within the work file database: one for 4-KB pages and one for 32-KB pages. Examples of names of the data sets for these table spaces are:

- DSNCAT.DSNDBD.*DSNDB07*.DSN4K*xx*.I0001.A001 for 4-KB pages
- DSNCAT.DSNDBD.*DSNDB07*.DSN32K*xx*.I0001.A001 for 32-KB pages

In these names, *DSNDB07* is the name of the work file database and *xx* is the number of the table space.

The space that is specified on this parameter is divided equally among each of the temporary 4-KB table spaces. For example, if you specify 16 for the TEMP 4K SPACE field and 4 for the TEMP 4K DATA SETS field, each 4-KB temporary data set is allocated 4 MB of space.

Because device characteristics differ, the actual amount of space may vary.

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

You can create additional temporary work file table spaces at any time except during migration. This capability lets you improve DB2 performance by reducing device contention among applications that require working storage. You can also concatenate temporary work file table spaces to support large temporary files. For information about creating additional temporary work file table spaces, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Updating: You can change the size of the data sets by deleting and redefining them when DB2 is not running or when the work file database is stopped. Job DSNTIJTM is a useful example. For information on job DSNTIJTM, see 272.

14. **TEMP 4K DATA SETS**

Acceptable values:	1 to 99
Default:	1
Update:	see field 13; not during migration
DSNZP*xxx*:	none

Estimate the number of temporary data sets for 4-KB pages.

15. **TEMP 32K SPACE**

Acceptable values:	0 to 2000000
Default:	4
Update:	see field 11 or see below; not during migration
DSNZP*xxx*:	none

Specify in megabytes the total size of the 32-KB table spaces in the work file database.

This field determines the total size of all 32-KB data sets. If you specify 0 for this option, job DSNTIJTM does not contain a statement to create a data set for 32-KB buffering. Be aware, however, that joins of smaller tables can produce rows that are longer than 4 KB. For this reason, you might want 32-KB data sets.

If you do not use 32-KB buffers, enter 0.

Because device characteristics differ, the actual amount of space will differ.

Updating: To update this field, create a 32-KB table space for the work file database. To create a 32-KB table space, follow these steps:

1. Stop DB2.
2. Run the CLIST in install mode. Be sure you enter positive values for fields TEMP 32K SPACE, TEMP 32K DATA SETS, and BP32K on the Buffer Pool Sizes Panel 3 (DSNTIP2).
3. Run job DSNTIJUZ to update the subsystem parameter values.
4. Start DB2. Do not access the 32-KB table space until you complete the next step. To control access, use the following command:
   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```
5. Edit the DSNTIJTM job as follows:
   - Delete everything except the DSNTIC procedure and steps DSNTTMP and DSNTIST.
   - Delete the control statements that define the 4-KB data set in DSNTTMP.
   - Delete the control statements that define the 4-KB table space in step DSNTIST.
   - Remove the step names that do not apply from the COND statements in the JCL.
   - Execute the job.

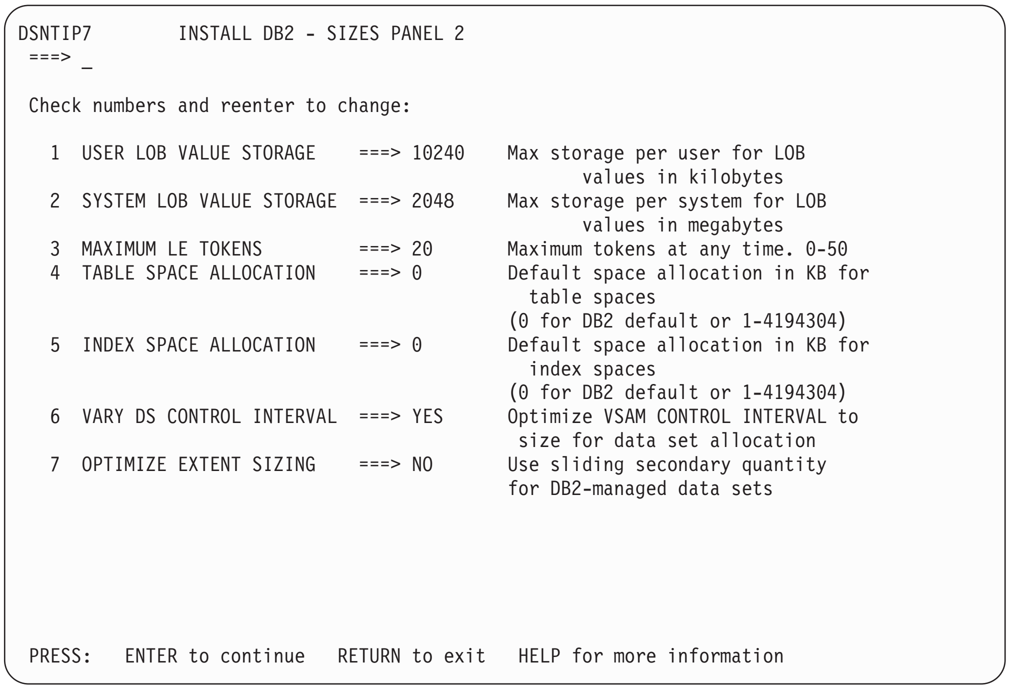
16. **TEMP 32K DATA SETS**

Acceptable values:	0 to 99
Default:	1
Update:	see fields 13 or 15; not during migration
DSNZP*xxx*:	none

Specify the number of temporary data sets for 32-KB pages.

Sizes panel 2: DSNTIP7

The entries on this panel establish limits for the amount of storage that can be used for storing large object (LOB) values.



```
DSNTIP7        INSTALL DB2 - SIZES PANEL 2
 ===> _

 Check numbers and reenter to change:

   1  USER LOB VALUE STORAGE    ===> 10240    Max storage per user for LOB
                                                     values in kilobytes
   2  SYSTEM LOB VALUE STORAGE  ===> 2048     Max storage per system for LOB
                                                     values in megabytes
   3  MAXIMUM LE TOKENS         ===> 20       Maximum tokens at any time. 0-50
   4  TABLE SPACE ALLOCATION    ===> 0        Default space allocation in KB for
                                                table spaces
                                              (0 for DB2 default or 1-4194304)
   5  INDEX SPACE ALLOCATION    ===> 0        Default space allocation in KB for
                                                index spaces
                                              (0 for DB2 default or 1-4194304)
   6  VARY DS CONTROL INTERVAL  ===> YES      Optimize VSAM CONTROL INTERVAL to
                                               size for data set allocation
   7  OPTIMIZE EXTENT SIZING    ===> NO       Use sliding secondary quantity
                                              for DB2-managed data sets





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

Figure 16. Sizes panel 2: DSNTIP7

1. **USER LOB VALUE STORAGE**

Acceptable values:	1 to 2097152
Default:	10240
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LOBVALA

Specify an upper limit for the amount of storage that each user can have for storing LOB values. The specified value indicates the numbers of kilobytes.

2. **SYSTEM LOB VALUE STORAGE**

Acceptable values:	1 to 51200
Default:	2048
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LOBVALS

Specify an upper limit for the amount of memory per system that can be used for storing LOB values. The specified value indicates the numbers of megabytes.

3. **MAXIMUM LE TOKENS**

Acceptable values:	0 to 50
Default:	20
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM LEMAX

Specify the maximum number of Language Environment tokens that are active at any time. If the value is 0, no tokens are available. A token is used each time one of the following functions is used:

- Trigonometry functions (SIN, SINH, ASIN, COS, COSH, ACOS, TAN, TANH, ATANH, ATAN, and ATAN2)
- DEGREES
- RADIANS
- RAND
- EXP
- POWER
- Log functions (LOG, and LOG10)
- UPPER
- LOWER
- TRANSLATE
- ROUND_TIMESTAMP
- TRUNC_TIMESTAMP
- LAST_DAY
- NEXT_DAY
- ADD_MONTHS

For details about these functions, see *DB2 SQL Reference*. DB2 might use a Language Environment token to perform conversion from one CCSID to another CCSID.

4. **TABLE SPACE ALLOCATION**

Acceptable values:	0 to 4194304
Default:	0
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TSQTY

Specify the amount of space in KB for primary and secondary space allocation for DB2-defined data sets for table spaces that are being created without the USING clause. A value of 0 indicates that DB2 is to use a default value of one cylinder for a non-LOB table space or ten cylinders for a LOB table space.

In a data sharing environment, this parameter has group scope.

5. **INDEX SPACE ALLOCATION**

Acceptable values:	0 to 4194304
Default:	0
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IXQTY

Specify the amount of space in KB for primary and secondary space allocation for DB2-defined data sets for index spaces that are being created without the USING clause. A value of 0 indicates that DB2 is to use a default allocation of one cylinder.

In a data sharing environment, this parameter has group scope.

6. **VARY DS CONTROL INTERVAL**

Acceptable values:	YES, NO
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DSVCI

Specify whether DB2-managed data sets created by CREATE TABLESPACE will have variable VSAM control intervals. If you specify YES, DB2 will create a DB2-managed data set for a table space. It will have a VSAM control interval that corresponds to the buffer pool used for the table space. A value of NO indicates that DB2-managed data sets are to be created with a fixed control interval of 4-KB, regardless of the buffer pool size.

This parameter is online-updatable. If you change this value from NO to YES, any pre-existing or migrated data sets remain in 4-KB control intervals until redefined. If you change this value from YES to NO, any pre-existing or migrated data sets in 8-KB, 16-KB, or 32-KB control intervals remain in those control intervals until redefined. You can explicitly redefine a data set. In addition, data sets are implicitly redefined by utilities such as LOAD REPLACE, REORG TABLESPACE, or RECOVER.

7. **OPTIMIZE EXTENT SIZING**

Acceptable values:	NO, YES
Default:	NO
Update:	option 8 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MGEXTSZ

Specify whether secondary extent allocations for DB2-managed data sets are to be sized according to a sliding scale that optimizes the likelihood of reaching the maximum data set size before secondary extents are exhausted. If you select NO, the default value, you will manage secondary extent allocations manually. If you select YES, DB2 will automatically optimize the secondary extent allocations.

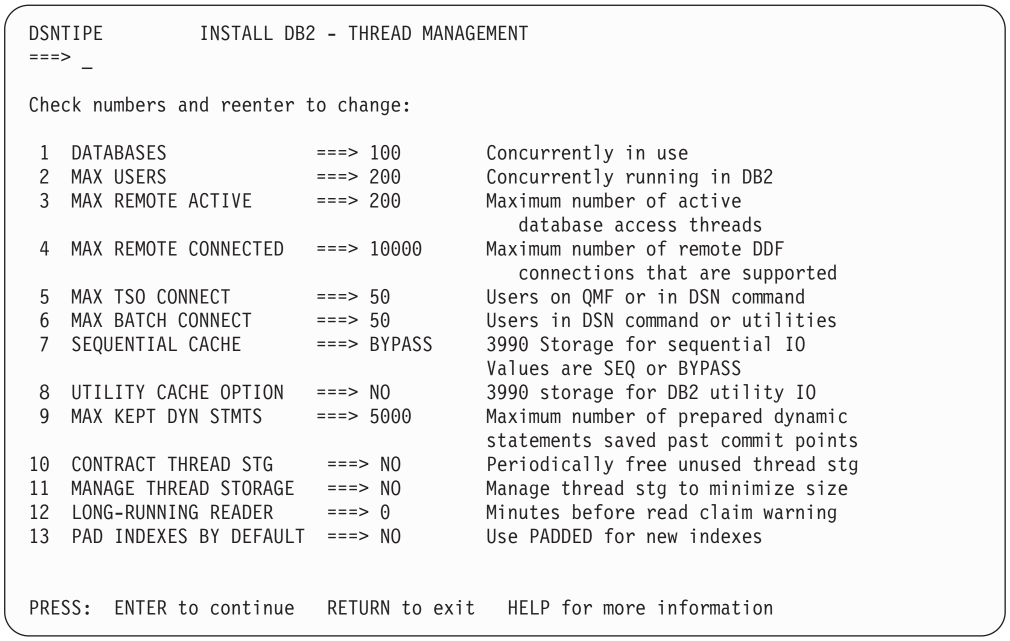
When the sliding scale is used, secondary extent allocations that are allocated earlier are smaller than those allocated later, until a maximum allocation is reached. The maximum allocation is 127 cylinders for data sets with a maximum size of 16 GB or less, and 559 cylinders for data sets with a maximum size of 32 GB or 64 GB.

In a data sharing environment, this parameter has group scope.

Thread management panel: DSNTIPE

The entries on this panel determine main storage sizes.

Updating the parameters: You can use UPDATE mode of the CLIST to update any value on this panel.



```
DSNTIPE         INSTALL DB2 - THREAD MANAGEMENT
===> _

Check numbers and reenter to change:

 1  DATABASES              ===> 100        Concurrently in use
 2  MAX USERS              ===> 200        Concurrently running in DB2
 3  MAX REMOTE ACTIVE      ===> 200        Maximum number of active
                                              database access threads
 4  MAX REMOTE CONNECTED   ===> 10000      Maximum number of remote DDF
                                              connections that are supported
 5  MAX TSO CONNECT        ===> 50         Users on QMF or in DSN command
 6  MAX BATCH CONNECT      ===> 50         Users in DSN command or utilities
 7  SEQUENTIAL CACHE       ===> BYPASS     3990 Storage for sequential IO
                                           Values are SEQ or BYPASS
 8  UTILITY CACHE OPTION   ===> NO         3990 storage for DB2 utility IO
 9  MAX KEPT DYN STMTS     ===> 5000       Maximum number of prepared dynamic
                                           statements saved past commit points
10  CONTRACT THREAD STG     ===> NO        Periodically free unused thread stg
11  MANAGE THREAD STORAGE   ===> NO        Manage thread stg to minimize size
12  LONG-RUNNING READER     ===> 0         Minutes before read claim warning
13  PAD INDEXES BY DEFAULT  ===> NO        Use PADDED for new indexes


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

Figure 17. Thread management panel: DSNTIPE

1. **DATABASES**

Acceptable values:	1 to 800
Default:	100
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the maximum number of databases that can be open at one time. The number is affected primarily by DSMAX on panel DSNTIPC, which specifies the number of open data sets. See "CLIST calculations panel 1: DSNTIPC" on page 238 for more information about DSMAX. For performance considerations, see Part 5 (Volume 2) of *DB2 Administration Guide*.

2. **MAX USERS**

Acceptable values:	1 to 2000
Default:	200
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CTHREAD

Specify the maximum number of allied threads (threads started at the local subsystem) that can be allocated concurrently. Count the following items as separate users:
- Each TSO user (whether running a DSN command or a DB2 request from DB2 QMF)
- Each batch job (whether running a DSN command or a DB2 utility)

- Each IMS region that can access DB2
- Each active CICS transaction that can access DB2
- Each utility (each utility uses one thread, plus one thread for each subtask)
- Each connection from users of CAF and RRSAF

The total number of threads accessing data that can be allocated concurrently is the sum of the MAX USERS value and the MAX REMOTE ACTIVE value. The maximum allowable value for this sum is 2000. When the number of users who are attempting to access DB2 exceeds the number you specify, excess plan allocation requests are queued. In most situations, the amount of real and virtual storage determines the maximum number of threads that DB2 can handle.

Due to parallelism, DB2 utilities each use a minimum of one thread, plus an additional thread for each subtask. Therefore, a single utility might use many threads. Specify a thread value accordingly to accommodate parallelism within utilities. Consider using a value that is higher than the default value or the value that you specified in a previous version of DB2.

3. **MAX REMOTE ACTIVE**

Acceptable values:	0 to 1999
Default:	200
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MAXDBAT

Specify the maximum number of database access threads (DBATs) that can be active concurrently.

The total number of threads accessing data concurrently is the sum of field 2, MAX USERS, and this field, MAX REMOTE ACTIVE. The maximum allowable value for this sum is 2000. If a request for a new connection to DB2 is received and MAX REMOTE ACTIVE has been reached, the resulting action depends on whether ACTIVE or INACTIVE is specified for option DDF THREADS on panel DSNTIPR.

If DDF THREADS is ...	Action taken is ...
ACTIVE	The allocation request is allowed but any further processing for the connection is queued waiting for an active database access thread to terminate.
INACTIVE	The allocation request is allowed and is processed when DB2 can assign an unused database access thread slot to the connection.

Setting MAX REMOTE ACTIVE to zero: You can use a 0 in this field to restrict DDF server activity on a member of a data sharing group. When this field is 0, expect the following results:

- DDF does not register the member's LU name with the VTAM generic LU name during DDF startup. This causes VTAM generic resource connections to be directed to DB2 members that specify a MAX REMOTE ACTIVE value of greater than 0.
- DDF does not register the member with WLM for member-specific Sysplex routing. This does not prevent the member from using WLM for enclave prioritization, but it prevents WLM from including this member in the Sysplex routing data that is sent to remote sites.

- DDF does not listen on the DRDA SQL port. This means TCP/IP SQL connections can be accepted only by members that specify MAX REMOTE ACTIVE greater than 0.
- DDF rejects requests for the Sysplex Routing TPN with the following sense code: SNA TPN not available.
- MAX REMOTE CONNECTED (CONDBAT) will be lowered to zero by default if it is not set explicitly.
- DB2 will still accept inbound Automatic Resynchronization requests.

4. **MAX REMOTE CONNECTED**

Acceptable values:	0 to 150000
Default:	10000
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CONDBAT

This field limits the total number of inbound DDF connections.

Specify the maximum number of concurrent remote connections. This value must be greater than or equal to MAX REMOTE ACTIVE. If MAX REMOTE ACTIVE is set to zero, MAX REMOTE CONNECTED will also be set to zero. When a request to allocate a new connection to DB2 is received, and MAX REMOTE CONNECTED has been reached or MAX REMOTE CONNECTED is zero, the connection request is rejected. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information.

5. **MAX TSO CONNECT**

Acceptable values:	1 to 2000
Default:	50
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDFORE

Specify the maximum number of users that can be identified to DB2 from TSO foreground at the same time. Count each of the following items as a separate user:
- Each TSO foreground user that is executing a DSN command.
- Each TSO foreground user that is connected to DB2 through the CAF or RRSAF. This can include DB2 QMF users who are running in TSO foreground or user-written CAF or RRSAF applications running in TSO foreground.

When the number of TSO users who are attempting to access DB2 exceeds the number you specify, excess connection requests are rejected. No DB2 subsystem parameter controls the maximum concurrent connections for IMS and CICS. You can control those limits by using IMS and CICS facilities. For the CICS attachment, the maximum number of connections to DB2 by using the resource control table (RCT) TYPE=INIT THRDMAX value.

6. **MAX BATCH CONNECT**

Acceptable values:	1 to 2000
Default:	50
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP IDBACK

Specify the maximum number of concurrent connections that are identified to DB2 from batch. Count each of the following items as a separate connection:

- Each DB2 utility.
- Each batch job that uses DB2 QMF.
- Each batch job that uses the DSN command processor.
- Each task that is connected to DB2 through the call attachment facility, which runs in batch. Among others, this can include:
  - Batch jobs that use DB2 QMF
  - TCP/IP FTP connections
- Each RRSAF connection that runs in batch.

Batch job requests to access DB2 that exceed this limit are rejected.

REBUILD INDEX processing uses DB2 connections and might cause message DSNU397I to be issued. If you receive message DSNU397I indicating the REBUILD INDEX utility is constrained, increase the number of concurrent connections.

7. **SEQUENTIAL CACHE**

Acceptable values:	BYPASS, SEQ
Default:	BYPASS
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SEQCACH

Specify whether to use the sequential mode to read cached data from a 3990 controller. If you accept the default, BYPASS, DB2 prefetch bypasses the cache. If you specify SEQ, DB2 prefetch uses sequential access for read activity. Many sites gain a performance benefit by specifying SEQCACH. See Part 5 (Volume 2) of *DB2 Administration Guide* for a discussion of these considerations.

Recommendation: Specify SEQCACH if you have current disk devices with good cache sizes, especially if the units are Enterprise Storage System (ESS) or RAMAC Virtual Array (RVA).

8. **UTILITY CACHE OPTION**

Acceptable values:	YES, NO
Default:	NO
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SEQPRES

Specify whether utilities that do a scan of a nonpartitioning index followed by an update of a subset of the pages in the index allow data to remain in cache longer when reading data. If you specify YES, these DB2 utility prefetch reads remain in cache longer, possibly improving performance of subsequent writes in the following cases for a table with very large nonpartitioned indexes:

- LOAD PART *integer* RESUME
- REORG TABLESPACE PART

The utility cache option is useful only with RAMAC disk attached to the 3990 Model 6.

If you specify NO, DB2 utilities use the 3990 cache the same way as any other application (as you specified in the SEQUENTIAL CACHE option).

9. **MAX KEPT DYN STMTS**

Acceptable values:	0 to 65535
Default:	5000
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAXKEEPD

Specify the total number of prepared, dynamic SQL statements that can be saved past a commit point by applications that run with the KEEPDYNAMIC(YES) bind option. This is a system-wide limit. This parameter does not limit the size of the dynamic cache itself.

When many applications that are bound with KEEPDYNAMIC(YES) run in a system that has the dynamic statement cache active, they can use a considerable amount of storage in the DBM1 address space. This parameter helps limit the amount of storage that these applications use by limiting the total number of prepared statements held by these applications past a commit point. If this limit is exceeded, DB2 honors the KEEPDYNAMIC(YES) behavior, but ″implicit″ prepares might be necessary to rebuild the executable version of some SQL statements when they are executed after a commit. For more information about the interaction of the KEEPDYNAMIC(YES) bind option and the dynamic statement cache, refer to Part 6 of *DB2 Application Programming and SQL Guide*.

When you enter 0, DB2 cannot keep the executable version of dynamic SQL statements past commit points. To retain the KEEPDYNAMIC(YES) behavior after a commit point, DB2 performs ″implicit″ prepares to rebuild the executable version of the dynamic SQL statements.

10. **CONTRACT THREAD STG**

Acceptable values:	YES, NO
Default:	NO
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CONTSTOR

Specify whether DB2 is to periodically ″contract″ each thread’s working storage area. Storage that a thread acquires is normally allocated to that thread until deallocation. If YES is specified for this parameter, DB2 examines threads at commit points and periodically returns to the operating system storage that is no longer in use.

Recommendation: For best performance, specify NO for this parameter. For subsystems that have many long-running threads and that are constrained on storage in the DBM1 address space, specifying YES can reduce the total amount of storage that is used in the DBM1 address space.

11. **MANAGE THREAD STORAGE**

Acceptable values:	YES, NO
Default:	NO
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MINSTOR

Specify whether DB2 is to use storage management algorithms that minimize the amount of working storage consumed by individual threads.

Recommendation: For best performance, specify NO for this parameter. For subsystems that have many long-running threads and are constrained on storage in the DBM1 address space, specifying YES can reduce the total amount of storage that is used in the DBM1 address space.

12. **LONG-RUNNING READER**

Acceptable values:	0 to 1439 minutes
Default:	0
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM LRDRTHLD

Specify the number of minutes that a read claim can be held by an agent before DB2 issues a warning message to report it as a long-running reader. If you specify a value of 0, DB2 will not report long-running readers.

13. **PAD INDEXES BY DEFAULT**

Acceptable values:	NO, YES
Default:	NO
Update:	option 9 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PADIX

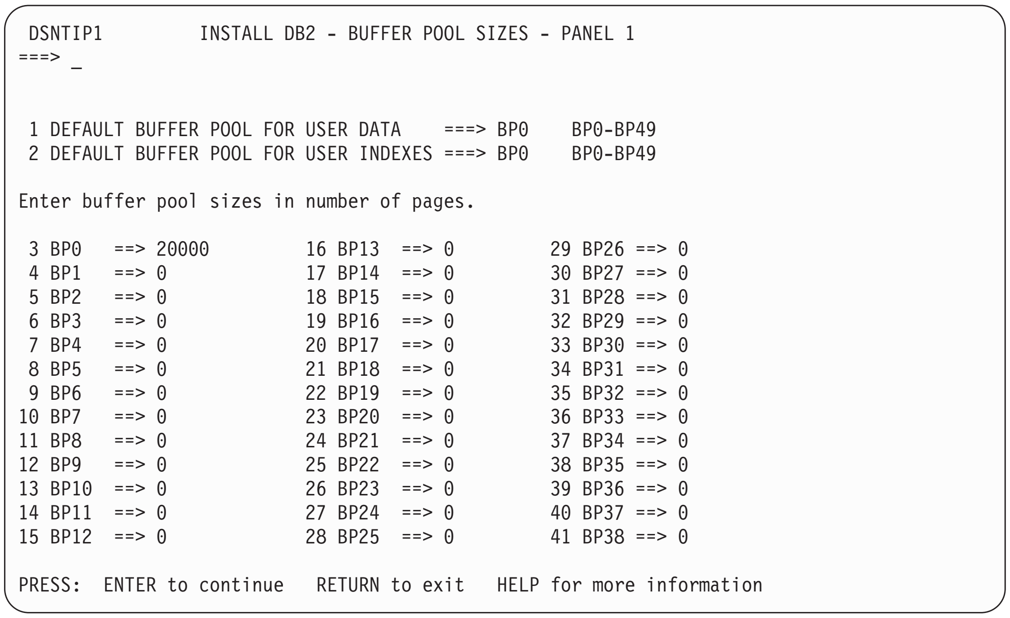
Specify whether new indexes should be padded by default. YES indicates that a new index will be padded unless the NOT PADDED option is specified on the CREATE INDEX statement. The default value, NO, indicates that a new index will not be padded unless the PADDED option is specified on the CREATE INDEX statement.

This parameter only affects indexes that have at least one varying-length column.

Buffer pool sizes panel 1: DSNTIP1

This is the first of two panels on which you can choose the size of your virtual buffer pools. For information about the other buffer pool sizes panel, see "Buffer pool sizes panel 2: DSNTIP2" on page 148. For a complete description of tuning buffer pools, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Updating the buffer pool sizes: You can change your buffer pool sizes online with the ALTER BUFFERPOOL command, but you cannot change these sizes by running the DSNTINST CLIST in update mode.



```
 DSNTIP1         INSTALL DB2 - BUFFER POOL SIZES - PANEL 1
===> _


 1 DEFAULT BUFFER POOL FOR USER DATA    ===> BP0    BP0-BP49
 2 DEFAULT BUFFER POOL FOR USER INDEXES ===> BP0    BP0-BP49

Enter buffer pool sizes in number of pages.

 3 BP0   ==> 20000        16 BP13  ==> 0        29 BP26 ==> 0
 4 BP1   ==> 0            17 BP14  ==> 0        30 BP27 ==> 0
 5 BP2   ==> 0            18 BP15  ==> 0        31 BP28 ==> 0
 6 BP3   ==> 0            19 BP16  ==> 0        32 BP29 ==> 0
 7 BP4   ==> 0            20 BP17  ==> 0        33 BP30 ==> 0
 8 BP5   ==> 0            21 BP18  ==> 0        34 BP31 ==> 0
 9 BP6   ==> 0            22 BP19  ==> 0        35 BP32 ==> 0
10 BP7   ==> 0            23 BP20  ==> 0        36 BP33 ==> 0
11 BP8   ==> 0            24 BP21  ==> 0        37 BP34 ==> 0
12 BP9   ==> 0            25 BP22  ==> 0        38 BP35 ==> 0
13 BP10  ==> 0            26 BP23  ==> 0        39 BP36 ==> 0
14 BP11  ==> 0            27 BP24  ==> 0        40 BP37 ==> 0
15 BP12  ==> 0            28 BP25  ==> 0        41 BP38 ==> 0

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

Figure 18. Buffer pool sizes panel 1: DSNTIP1

1. **DEFAULT BUFFER POOL FOR USER DATA**

Acceptable values:	Any 4-KB buffer pool names
Default:	BP0
Update:	use ALTER BUFFERPOOL to make changes dynamically
DSNZP*xxx*:	DSN6SYSP TBSBPOOL

Specify the default buffer pool to use for user table spaces. This must be a 4-KB buffer pool.

2. **DEFAULT BUFFER POOL FOR USER INDEXES**

Acceptable values:	Any 4-KB buffer pool names
Default:	BP0
Update:	see below
DSNZP*xxx*:	DSN6SYSP IDXBPOOL

Specify the default buffer pool to use for indexes on user data. This must be a 4–KB buffer pool.

3-41. **BUFFERPOOL**

Acceptable values:	for BP0, 2000 to 250 000 000; for BP1-BP39, 0 to 250 000 000
Default:	for BP0, 20000; for BP1 to BP38, 0
Update:	ALTER BUFFERPOOL command
DSNZP*xxx*:	none

Specify the total number of 4-KB buffers in the given virtual buffer pool (BP0-BP38).

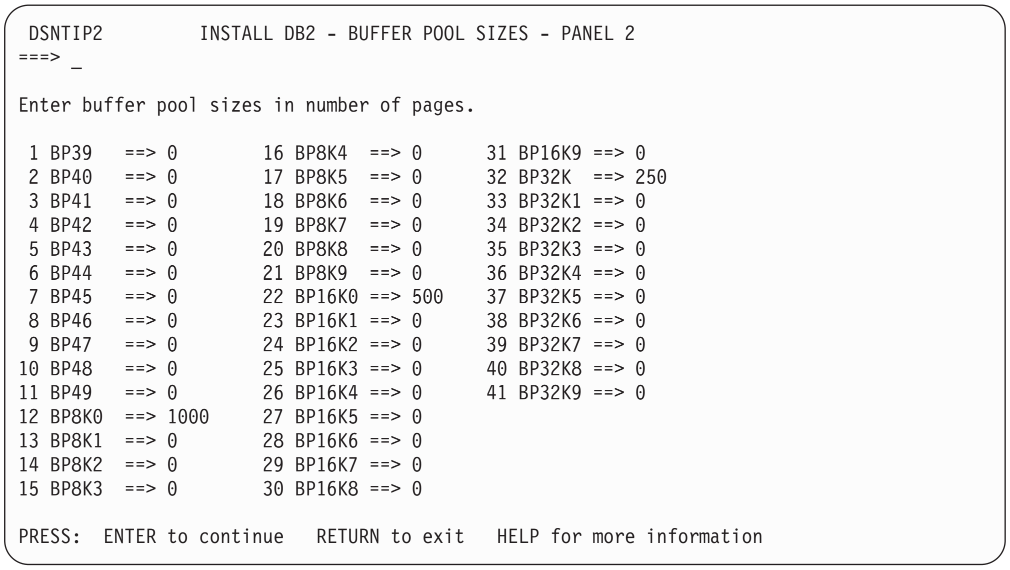
Important: The summation of the storage available in all buffer pools cannot exceed 1 TB. The amount of storage configured on the subsystem can further limit the buffer pool sizes. If you specify more storage than the total real storage available, performance will suffer.

Buffer pool sizes panel 2: DSNTIP2

This is the second of the two panels that lets you choose the size of your virtual buffer pools and whether you want your buffer pools all in the primary address space (database services address space) or in a z/OS data space. The first panel is described in “Buffer pool sizes panel 1: DSNTIP1” on page 146. For a complete description of tuning buffer pools, see Part 5 (Volume 2) of *DB2 Administration Guide*.

The total of all buffer pool sizes cannot exceed 1 TB.

Updating the buffer pool sizes: You can change your buffer pool sizes online with the ALTER BUFFERPOOL command, but you cannot change these sizes by running the DSNTINST CLIST in update mode.



```
 DSNTIP2         INSTALL DB2 - BUFFER POOL SIZES - PANEL 2
===> _

Enter buffer pool sizes in number of pages.

 1 BP39   ==> 0       16 BP8K4  ==> 0      31 BP16K9 ==> 0
 2 BP40   ==> 0       17 BP8K5  ==> 0      32 BP32K  ==> 250
 3 BP41   ==> 0       18 BP8K6  ==> 0      33 BP32K1 ==> 0
 4 BP42   ==> 0       19 BP8K7  ==> 0      34 BP32K2 ==> 0
 5 BP43   ==> 0       20 BP8K8  ==> 0      35 BP32K3 ==> 0
 6 BP44   ==> 0       21 BP8K9  ==> 0      36 BP32K4 ==> 0
 7 BP45   ==> 0       22 BP16K0 ==> 500    37 BP32K5 ==> 0
 8 BP46   ==> 0       23 BP16K1 ==> 0      38 BP32K6 ==> 0
 9 BP47   ==> 0       24 BP16K2 ==> 0      39 BP32K7 ==> 0
10 BP48   ==> 0       25 BP16K3 ==> 0      40 BP32K8 ==> 0
11 BP49   ==> 0       26 BP16K4 ==> 0      41 BP32K9 ==> 0
12 BP8K0  ==> 1000    27 BP16K5 ==> 0
13 BP8K1  ==> 0       28 BP16K6 ==> 0
14 BP8K2  ==> 0       29 BP16K7 ==> 0
15 BP8K3  ==> 0       30 BP16K8 ==> 0

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

Figure 19. Buffer pool sizes panel 2: DSNTIP2

1-41. **BUFFERPOOL**

Acceptable values:	for BP39-49, 0 to 250 000 000; for BP8K0 to BP8K9, 0 to 125 000 000; for BP16K0 to BP16K9, 0 to 62 500 000; for BP32K, 250 to 31 250 000; for BP32K1 to BP32K9, 0 to 31 250 000
Default:	for BP8K0, 1000; for BP16K0, 500; for BP32K, 250; for all others, 0
Update:	ALTER BUFFERPOOL command
DSNZP*xxx*:	none

Specify the total number or 4-KB buffers in the given virtual buffer pool (BP39 to BP32K9).

Restriction: The summation of the storage available in all buffer pools cannot exceed 1 TB. The amount of storage configured on the subsystem can further limit the buffer pool sizes.

Tracing parameters panel: DSNTIPN

The entries on this panel affect the audit, global, accounting, and monitor traces, as well as checkpoint frequency. For more information about these trace categories, see Part 5 (Volume 2) of *DB2 Administration Guide*.

```
DSNTIPN         INSTALL DB2 - TRACING PARAMETERS

===> _
Enter data below:

 1  AUDIT TRACE        ===> NO         Audit classes to start. NO,YES,list
 2  TRACE AUTO START   ===> NO         Global classes to start. YES,NO,list
 3  TRACE SIZE         ===> 64K        Trace table size in bytes. 4K-396K
 4  SMF ACCOUNTING     ===> 1          Accounting classes to start. NO,YES,list
 5  SMF STATISTICS     ===> YES        Statistics classes to start. NO,YES,list
 6  STATISTICS TIME    ===> 30         Time interval in minutes. 1-1440
 7  STATISTICS SYNC    ===> NO         Synchronization within the hour. NO,0-59
 8  DATASET STATS TIME ===> 5          Time interval in minutes. 1-1440
 9  MONITOR TRACE      ===> NO         Monitor classes to start. NO,YES,list
10  MONITOR SIZE       ===> 256K       Default monitor buffer size. 256K-16M
11  UNICODE IFCIDS     ===> NO         Include UNICODE data when writing IFCIDS
12  DDF/RRSAF ACCUM    ===> 10         Rollup accting for DDF/RRSAF. NO, 2-64K
13  AGGREGATION FIELDS ===> 0          Rollup accting aggregation fields

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

Figure 20. Tracing panel: DSNTIPN

1. **AUDIT TRACE**

Acceptable values:	YES, NO, list of classes, an asterisk (*)
Default:	NO
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP AUDITST

Specify whether to start the audit trace automatically when DB2 is started, and specify the classes for which to start it.

NO specifies no automatic start; if the audit trace is to be used, it must be started with the START TRACE command.

YES starts the trace for the default class (class 1) whenever DB2 is started.

To specify other classes for which trace must start automatically, list the numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 10 are defined by DB2. Enter an asterisk (*) to start audit trace for all classes.

For information about audit classes and the effect of the audit trace, see Part 3 (Volume 1) of *DB2 Administration Guide*.

2. **TRACE AUTO START**

Acceptable values:	YES, NO, list of classes, an asterisk (*)
Default:	NO
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TRACSTR

Specify whether to start the global trace automatically when DB2 is started, and specify the classes for which to start it.

NO specifies no automatic start; if the trace is to be used, it must be started with a special START TRACE command, as documented in *DB2 Diagnosis Guide and Reference*.

YES starts the global trace for the default classes (classes 1, 2, and 3) whenever DB2 is started, and it performs additional data consistency checks whenever a data or index is modified.

To start specific classes, enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 9 are defined by DB2. Enter an asterisk (*) to start global trace for all classes.

The global trace is used to diagnose problems in DB2. Users with production systems that require high performance might consider turning off global trace. However, be aware that turning off global trace presents a serviceability exposure. In the event of a system failure, IBM Software Support might request that you turn on global trace and attempt to re-create the problem. For information about the global trace facility, see *DB2 Diagnosis Guide and Reference*.

3. **TRACE SIZE**

Acceptable values:	4K to 396K
Default:	64K
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP TRACTBL

Specify the size, in bytes, of the RES trace table. This table is the default destination for the global trace records in DB2. Most trace records require 32-byte entries; events with more than three data items require 64-byte entries.

You can use the abbreviation K for multiples of 1024 bytes. The actual value is rounded up to a multiple of 4. If you use 50K, for example, the actual table size is 52 KB.

In the subsystem parameter, use a multiple of 4. For example, to get a 64-KB table, code TRACTBL=16.

4. **SMF ACCOUNTING**

Acceptable values:	YES, NO, list of classes, an asterisk (*)
Default:	1
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SMFACCT

Specify whether DB2 is to send accounting data to SMF automatically when DB2 is started. This field also specifies what classes are sent.

NO specifies no automatic start. YES starts the trace for the default class (class 1). You might also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write the records.

To start specific classes, enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5, 7, and 8 are defined by DB2. To start all classes, enter an asterisk (*). For information about using the trace, see "Installation step 7: Record DB2 data to SMF (optional)" on page 263. For information about accounting classes, see Part 5 (Volume 2) of *DB2 Administration Guide*.

5. **SMF STATISTICS**

Acceptable values:	YES, NO, list of classes, an asterisk (*)
Default:	YES
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SMFSTAT

Specify whether DB2 is to send statistical data to SMF automatically when DB2 is started. This field also specifies what classes are sent.

NO specifies no automatic start. YES starts the trace for the default classes (classes 1, 3, 4, 5, and 6). You might also need to update the SMFPRM*xx* member of SYS1.PARMLIB to permit SMF to write the records.

To start specific classes, enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5 are defined by DB2. To start all classes, enter an asterisk (*). For information about using the trace, see "Installation step 7: Record DB2 data to SMF (optional)" on page 263. For information about statistics classes, see Part 5 (Volume 2) of *DB2 Administration Guide*.

6. **STATISTICS TIME**

Acceptable values:	1 to 1440
Default:	30
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STATIME

Specify the time interval, in minutes, between statistics collections. Statistics records are written approximately at the end of this interval.

7. **STATISTICS SYNC**

Acceptable values:	NO, 0 to 59
Default:	NO
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP SYNCVAL

Specify whether DB2 statistics recording is to be synchronized with some part of the hour. You can specify that the DB2 statistics recording interval is to be synchronized with the beginning of the hour (0 minutes past the hour) or with any number of minutes past the hour up to 59. If NO is specified, no synchronization is done. NO is the default. This parameter has no effect if STATIME is greater than 60.

Example: You want the DB2 statistics recording interval to have a length of 15 minutes and to be synchronized with 15 minutes past the hour. Thus, DB2 statistics are recorded at 15, 30, 45, and 60 minutes past the hour. To establish this interval, you would specify the following:

```
STATIME=15
SYNCVAL=15
```

8. **DATASET STATS TIME**

Acceptable values:	1 to 1440
Default:	5 (minutes)
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DSSTIME

The value in this field specifies the time interval, in minutes, between the resetting of data set statistics for online performance monitors. Online performance monitors can request DB2 data set statistics for the current interval with an IFI READS request for IFCID 0199.

9. **MONITOR TRACE**

Acceptable values:	YES, NO, list of classes, an asterisk (*)
Default:	NO
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MON

Specify whether to start the monitor trace automatically when DB2 is started. This field also specifies what classes are sent.

NO specifies no automatic start. YES starts the trace for the default classes (class 1) whenever DB2 is started. To start the trace automatically for other classes, enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 8 are defined by DB2. To start all classes, enter an asterisk (*). For information about the monitor trace, see Appendix F (Volume 2) of *DB2 Administration Guide*.

10. **MONITOR SIZE**

Acceptable values:	256K to 16M
Default:	256K
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP MONSIZE

Specify the default buffer size for monitor trace when sending data to monitor destinations. You can enter the value in bytes (for example, 8192) or use the abbreviation K for kilobytes (for example, 256K) or M for megabytes (for example, 4M).

11. **UNICODE IFCIDS**

Acceptable values:	NO, YES
Default:	NO
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP UIFCIDS

Specify whether output from IFC records should include Unicode information. Only a subset of the character fields (identified in the IFCID record definition by a

%U in the comment area to the right of the field declaration in the DSNDQWxx copy files) are encoded in Unicode. The remaining fields maintain the same encoding of previous releases.

12. **DDF/RRSAF ACCUM**

Acceptable values:	NO, 2 to 65535
Default:	10
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ACCUMACC

Specify whether DB2 accounting data should be accumulated by the user for DDF and RRSAF threads. If NO is specified, DB2 writes an accounting record when a DDF thread is made inactive or when signon occurs for an RRSAF thread. If a value between 2 and 65535 is specified, DB2 writes an accounting record every *n* occurrences of the user on the thread, where *n* is the number that is specified for this parameter. A user is identified by the concatenation of the following three values:

- User ID
- User transaction name
- User workstation name

These values can be set by DDF threads via Server Connect and Set Client calls, and by RRSAF threads via the RRSAF SIGN, AUTH SIGNON, and CONTEXT SIGNON functions.

If a value between 2 and 65535 is specified, an accounting record might be written prior to the *n*th occurrence of the user in the following cases:

- An internal storage threshold is reached for the accounting rollup blocks.
- *accounting-interval* = 'COMMIT' was specified on the RRSAF signon call.
- When the thread deallocates, the accumulated accounting data for all users on this thread is written (one record per user).
- If the subsystem parameter is dynamically altered to deactivate accounting accumulation, the next end-UR (for a DDF thread) or signon (for an RRSAF thread) causes DB2 to write the accumulated accounting data for all users on this thread (one record per user).

13. **AGGREGATION FIELDS**

Acceptable values:	0 to 10
Default:	0
Update:	option 12 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYS ACCUMUID

Specify the aggregation fields to be used for DDF and RRSAF accounting rollup.
The values are defined in Table 36 on page 154.

Table 36. Values for the aggregation fields that are used for DDF and RRSAF accounting rollup

Value	Rollup criteria	String of X′00′ or string of X′40′ considered for rollup?
0	End user ID, transaction name, and workstation name	Yes[1]
1	End user ID	No
2	End user transaction name	No
3	End user workstation name	No
4	End user ID and transaction name	Yes[1]
5	End user ID and workstation name	Yes[1]
6	End user transaction name and workstation name	Yes[1]
7	End user ID, transaction name, and workstation name	No
8	End user ID and transaction name	No
9	End user ID and workstation name	No
10	End user transaction name and workstation name	No

Note:
[1] At least one value in the set of criteria must have a value other than a string of X′00′s or a string of X′40′s.

For example, if a thread has the end user ID set to a string of X′00′ and the workstation name set to myws, the thread qualifies for rollup if ACCUMUID is set to 5, but not if ACCUMUID is set to 9.

Assume ACCUMUID is set to 5. The threads in Table 37 all qualify for rollup.

Table 37. Values for end user ID and workstation name

End user ID	Workstation name
myuser	myws
myuser	A string of X′40′
A string of X′40′	myws
myuser	A string of X′00′
A string of X′00′	myws

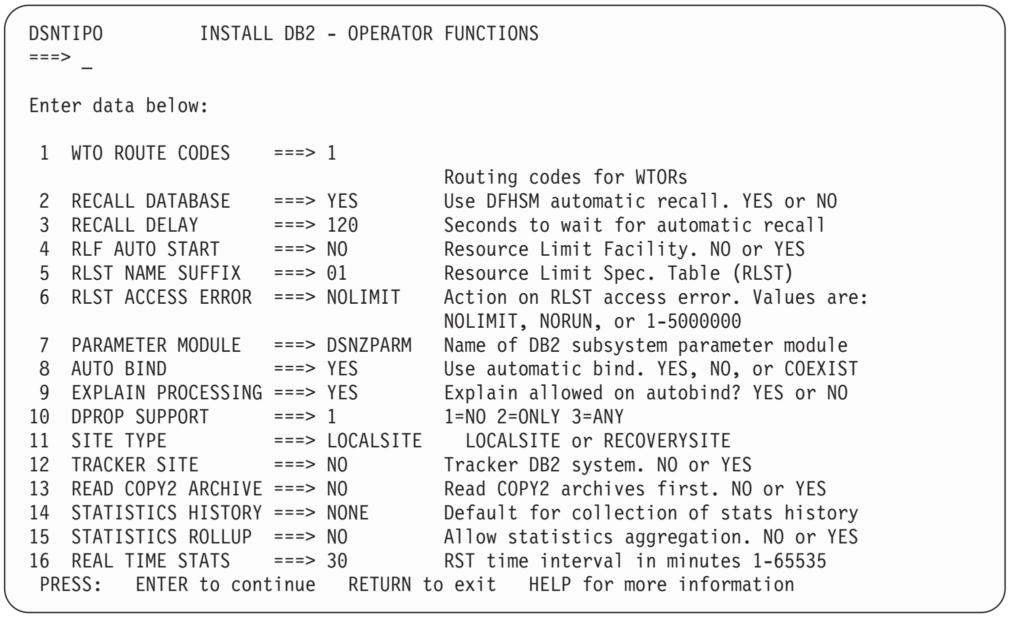
The thread with end user ID set to myuser and workstation name set to a string of X′40′ and the thread with end user ID set to myuser and workstation name set to a string or X′00′ are included in the same rollup record. The thread with end user ID set to a string of X′40′ and workstation name set to myws and the thread with end user ID set to a string of X′00′ and workstation name set to myws are included in the same rollup record.

DB2 writes individual accounting threads for threads that do not meet the criteria for rollup.

The value of ACCUMUID is ignored if DDF or RRSAF accounting is not being used.

Operator functions panel: DSNTIPO

The entries on this panel affect various operator functions, such as write-to-operator route codes, automatic recall, and the maximum amount of CPU time that is to be allocated for a dynamic SQL statement.



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

Enter data below:

 1  WTO ROUTE CODES    ===> 1
                                           Routing codes for WTORs
 2  RECALL DATABASE    ===> YES        Use DFHSM automatic recall. YES or NO
 3  RECALL DELAY       ===> 120        Seconds to wait for automatic recall
 4  RLF AUTO START     ===> NO         Resource Limit Facility. NO or YES
 5  RLST NAME SUFFIX   ===> 01         Resource Limit Spec. Table (RLST)
 6  RLST ACCESS ERROR  ===> NOLIMIT    Action on RLST access error. Values are:
                                       NOLIMIT, NORUN, or 1-5000000
 7  PARAMETER MODULE   ===> DSNZPARM   Name of DB2 subsystem parameter module
 8  AUTO BIND          ===> YES        Use automatic bind. YES, NO, or COEXIST
 9  EXPLAIN PROCESSING ===> YES        Explain allowed on autobind? YES or NO
10  DPROP SUPPORT      ===> 1          1=NO 2=ONLY 3=ANY
11  SITE TYPE          ===> LOCALSITE    LOCALSITE or RECOVERYSITE
12  TRACKER SITE       ===> NO         Tracker DB2 system. NO or YES
13  READ COPY2 ARCHIVE ===> NO         Read COPY2 archives first. NO or YES
14  STATISTICS HISTORY ===> NONE       Default for collection of stats history
15  STATISTICS ROLLUP  ===> NO         Allow statistics aggregation. NO or YES
16  REAL TIME STATS    ===> 30         RST time interval in minutes 1-65535
 PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 21. Operator functions panel: DSNTIPO

1. **WTO ROUTE CODES**

Acceptable values:	see below
Default:	1
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP ROUTCDE

Specify the z/OS console routing codes assigned to messages that are not solicited from a specific console. You can specify from 1 to 16 route codes. You must use at least one code. Separate codes in a list by commas only, with no blanks; for example: 1,3,5,7,9,10,11. For more information about routing codes, refer to *z/OS MVS Routing and Descriptor Codes*.

2. **RECALL DATABASE**

Acceptable values:	YES, NO
Default:	YES
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RECALL

Specify whether DFSMShsm automatic recall is to be performed for DB2 databases. NO indicates that a DB2 table space that has been migrated is considered to be an unavailable resource. It must be recalled explicitly before DB2 can use it. YES indicates that DFSMShsm is invoked to recall it automatically.

3. **RECALL DELAY**

Acceptable values:	0 to 32767
Default:	120
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RECALLD

Specify the maximum length of time, in seconds, that a program can delay for a DFSMShsm recall. If the recall is not completed within the specified number of seconds, the program receives an error message indicating that the set is unavailable, but that a recall was initiated. If you use 0 and RECALL DATABASE (field 2) is YES, the recall is performed asynchronously. This field is ignored if the RECALL DATABASE field is NO.

The RECALL DELAY option is not used when running a DB2 utility against a DB2-migrated data set.

4. **RLF AUTO START**

Acceptable values:	YES, NO
Default:	NO
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLF

Specify whether the resource limit facility (governor) is to automatically start each time DB2 is started. For information about using the governor, see Part 5 (Volume 2) of *DB2 Administration Guide*.

5. **RLST NAME SUFFIX**

Acceptable values:	any 2 alphanumeric characters; national characters are not allowed
Default:	01
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFTBL

Specify the suffix of the default resource limit specification table (RLST). The default RLST is used when the resource limit facility (governor) is automatically started or when you start the governor without specifying a suffix.

6. **RLST ACCESS ERROR**

Acceptable values:	NOLIMIT, NORUN, 1 to 5 000 000
Default:	NOLIMIT
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFERR

Specify what action DB2 is to take if the governor encounters a condition that prevents it from accessing the resource limit specification table or if it cannot find a row in the table that applies to the authorization ID, the plan or package name, and the logical unit of work name of the query user.

- NOLIMIT allows all dynamic SQL statements to run without limit.

- NORUN terminates all dynamic SQL statements immediately with an SQL error code.
- A number from 1 to 5 000 000 is the default limit; if the limit is exceeded, the SQL statement is terminated. For guidelines in choosing the default limit, see Part 5 (Volume 2) of *DB2 Administration Guide*.

7. **PARAMETER MODULE**

Acceptable values:	1 to 8 characters
Default:	DSNZPARM
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the member name of the load module for DB2 subsystem parameters. The module resides in library *prefix*.SDSNEXIT. To avoid conflict with members of *prefix*.SDSNLOAD, use DSNZ*xxx*, where *xxx* is any set of three alphanumeric characters. DB2 puts this name in the startup JCL procedure in SYS1.PROCLIB, but you can override this value by using the START DB2 command.

8. **AUTO BIND**

Acceptable values:	YES, NO, COEXIST
Default:	YES
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ABIND

Specify whether plans or packages are to be automatically rebound (or autobound) at run time in certain situations. Automatic rebinds can improve availability and administration of plans and packages because you do not need to explicitly rebind some plans and packages. However, automatic rebinds can also hinder performance because they require access to the DB2 directory and catalog.

Specify the AUTO BIND value as YES, NO or COEXIST:

YES Specifying YES allows automatic rebind operations to be performed at execution time when a plan or package:

- Is not valid; the SYSPLAN or SYSPACKAGE column VALID contains ′N′. (See Part 4 of *DB2 Application Programming and SQL Guide* for information about how a plan or package becomes invalid.)
- Was last bound in DB2 Version 8 but is now running on a previous version of DB2.

  This autobind is performed on the previous version of DB2 because that is where the user is attempting to run the plan or package. This autobind is performed only for the first attempt to run the plan or package on DB2 Version 7.
- Was last autobound on DB2 Version 7 but is now running again on DB2 Version 8.

  This ″Version 7-to-Version 8″ autobind is performed on DB2 Version 8 because that is where the user is attempting to run the plan or package. This autobind is performed only for the first attempt to run the plan or package on Version 8 after an autobind on Version 7. If the plan or package is later run again on DB2 Version 7, the autobind process starts again as previously described.

NO Specifying NO for this field prevents DB2 from performing any automatic rebind operations under any circumstances. This means that when the following conditions are true, you must explicitly rebind the plan or package before it can be run again:

- You want to run a plan or package that is not valid; the SYSPLAN or SYSPACKAGE column VALID contains 'N'. (See Part 4 of *DB2 Application Programming and SQL Guide* for more information about how a plan or package becomes invalid.)
- You are attempting to run a plan or package on a previous version that was last bound (explicitly or automatically) in DB2 Version 8.
- You are attempting to run a plan or package on DB2 Version 8 for which DB2 last performed an autobind. If AUTO BIND = NO has always been in effect on Version 7, then the autobind might not have been done previously on Version 7.

If you attempt to run any plan or package on DB2 Version 8 in one of the situations described above, you receive SQLCODE -908 SQLSTATE 23510.

COEXIST Specifying COEXIST allows automatic rebind operations to be performed in a DB2 data sharing coexistence environment only when the plan or package:

- Is marked not valid; the SYSPLAN or SYSPACKAGE column VALID contains 'N'. (See Part 4 of *DB2 Application Programming and SQL Guide* for more information about how a plan or package becomes invalid), or
- Was last bound on DB2 Version 8 and is now running on DB2 Version 7.

  For this case, DB2 performs an autobind on Version 7 for the plan or package before running it there. This autobind is performed only for the first attempt to run the plan or package on DB2 Version 7.

An automatic rebind operation is not performed on DB2 Version 8 in a data sharing coexistence environment for any Version 8 plan or package for which DB2 last performed an autobind on the previous version and that plan or package is now run again on Version 8. The plan or package will run on Version 8 as a Version 7-bound plan or package. However, no Version 8-only features such as optimization enhancements, improved access paths, or index usage enhancements will be used because the plan or package was not autobound on DB2 Version 8.

After all members of a data sharing group have been migrated to the same release level, DB2 interprets a value of COEXIST as YES, which allows all types of autobinds to occur.

The value COEXIST is relevant only for DB2 subsystems in data sharing mode. If COEXIST is specified in a non-data-sharing environment, DB2 ignores the value and uses the default value of YES when determining whether an autobind can be done. YES allows a Version 7-to-Version 8 autobind on Version 8 after a previous Version 8 to Version 7 autobind completes successfully on Version 7.

If you have a DB2 data sharing group in which some of the DB2 members are at different DB2 release levels, the rate of automatic rebinds might increase. Subsequently, a degradation in run-time performance for a plan or package might occur if one of the following events occurs:

- The plan or package was last bound on DB2 Version 8 and is now run on DB2 Version 7 (a Version 8-to-Version 7 autobind occurs on Version 7), or
- DB2 last performed a Version 8-to-Version 7 autobind for that plan or package on DB2 Version 7, and the plan or package is now run on Version 8 (a Version 7-to-Version 8 autobind occurs on Version 8).

To reduce the rate of automatic rebinds in this type of data sharing environment, consider specifying COEXIST for AUTO BIND.

9. **EXPLAIN PROCESSING**

Acceptable values:	YES, NO
Default:	YES
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ABEXP

Specify whether you want EXPLAIN processing to occur during automatic rebind.

YES specifies that you want EXPLAIN processing to occur during automatic rebind of a plan or package when the bind option EXPLAIN(YES) is specified. If the PLAN_TABLE does not exist, automatic rebind continues, but generates no EXPLAIN output. If you specify YES in this field, but you have a plan or package with the bind option EXPLAIN(NO), EXPLAIN processing does not occur during automatic rebind.

NO specifies that you do not want EXPLAIN processing to occur during the automatic rebind of a plan or package.

10. **DPROP SUPPORT**

Acceptable values:	1, 2, 3
Default:	1
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDPROP, DSN6SPRM CHGDC

Specify whether you want to use IMS DataPropagator™ to propagate SQL changes to tables defined with DATA CAPTURE CHANGES.

A value of 1 specifies that you do not intend to propagate changes.

A value of 2 specifies that you intend to use IMS DataPropagator to propagate SQL changes, and that changes made to tables defined with DATA CAPTURE CHANGES are allowed **only** when all of the following conditions are met:

- Monitor trace class 6 is active.
- IMS DataPropagator is installed.
- The DB2 application is running in an IMS environment.

If you choose 2 for DPROP SUPPORT and monitor trace class 6 is not active, IMS DataPropagator is not installed, or the IMS DataPropagator application is not running in an IMS environment, then no changes to the DB2 table are permitted.

A value of 3 specifies that data propagation occurs when all of the following conditions are met:
- Monitor trace class 6 is active.
- IMS DataPropagator is installed.
- The DB2 application is running in an IMS environment.

The ANY option for IMS DataPropagator Support is intended for subsystems that need to propagate some data with IMS DataPropagator and need to propagate some data with a different program.

If you choose 3 for IMS DataPropagator support, an application that is not running in an IMS environment can update DB2 tables that are defined with DATA CAPTURE CHANGES. However, these changes are not propagated to IMS. You might have tables in your DB2-IMS environment that you want to be updated only by DB2 applications. You can protect these tables by using any of the following methods:
- Using the ENABLE parameter on BIND to specify a specific attachment facility through which updates to data propagation tables can be made.
- Defining a validation procedure for data propagation tables to allow only certain plans to update those tables.
- Using a group authorization ID to allow update authority for data propagation tables to a group of authorization IDs that can run only in the IMS environment.

For more information about IMS DataPropagator, see *IMS DataPropagator: An Introduction* and http://www.ibm.com/software/data/db2imstools/imstools/imsdprop.html.

11. **SITE TYPE**

Acceptable values:	LOCALSITE, RECOVERYSITE
Default:	LOCALSITE
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SITETYP

Specify whether the current system is at a local site or a recovery site. LOCALSITE is defined as the site where the multiple image copies are made and are operational. RECOVERYSITE is the site that is named as an alternative for recovery purposes.

The RECOVER utility looks at this value to determine what site the current system is on and recovers everything from the copies of data that is registered at that site. The RECOVER and MERGECOPY utilities look at this value to determine whether COPYDDN or RECOVERDDN is allowed with NEWCOPY NO.

12. **TRACKER SITE**

Acceptable values:	NO or YES
Default:	NO
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM TRKRSITE

The value in this field indicates whether the subsystem that is being installed is to be used as a remote tracker site for another DB2 subsystem in case of a disaster. See Part 4 (Volume 1) of *DB2 Administration Guide* for more information about disaster recovery by using a tracker site.

13. **READ COPY2 ARCHIVE**

Acceptable values:	NO or YES
Default:	NO
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP ARC2FRST

Indicate whether COPY2 archives should be read first when the DB2 subsystem is started.

14. **STATISTICS HISTORY**

Acceptable values:	SPACE, NONE, ALL, ACCESSPATH
Default:	NONE
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATHIST

Specify which inserts and updates are recorded in catalog history tables.

SPACE
When the SPACE option is specified, all inserts and updates that DB2 makes to space-related catalog statistics are recorded.

ACCESSPATH
When the ACCESSPATH option is specified, all inserts and updates DB2 makes to ACCESSPATH-related catalog statistics are recorded.

ALL When the ALL option is specified, all inserts and updates that DB2 makes in the catalog are recorded.

NONE
When the NONE option is specified, changes that DB2 makes in the catalog are not recorded. This is the default for the HISTORY subsystem parameter.

15. **STATISTICS ROLLUP**

Acceptable values:	YES or NO
Default:	NO
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATROLL

Specify whether the RUNSTATS utility is to aggregate the partition-level statistics, even though some parts may not contain data. For DB2 subsystems that have large partitioned table spaces and indexes, the recommended parameter is YES. This enables the aggregation of partition-level statistics and helps the optimizer to choose a better access path.

16. **REAL TIME STATS**

Acceptable values:	1 to 1440

Default:	30
Update:	option 13 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STATSINT

Specify the time, in minutes, that DB2 waits before attempting to write out page set statistics to the real-time statistics tables.

Application programming defaults panel 1: DSNTIPF

The entries on this panel and the next set application programming defaults. The values that you specify on these panel are used as default values by the program preparation panels, the program preparation CLIST (DSNH), and the precompiler. They can also be used as defaults by other programs, such as DB2 QMF.

Migrating or updating the parameters: If you alter parameter values for which a change during migration or update is "not recommended," this change can invalidate the syntax of existing SQL statements or affect the way that application programs run. Update is allowed, but must be handled with caution.

Most of the values that are set here and on the next panel are contained in load module DSNHDECP, in library *prefix*.SDSNEXIT, which can be loaded and accessed by application programs. When modifying DSNHDECP, do so only by changing and running the installation CLIST.

Important: You should always use the CLIST to modify installation parameters. Do not modify the data in DSNHDECP. If you modify any installation parameters by changing job DSNTIJUZ directly, these values are not recorded for later updates, new installations, or migrations. In addition, these values are not checked for validity. If you do not use the CLIST to modify these parameters, DB2 may not start.

Many of the fields on this panel involve the selection of coded character set identifiers (CCSIDs). Here is some information that can help you choose values for these fields:

- If you choose YES for the MIXED DATA field, you must specify a mixed data CCSID from Table 122 on page 517 or Table 123 on page 517. An error occurs if you do not specify a CCSID or if the CCSID you specify is not listed in the table.
- If you specify an incorrect CCSID, data can become corrupted. For example, assume that the coded character set used at your site is 37, but you specify 500 as the system CCSID. If DB2 receives data with a CCSID of 500, the data can become corrupted because character conversion does not occur. Conversely, if DB2 receives data with a CCSID other than 500 and a conversion is made from that CCSID **to** 500, the data can become corrupted because character conversion occurs.
  Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.
- If you need to convert to a CCSID that supports the euro symbol, you can correct it by altering the CCSID field for your default encoding scheme. See "Converting to the euro symbol" on page 518 for the detailed steps.
- During code conversion, DB2 first looks in the SYSSTRINGS table to see if a conversion is defined. If DB2 finds a conversion, it is used. If DB2 does not find a conversion, it uses z/OS Unicode Services. In some cases, z/OS Unicode Services is used instead of the value in SYSSTRINGSS. See *OS/390 C/C++ Programming Guide* for additional conversions that might be supported. If the conversion is not available, an error occurs.
- Converting statements to Unicode for parsing depends on having the correct input CCSID specified. The system CCSIDs must be set up correctly at installation time.

  During connect processing, a requester and server provide default CCSIDs for character data sent on the connection.

The DB2 requester uses the application encoding scheme for its default CCSIDs. If an application provides character data that is not in the CCSID that the application encoding scheme identifies, DB2 overrides the default CCSID by tagging each field with the actual CCSID prior to sending to the server. For applications that use the Unicode encoding scheme, the DB2 requester overrides the application encoding scheme CCSIDs with the system EBCDIC CCSIDs and converts the Unicode data to EBCDIC if DB2 determines that the server does not support Unicode character data. If the server cannot accept character data in these CCSIDs, connect fails with a -332 SQLCODE.

The DB2 server uses the system default encoding scheme to determine the default CCSID values for character data that is to be returned to the requester. For servers using the encoding scheme of Unicode, the DB2 server overrides the Unicode encoding scheme CCSIDs with the system EBCDIC CCSIDs and converts the Unicode data to EBCDIC if DB2 determines that the requester does not support Unicode character data. If the requester cannot accept character data in these CCSIDs, connect fails with a -332 SQLCODE.

For more information about the fields on this panel, see Part 4 of *DB2 Application Programming and SQL Guide*.

```
DSNTIPF         INSTALL DB2 - APPLICATION PROGRAMMING DEFAULTS PANEL 1
===> _

Enter data below:

 1  LANGUAGE DEFAULT     ===> IBMCOB   ASM,C,CPP,IBMCOB,FORTRAN,PLI
 2  DECIMAL POINT IS     ===> .      . or ,
 3  STRING DELIMITER     ===> DEFAULT  DEFAULT, " or ' (COBOL only)
 4  SQL STRING DELIMITER ===> DEFAULT  DEFAULT, " or '
 5  DIST SQL STR DELIMTR ===> '        ' or "
 6  MIXED DATA           ===> NO       NO or YES for mixed DBCS data
 7  EBCDIC CCSID         ===>          CCSID of SBCS or mixed data. 1-65533.
 8  ASCII CCSID          ===>          CCSID of SBCS or mixed data. 1-65533.
 9  UNICODE CCSID        ===> 1208     CCSID of UNICODE UTF-8 data.
10  DEF ENCODING SCHEME  ===> EBCDIC   EBCDIC, ASCII, or UNICODE
11  APPLICATION ENCODING ===> EBCDIC   EBCDIC, ASCII, UNICODE, ccsid (1-65533)
12  LOCALE LC_CTYPE      ===>

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

Figure 22. Application programming defaults panel: DSNTIPF

1. **LANGUAGE DEFAULT**

Acceptable values:	see below
Default:	IBMCOB
Update:	option 14 on panel DSNTIPB
DSNHDECP:	DEFLANG

Specify the default programming language for your site. Use any of the following values:

ASM for High Level Assembler/MVS
C for C Language
CPP for C++
IBMCOB for Enterprise COBOL for z/OS
FORTRAN for Fortran
PLI for PL/I

If you specify C or CPP in this field, you can fold SQL identifiers to uppercase. However, this is not a default from any installation panel. For more information about this precompiler option, see *DB2 Application Programming and SQL Guide*.

2. **DECIMAL POINT IS**

Acceptable values:	. (period) or , (comma)
Default:	. (period)
Update:	recommended only to recover an error
DSNHDECP:	DECIMAL

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

This parameter is used in the following cases:

- For running dynamic SQL statements with DYNAMICRULES:
  - Whether the value of field DECIMAL POINT IS is COMMA or PERIOD, DB2 recognizes the value as the decimal point for numbers.
- For binding, defining, or invoking dynamic SQL statements with DYNAMICRULES:
  - If the value of field USE FOR DYNAMICRULES is NO, DB2 does not use the value in field DECIMAL POINT IS if you specify the option COMMA or PERIOD when you precompile the application that contains the dynamic SQL statements. DB2 uses the precompiler option to determine the decimal point for numbers.
  - If the value of field USE FOR DYNAMICRULES is YES, and the value of field DECIMAL POINT IS is PERIOD or COMMA, DB2 recognizes the value as the decimal point for numbers.
- For static SQL statements in COBOL programs, DECIMAL POINT IS specifies the default precompiler option (PERIOD or COMMA).

This parameter is the default for binds at this DB2 site that are requested by a remote system that does not indicate whether the period or the comma is used to represent a decimal point. In most cases, however, requesting systems give DB2 this information.

3. **STRING DELIMITER**

Acceptable values:	DEFAULT, " (quotation mark), ' (apostrophe)
Default:	DEFAULT
Update:	option 14 on panel DSNTIPB
DSNHDECP:	DELIM

Specify the value of the string delimiter for COBOL. If you specify DEFAULT, the string delimiter is the quotation mark. This option is effective for all varieties of COBOL. See field 5 for a description of how to use this field to get the desired set of character string delimiters for COBOL and SQL.

4. **SQL STRING DELIMITER**

Acceptable values:	DEFAULT, " (quotation mark), ' (apostrophe)
Default:	DEFAULT

Update:	not recommended
DSNHDECP:	SQLDELI

Specify the value of the SQL string delimiter that sets off character strings in dynamic SQL. This option is effective for all varieties of COBOL.

The value in this field also determines which character is the escape character for delimited identifiers in dynamic SQL. If you specify an apostrophe in this field, you get a quotation mark for your SQL escape character. If you specify a quotation mark in this field, you get an apostrophe for your SQL escape character.

For SQL statements that are embedded in COBOL programs, COBOL precompiler options specify which character is the SQL string delimiter and which character is the SQL escape character. If you specify DEFAULT in this field, a quotation mark is passed to the precompiler as the default SQL string delimiter.

Some applications might require a particular value for the SQL STRING DELIMITER. Determine the required values for those applications before installing DB2.

Table 38 shows you the different combinations of character string delimiters you get by specifying different values in fields 4 and 5.

Table 38. Effect of fields 4 and 5 on SQL and COBOL string delimiters

When you want this combination of character string delimiters...			**Specify this in field 4**	**Specify this in field 5**
COBOL	**Dynamic SQL**	**Embedded SQL**		
"	'	"	DEFAULT	DEFAULT
'	'	'	'	'
"	"	"	"	"
"	'	'	"	'

The values that you specify in fields 4 and 5 are also used by the program preparation panels, the DSNH CLIST, and the precompiler. Table 39 shows you why you might specify different combinations of values in these fields.

Table 39. Effect of fields 4 and 5 on precompiler options

Purpose	**Field 4**	**Field 5**
Force APOST default (even in COBOL) and provide a default similar to APOST in DB2 Server for VSE & VM	'	'
Change dynamic query string delimiter to the quotation mark. Helpful if you use COBOL with the QUOTE option—allows queries to be tested with dynamic SQL and moved into the program more easily	"	"
Compatibility with the DB2 Server for VSE & VM option	"	'

5. **DIST SQL STR DELIMTR**

Acceptable values:	' (apostrophe) or " (quotation mark)
Default:	' (apostrophe)
Update:	not recommended

DSNHDECP:	DSQLDELI

Specify whether the apostrophe or the quotation mark is used as the SQL string delimiter for bind operations at this DB2 site when the requester does not give DB2 that information. In most cases, requesters tell DB2 whether the apostrophe or the quotation mark is to be used as the SQL string delimiter.

6. **MIXED DATA**

Acceptable values:	YES, NO
Default:	NO
Update:	not recommended
DSNHDECP:	MIXED

Indicates how the EBCDIC CCSID and ASCII CCSID fields are to be interpreted by DB2. The MIXED DATA option has no effect on the UNICODE CCSID field. Regardless of the setting for MIXED DATA, UNICODE UTF-8 data is considered mixed data and is processed according to the rules for mixed data.

Important: MIXED DATA applies to both the EBCDIC CCSID and ASCII CCSID. If you choose MIXED DATA YES, you must select mixed CCSIDs for EBCDIC and ASCII.

With MIXED DATA YES, the CCSID that is specified in the EBCDIC CCSID and ASCII CCSID field must be the mixed CCSID for the encoding scheme. See Table 122 on page 517 and Table 123 on page 517 for the appropriate CCSID for the encoding scheme. From this, DB2 determines the associated SBCS and DBCS CCSIDs for the encoding scheme. MIXED DATA YES allows EBCDIC and ASCII mixed-character data and graphic data to be defined.

With MIXED DATA NO, the CCSID specified in the EBCDIC CCSID or ASCII CCSID field must be the CCSID for the encoding scheme. MIXED DATA NO does not allow for EBCDIC or ASCII mixed character data or graphic data to be defined.

For EBCDIC data, specify whether the code points X'0E' and X'0F' have special meaning as the shift-out and shift-in controls for character strings that include double-byte characters.

- NO indicates that these code points have no special meaning. Therefore, all character strings are single-byte character set (SBCS) data.
- YES indicates that these code points have the special meaning described above. Therefore, character strings can be either SBCS or MIXED data.

7. **EBCDIC CCSID**

Acceptable values:	1 to 65533
Default:	(none)
Update:	not recommended; data integrity may be compromised
DSNHDECP:	SCCSID (single-byte), MCCSID (mixed), GCCSID (graphic)

Specify the default CCSID for EBCDIC-encoded character data that is stored in your DB2 subsystem or data sharing system. DB2 uses this value to perform conversion of character data that is received from external sources including other database management systems. Choose this value carefully to avoid loss of data integrity.

The values that you choose for EBCDIC CCSID and ASCII CCSID are closely related. You must choose values for these parameters that

If you specify MIXED DATA NO, the MCCSID and GCCSID values are 65534. If you specify MIXED DATA YES, ensure that you use the correct single-byte CCSID and MCCSID to use. To determine what single-byte CCSID to use, see Table 122 on page 517 To select a CCSID for mixed data (an MCCSID), see Table 123 on page 517.

Recommendation: Use this parameter to specify an MCCSID. By doing so, you also receive system CCSIDs for your SBCS and GRAPHIC data. If you edit DSNHDECP directly, you will not receive the system CCSIDS for your SBCS and GRAPHIC data.

Conversions are determined in the following order:
1. SYSIBM.SYSSTRINGS.
2. z/OS Unicode conversion services.

See also *Character Data Representation Architecture Overview* and *Character Data Representation Architecture Reference and Registry* for more information.

Considerations for mixed data:
- If MIXED DATA=YES, you must specify a MCCSID from Table 122 on page 517. An error occurs if you do not specify a CCSID or if the CCSID you specify is not listed in the table.
- If you specify 930, 1390, or 5026, Katakana characters are allowed in ordinary identifiers, and letters are not changed to uppercase.

If you specify a CCSID that is recognized by DB2 but is inappropriate for your site, data might be corrupted. For example, assume that the coded character set at your site is 37, but you specify 500 as the system CCSID. If DB2 receives data with a CCSID of 37, the data might be corrupted because character conversion does not occur. Conversely, if DB2 receives data with a CCSID other than 500 and a conversion is made from that CCSID to 500, the data may be corrupted because character conversion does occur.

Altering CCSIDs can be very disruptive to a system. Converting to a CCSID that supports the euro symbol is potentially less disruptive because specific pre-euro CCSIDs map to specific CCSIDs for the euro. See “Converting to the euro symbol” on page 518 for the detailed steps. Converting to a different CCSID for other reasons, particularly when a DB2 system has been operating with the wrong CCSID, could render data unusable and unrecoverable.

Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.

8. **ASCII CCSID**

Acceptable values:	1 to 65533
Default:	(none)
Update:	not recommended; data integrity may be compromised

DSNHDECP:	ASCCSID (single-byte), AMCCSID (mixed), AGCCSID (graphic)

Specify the default CCSID for ASCII-encoded character data that is stored in your DB2 subsystem or data sharing system. DB2 uses this value to perform conversion of character data that is received from external sources, including other database management systems. You must specify a value for this field, even if you do not have or plan to create ASCII-encoded objects. Choose this value carefully to prevent loss of data integrity.

To determine which single-byte ASCII CCSID to use, see Table 121 on page 515. To determine which double-byte ASCII CCSID to use, see Table 123 on page 517.

For more considerations, see the discussion for the 167 field.

Recommendation: Never change CCSIDs on an existing DB2 system without specific guidance from IBM Software Support.

9. **UNICODE CCSID**

Acceptable values:	1208
Default:	1208
Update:	not recommended; data integrity may be compromised
DSNHDECP:	USCCSID (367 for single-byte), UMCCSID (1208 for mixed), UGCCSID (1200 for graphic)

Accept the default CCSID for Unicode. DB2 currently allows specification of only CCSID 1208 for this value. DB2 automatically chooses the CCSIDs for double-byte and single-byte data. Do not change CCSID values after they have been specified. SQL results might be unpredictable if you do not accept the default.

10. **DEF ENCODING SCHEME**

Acceptable values:	EBCDIC, ASCII, UNICODE
Default:	EBCDIC
Update:	not recommended
DSNHDECP:	ENSCHEME

Specify the default format in which to store data in DB2. If you specify DEF ENCODING SCHEME=ASCII or EBCDIC and MIXED DATA=YES, specify a mixed CCSID.

The DDL uses the default encoding scheme in the following cases:
- CREATE DATABASE
- CREATE DISTINCT TYPE
- CREATE FUNCTION
- CREATE GLOBAL TEMPORARY TABLE
- DECLARE GLOBAL TEMPORARY TABLE
- CREATE TABLESPACE (in DSNDB04 database)

11. **APPLICATION ENCODING**

Acceptable values:	ASCII, EBCDIC, UNICODE, or ccsid (1 to 65533)
Default:	EBCDIC
Update:	not recommended
DSNHDECP:	APPENSCH

Specify the system default application encoding scheme.

The system default application encoding scheme affects how DB2 interprets data coming into DB2. For example, if you set your default application encoding scheme to 37, and your EBCDIC coded character set to 500, DB2 converts all data coming into the system to 500 from 37 before using it. This includes, but is not limited to, SQL statement text and host variables.

The default value, EBCDIC, causes DB2 to retain the behavior of previous versions of DB2. (Assume that all data is in the EBCDIC system CCSID.)

12. **LOCALE LC_CTYPE**

Acceptable values:	A valid locale of 0 to 50 characters. See *OS/390 C/C++ Programming Guide* or *DB2 SQL Reference* for the format of valid LOCALE LC_CTYPE names.
Default:	Blank
Update:	option 17 of DSNTIPB
DSNHDECP:	LC_TYPE

Specify the system LOCALE LC_CTYPE. A *locale* is the part of your system environment that depends on language and cultural conventions. An LC_TYPE is a subset of a locale that applies to character functions.

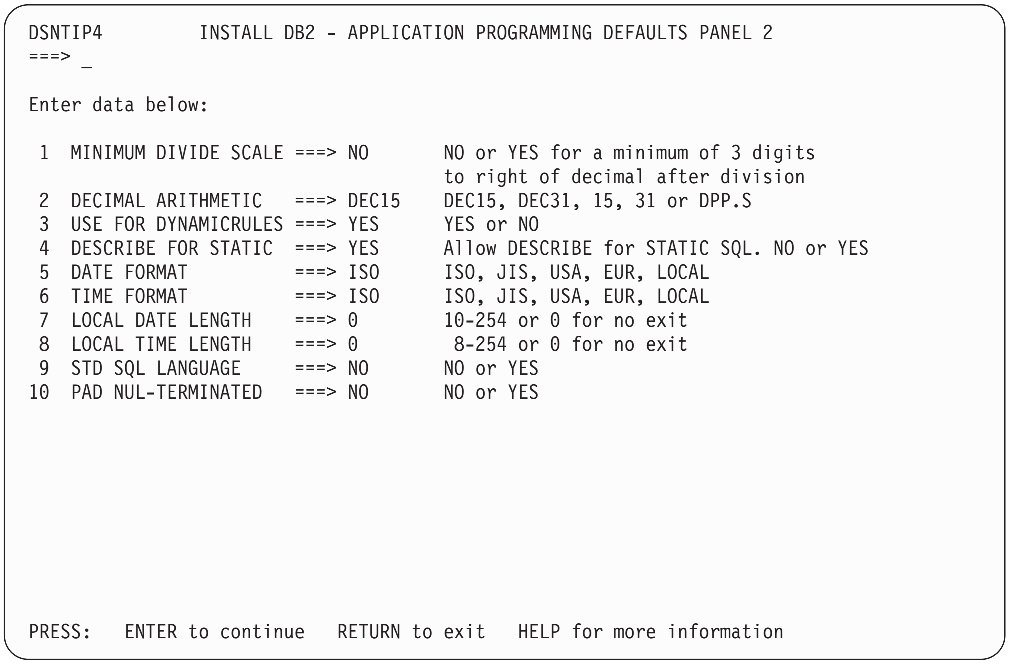
The UPPER, LOWER, and TRANSLATE scalar functions use the CURRENT LOCALE LC_CTYPE system default or special register. The results of these functions can vary, depending on the setting of the locale.

Recommendation: Use the default value for LOCALE LC_CTYPE unless you need to execute the UPPER, LOWER, or TRANSLATE functions for data that must be interpreted by using the rules provided by specific locales. For example, specify En_US for English in the United States or Fr_CA for French in Canada.

Application programming defaults panel 2: DSNTIP4

This panel is a continuation of DSNTIPF and is used to set application programming defaults. The values that you specify on this panel are used as default values by the program preparation panels, the program preparation CLIST (DSNH), and the precompiler. They can also be used as defaults by other programs, such as DB2 QMF.

For more information about the fields in this panel, see *DB2 SQL Reference*.



```
DSNTIP4        INSTALL DB2 - APPLICATION PROGRAMMING DEFAULTS PANEL 2
===> _

Enter data below:

 1  MINIMUM DIVIDE SCALE ===> NO       NO or YES for a minimum of 3 digits
                                       to right of decimal after division
 2  DECIMAL ARITHMETIC   ===> DEC15    DEC15, DEC31, 15, 31 or DPP.S
 3  USE FOR DYNAMICRULES ===> YES      YES or NO
 4  DESCRIBE FOR STATIC  ===> YES      Allow DESCRIBE for STATIC SQL. NO or YES
 5  DATE FORMAT          ===> ISO      ISO, JIS, USA, EUR, LOCAL
 6  TIME FORMAT          ===> ISO      ISO, JIS, USA, EUR, LOCAL
 7  LOCAL DATE LENGTH    ===> 0        10-254 or 0 for no exit
 8  LOCAL TIME LENGTH    ===> 0         8-254 or 0 for no exit
 9  STD SQL LANGUAGE     ===> NO       NO or YES
10  PAD NUL-TERMINATED   ===> NO       NO or YES








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

Figure 23. Application programming defaults panel: DSNTIP4

1. **MINIMUM DIVIDE SCALE**

Acceptable values:	YES, NO
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DECDIV3

Specify YES to retain at least three digits to the right of the decimal point after any decimal division. Certain accounting applications might need this option. Use NO, the default, to accept the usual rules for decimal division in SQL. For more information about decimal division in SQL, see *DB2 SQL Reference*.

2. **DECIMAL ARITHMETIC**

Acceptable values:	DEC15, DEC31, 15, 31
Default:	DEC15
Update:	not recommended; cannot be changed during migration
DSNHDECP:	DECARTH

Specify the rules that are to be used when both operands in a decimal operation have precisions of 15 or less. DEC15 specifies the rules that do not allow a precision greater than 15 digits, and DEC31 specifies the rules that allow a precision of up to 31 digits. The rules for DEC31 are always used if either operand has a precision greater than 15. If you chose DEC15 for your previous installation, choosing DEC31 can produce different results for operations on existing data.

This installation option applies to dynamic SQL by becoming the initial value for the CURRENT PRECISION special register, and it provides the default for the DEC precompiler option. DEC15 is sufficient for most sites. Do not choose DEC31 unless you are certain that you need the extra precision. If you use DEC31, you are more likely to get a bind error, particularly in division operations. See *DB2 SQL Reference* for information about arithmetic with two decimal operands.

3. **USE FOR DYNAMICRULES**

Acceptable values:	YES or NO
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNHDECP:	DYNRULS

Specify whether DB2 should use the application programming defaults that are specified on this panel or use the values of the DB2 precompiler options for dynamic SQL statements that are bound by using DYNAMICRULES bind, define, or invoke behavior.

Specify YES to use the application programming defaults for these fields regardless of the DYNAMICRULES option:
- DECIMAL POINT IS
- STRING DELIMITER
- SQL STRING DELIMITER
- MIXED DATA
- DECIMAL ARITHMETIC

Specify NO to use the DB2 precompiler values for dynamic SQL statements in plans or packages bound using the DYNAMICRULES bind, define, or invoke behavior.

4. **DESCRIBE FOR STATIC**

Acceptable values:	NO or YES
Default:	YES
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DESCSTAT

Specify whether DB2 is to build a DESCRIBE SQLDA when binding static SQL statements. Normally, a DESCRIBE cannot be issued against a static SQL statement, with the following exceptions:
- In a distributed environment, where DB2 UDB for z/OS is the server, and the requester supports extended dynamic SQL. In this scenario, a DESCRIBE request that is executed on an SQL statement in the extended dynamic package appears to DB2 as a DESCRIBE on a static SQL statement in the DB2 package.
- When an application uses a stored procedure result set, and the application must allocate a cursor for that result set. The application can describe that cursor by using a DESCRIBE CURSOR statement. The SQL statement that is actually

described is the one for which the cursor is declared in the stored procedure. If that statement is static, this requires that a static SQL statement must be described.

NO means that DB2 does not generate a DESCRIBE SQLDA at bind time for static SQL statements. If a DESCRIBE request is received at execution time, DB2 generates an error. However, if the describe request comes from a DESCRIBE CURSOR statement, DB2 satisfies the request but is able to provide only data type and length information. Column names are not provided.

YES, the default, means that DB2 does generate a DESCRIBE SQLDA at bind time so that DESCRIBE requests for static SQL can be satisfied during execution. You must rebind the package after this value has been set to YES. Specifying YES increases the size of some packages because the DESCRIBE SQLDA is now stored with each statically bound SQL SELECT statement.

5. **DATE FORMAT**

Acceptable values:	ISO, USA, EUR, JIS, LOCAL
Default:	ISO
Update:	not recommended
DSNHDECP:	DATE

Specify one of the abbreviations that are shown in Table 40 as a default output format to represent dates.

Table 40. Date formats

Format name	Abbreviation	Format	Example
International Standards Organization	ISO	yyyy-mm-dd	2003-12-23
IBM USA standard	USA	mm/dd/yyyy	12/23/2003
IBM European standard	EUR	dd.mm.yyyy	23.12.2003
Japanese Industrial Standard Christian Era	JIS	yyyy-mm-dd	2003-12-23
Locally defined (by an installation exit routine)	LOCAL	your choice	

DB2 can accept a date in any format as input. DB2 interprets the input date based on the punctuation and then provides date output in the format that you specify for this parameter. If you use LOCAL, you must provide a date exit routine to perform date formatting; for information, see Appendixes (Volume 2) of *DB2 Administration Guide*.

6. **TIME FORMAT**

Acceptable values:	ISO, USA, EUR, JIS, LOCAL
Default:	ISO
Update:	not recommended
DSNHDECP:	TIME

Specify one of the formats that are shown in Table 41 on page 174 as a default output to represent times.

Table 41. Time formats

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
Locally defined (by exit routine)	LOCAL	your choice	

DB2 can accept a time in any format as input. DB2 interprets the input time based on the punctuation and then provides time output in the format you specify for this parameter. If you use LOCAL, you must provide a time exit routine to perform time formatting; for information, see Appendix B (Volume 2) of *DB2 Administration Guide*.

7. **LOCAL DATE LENGTH**

Acceptable values:	0, 10 to 254
Default:	0
Update or migrate:	not recommended
DSNHDECP:	DATELEN

Accept the default value **0** if you want to use one of the IBM-supplied date formats (ISO, JIS, USA, or EUR). This indicates that no user-defined date format exists in your system. If you use a locally defined date exit routine, enter the length of the longest field that is required to hold a date. If you want your own date format to be the default, enter LOCAL for field 1 on this panel.

8. **LOCAL TIME LENGTH**

Acceptable values:	0, 8 to 254
Default:	0
Update or migrate:	not recommended
DSNHDECP:	TIMELEN

Accept the default value **0** if you want to use one of the IBM-supplied time formats (ISO, JIS, USA, or EUR). This indicates that no user-defined time format exists in your system. If you use a locally defined time exit routine, enter the length of the longest field that is required to hold a time. If you want your own time format to be the default, enter LOCAL for field 2 on this panel.

9. **STD SQL LANGUAGE**

Acceptable values:	YES, NO
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNHDECP:	STDSQL

To specify that the SQL language that is used in application programs conforms to the portions of the 1992 ANSI SQL standard that are implemented by DB2, choose YES.

If you choose NO, you specify that programs are written in accordance with the SQL language that is defined by DB2.

10. **PAD NUL-TERMINATED**

Acceptable values:	NO, YES
Default:	NO
Update:	option 15 on panel DSNTIPB
DSNZP*xxx*:	DSNHDECP PADNTSTR

Specify whether output host variables that are nul-terminated strings are padded with blanks and a nul-terminator.

If NO is specified, nul-terminated output host variables have the nul-terminator placed at the end of actual data that is returned in the host variable.

If YES is specified, nul-terminated output host variables have the nul-terminator placed at the end of the string, after the string has been padded with blanks from the end of the actual data to the length of the output host variable.

Performance and optimization panel: DSNTIP8

This panel is a continuation of DSNTIP4 and is used to set application programming defaults pertaining to performance and optimization.

```
DSNTIP8          INSTALL DB2 - PERFORMANCE AND OPTIMIZATION
===> _

Enter data below:

 1  CURRENT DEGREE        ===> 1         1 or ANY
 2  CACHE DYNAMIC SQL     ===> YES       NO or YES
 3  OPTIMIZATION HINTS    ===> NO        Enable optimization hints. NO or YES
 4  VARCHAR FROM INDEX    ===> NO        Get VARCHAR data from index. NO or YES
 5  RELEASE LOCKS         ===> YES       Release cursor with held locks. YES, NO
 6  MAX DEGREE            ===> 0         Maximum degree of parallelism. 0-254
 7  UPDATE PART KEY COLS  ===> YES       Allow update of partitioning key
                                        columns. YES, NO, SAME
 8  LARGE EDM BETTER FIT  ===> NO        NO or YES
 9  IMMEDIATE WRITE       ===> NO        NO, YES
10  EVALUATE UNCOMMITTED  ===> NO        Evaluate uncommitted data. NO or YES
11  SKIP UNCOMM INSERTS   ===> NO        Skip uncommitted inserts. NO or YES
12  CURRENT REFRESH AGE   ===> 0         0 or ANY
13  CURRENT MAINT TYPE    ===> SYSTEM    NONE, SYSTEM, USER, ALL
14  STAR JOIN QUERIES     ===> DISABLE   DISABLE, ENABLE, 1-32768
15  STAR JOIN MAX POOL    ===> 20        0-1024


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

Figure 24. Performance and optimization panel: DSNTIP8

1. **CURRENT DEGREE**

Acceptable values:	1, ANY
Default:	1
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CDSSRDEF

Specify the default for the CURRENT DEGREE special register when no degree is explicitly set using the SQL statement SET CURRENT DEGREE.

Recommendation: In almost all situations, accept the default value of 1. You should use parallelism selectively where it provides value, rather than globally. Although parallelism can provide a substantial reduction in elapsed time for some queries with only a modest overhead in processing time, parallelism does not always provide the intended benefit. For some queries and in many other situations, query parallelism does not provide an improvement, or it uses too many resources. If you are using nearly all of your CPU, I/O, or storage resources, parallelism is more likely to cause degradation of performance. Use parallelism only where it is most likely to provide benefits.

2. **CACHE DYNAMIC SQL**

Acceptable values:	YES, NO
Default:	YES
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHEDYN

Specify whether to cache prepared, dynamic SQL statements for later use by eligible application processes. These prepared statements are cached in the EDM dynamic statement cache. When specifying YES, consider this usage when calculating your EDM pool size. See "Calculating the EDM pool space for the prepared-statement cache" on page 39 for details about estimating storage. If you specify YES, you must specify YES for USE PROTECTION on panel DSNTIPP.

3. **OPTIMIZATION HINTS**

Acceptable values:	NO or YES
Default:	NO
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM OPTHINTS

Specify that you are to pass information to DB2 that might influence the access path selected for certain queries. The information is passed in the form of rows in the PLAN_TABLE. If you accept the default value, you cannot use optimization hints. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information on the PLAN_TABLE.

4. **VARCHAR FROM INDEX**

Acceptable values:	NO or YES
Default:	NO
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RETVLCFK

Specify whether the VARCHAR column is to be retrieved from a padded index. The data sharing scope of this parameter is GROUP.

If you choose NO, DB2 must go to the data to retrieve data because index-only access of varying-length column data in padded indexes is not possible. With a setting of NO, data is retrieved with no padding.

If you choose YES, better performance results because index-only access of varying-length column data is enabled for padded indexes. The data that is retrieved from the index is padded with blanks to the maximum length of the column.

Important: Applications must be able to handle the padding blanks. If your application is sensitive to these blanks, keep the default value of NO or consider using non-padded indexes. You must rebind plans and packages to enable the change.

Recommendation: Accept the default value and use padded indexes. Use NOT PADDED indexes if you want the index to use less space, to allow index-only retrieval with variable characters, and do not want the incompatible retrieval of the full column width.

5. **RELEASE LOCKS**

Acceptable values:	YES or NO
Default:	YES
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RELCURHL

Specify whether DB2 should, at commit time, release a data or row lock on which a cursor that is defined WITH HOLD is positioned. This lock is **not** necessary for maintaining cursor position. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about locks for cursors that are defined WITH HOLD.

If you choose YES, the default, DB2 releases this data or row lock after a COMMIT is issued for cursors that are defined WITH HOLD. Specifying YES can improve concurrency.

If you choose NO, DB2 holds the data or row lock for WITH HOLD cursors after the COMMIT. This option is provided so that existing applications that rely on this lock can continue to work correctly.

6. **MAX DEGREE**

Acceptable values:	0 to 254
Default:	0
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PARAMDEG

Specify the maximum degree of parallelism for a parallel group. When you specify a value, you limit the degree of parallelism so that DB2 cannot create too many parallel tasks that use virtual storage. The default value of 0 means that DB2 will choose a maximum degree of parallelism that is based on the system configuration. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about limiting the degree of parallelism.

7. **UPDATE PART KEY COLS**

Acceptable values:	YES, NO, or SAME
Default:	YES
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM PARTKEYU

Specify whether values in columns that participate in partitioning keys can be updated. The acceptable values are:

YES Values in columns that participate in partitioning keys can be updated. This is the default.

NO Values in columns that participate in partitioning keys cannot be updated.

SAME Values in columns that participate in partitioning keys can be updated if and only if the update leaves the row belonging to the same partition.

| Specifying NO or SAME will not improve concurrency. DB2 does not take
| exclusive control of the objects to perform the update.

Attempts to inappropriately update the value in a partitioning key column result in the failure of the SQL statement with a -904 resource-unavailable SQL code. The accompanying DSNT501I message identifies the unavailable resource as code type X'3000' and a reason code of 00C900C7.

8. **LARGE EDM BETTER FIT**

Acceptable values:	YES, NO
Default:	NO
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMBFIT

Specify how free space is to be utilized for large EDM pools (greater than 40 MB). Specify NO to optimize for performance. YES optimizes for better storage utilization. In the trade-off between performance and space utilization, space is normally more critical for smaller EDM pools, and performance is more critical for larger EDM pools.

9. **IMMEDIATE WRITE**

Acceptable values:	YES, NO
Default:	NO
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6GRP IMMEDWRI

Specify when updates to group buffer pool-dependent buffers are to be written to the coupling facility (CF). If you specify NO, DB2 does not immediately write the change buffer to the CF. Instead, DB2 waits until phase 1 of commit. If you specify YES, DB2 immediately writes the page to the CF after the update occurs.

If either the IMMEDWRITE bind option or the IMMEDWRI subsystem parameter is set to YES, the value of the immediate write option at run time is also YES.

10. **EVALUATE UNCOMMITTED**

Acceptable values:	NO, YES
Default:	NO
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EVALUNC

Specify whether predicate evaluation can occur on uncommitted data of other transactions. The option applies only to stage 1 predicate processing that uses table access (table space scan, index-to-data access, and RID-list processing) for queries with isolation level RS or CS.

Although the option influences whether predicate evaluation can occur on uncommitted data, it does not influence whether uncommitted data is returned to an application. Queries with isolation level RS or CS return only committed data. They never return the uncommitted data of other transactions, even if predicate evaluation occurs. If data satisfies the predicate during evaluation, the data is locked as needed, and the predicate is re-evaluated as needed before the data is returned to the application.

If you specify NO, the default, predicate evaluation occurs only on committed data (or on the application's own uncommitted changes). NO ensures that all qualifying data is always included in the answer set.

If you specify YES, predicate evaluation can occur on uncommitted data of other transactions. With YES, data might be excluded from the answer set. Data that does not satisfy the predicate during evaluation but then, because of undo processing (ROLLBACK or statement failure), reverts to a state that does satisfy the predicate is missing from the answer set. A value of YES enables DB2 to take fewer locks during query processing. The number of avoided locks depends on:

- The query's access path
- The number of evaluated rows that do not satisfy the predicate
- The number of those rows that are on overflow pages

Recommendation: Specify YES to improve concurrency if your applications can tolerate returned data that might falsely exclude any data that would be included as the result of undo processing (ROLLBACK or statement failure).

11. **SKIP UNCOMM INSERTS**

Acceptable values:	NO, YES
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SKIPUNCI

Specify whether cursors bound with read stability or cursor stability will ignore uncommitted inserts made by other transactions.

If you specify NO, the default value, uncommitted inserts are evaluated for return with or without waiting for committedness according to the value that you set in the EVALUATE UNCOMMITTED field.

If you specify YES, uncommitted inserts will be treated as if they had not yet arrived. In a data sharing environment, uncommitted inserts of transactions that were designated as spawning transactions will be treated in this manner. Immediate write occurs for inserts, updates, and deletes, but readers do not wait for the outcome of uncommitted inserts.

12. **CURRENT REFRESH AGE**

Acceptable values:	0, ANY
Default:	0
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM REFSHAGE

Specify the default value for the CURRENT REFRESH AGE special register when no value is explicitly set using the SQL statement SET CURRENT REFRESH AGE. Accepting the default value disables query rewrite using deferred materialized query tables.

13. **CURRENT MAINT TYPES**

Acceptable values:	NONE, SYSTEM, USER, ALL
Default:	SYSTEM
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAINTYPE

Specify the default value for the CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION special register when no value is explicitly set by using the SQL statement SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION. Accepting the default value allows query rewrite using system-maintained materialized query tables (SYSTEM) when CURRENT REFRESH AGE is set to ANY. Alternatively, specifying USER allows query rewrite by using user-maintained materialized query tables when CURRENT REFRESH AGE is set to ANY. Specifying ALL allows query rewrite by using both system-maintained and user-maintained materialized query tables.

In a data sharing environment, this parameter has member scope.

14. **STAR JOIN QUERIES**

Acceptable values:	DISABLE, ENABLE, 1 to 32768
Default:	DISABLE
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM STARJOIN

Specify whether star join processing is enabled. A value of 1 indicates that the fact table will be the largest table in a star join query that does not have fact/dimension ratio checking. A value of 2 to 32768 indicates that DB2 should use the ratio of the star join table and the largest dimension table.

15. **STAR JOIN MAX POOL**

Acceptable values:	0 to 1024
Default:	20
Update:	option 16 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SJMXPOOL

Specify the maximum size, in MB, of the virtual memory pool for star join queries. If you specify zero, DB2 does not allocate a memory pool for star join queries, even if star join queries are enabled. If you specify a value between 1 and 1024, DB2 creates a dedicated memory pool up to the specified size for star join queries.

Recommendation: Choose a value that avoids system paging that can slow down the entire system throughput. For more information about choosing a value, see Part 5 (Volume 2) of *DB2 Administration Guide*.

IRLM panel 1: DSNTIPI

The entries on this panel affect the installation of the internal resource lock manager (IRLM).

You must use one IRLM for each DB2 subsystem. See “IRLM address space (IRLMPROC)” on page 29 for more information about DB2 and IRLM. See *DB2 Data Sharing: Planning and Administration* for recommendations about choosing values for fields on this panel.

Updating the parameters: The update option in each parameter description in this topic indicates the correct procedure:

Note A	Change by editing the IRLM start procedure.
Note B	Change by editing the associated parameter in job DSNTIJUZ, the IRLM start procedure, and input member DSNTIDXA. Then, execute DSNTIJUZ and restart DB2.

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

Enter data below:

 1  INSTALL IRLM       ===> YES        IRLM is required for DB2. Should the
                                       IRLM distributed with DB2 be installed?
 2  SUBSYSTEM NAME     ===> IRLM       IRLM MVS subsystem name
 3  RESOURCE TIMEOUT   ===> 60         Seconds to wait for unavailable resource
 4  AUTO START         ===> YES        Start IRLM if not up. YES or NO
 5  PROC NAME          ===> IRLMPROC   Name of start procedure for IRLM
 6  TIME TO AUTOSTART  ===> 300        Time DB2 will wait for IRLM autostart
 7  UTILITY TIMEOUT    ===> 6          Utility wait time multiplier
 8  U LOCK FOR RR/RS   ===> NO         Lock mode for update cursor with
                                       RR or RS isolation. YES or NO
 9  X LOCK FOR SEARCHED U/D ===> NO    Use X lock for serached updates or
                                       deletes. NO or YES or TARGET.
10  START IRLM CTRACE  ===> NO         Start IRLM component traces at startup?
                                       NO or YES
11  IMS BMP TIMEOUT    ===> 4          Timeout multiplier for BMP. 1-254
12  DL/I BATCH TIMEOUT ===> 6          Timeout multiplier for DL/I. 1-254
13  RETAINED LOCK TIMEOUT ===> 0       Retained lock timeout multiplier. 0-254
PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 25. IRLM panel 1: DSNTIPI

1. **INSTALL IRLM**

Acceptable values:	YES, NO
Default:	YES
Update:	see note B on 182
DSNZP*xxx*:	none

Specify whether to provide IRLM subsystem entries in job DSNTIJMV and to build an IRLM procedure. If you specify NO, no IRLM procedure is produced. On installation panel DSNTIPJ, all values are ignored with the exception of fields 3 and 4.

If you specify YES, the required entries are provided, and the IRLM procedure is built.

If you do not have a new IRLM procedure created, ensure that your old IRLM procedure is updated with any new keywords that were added.

2. **SUBSYSTEM NAME**

Acceptable values:	1 to 4 characters. First must be A-Z, #, $, or @. Others must be A-Z, 1-9, #, $, or @.
Default:	IRLM
Update:	see note B on 182
DSNZP*xxx*:	DSN6SPRM IRLMSID

Specify the name by which z/OS knows the IRLM subsystem. The name is used for communication between DB2 and the IRLM. This name is included in the z/OS subsystem table IEFSSN*xx*, where *xx* is the value that you supply in field 3 (SUBSYSTEM MEMBER) on installation panel DSNTIPM found on 199.

If you installed the IRLM for IMS, DB2's IRLM name must be different. Two IRLMs that reside in the same z/OS system must have unique z/OS subsystem names. If you already have IRLM installed, use the z/OS subsystem name for that IRLM. Otherwise, accept the default value, IRLM. For more information, see *IMS Command Reference*.

IRLM PROC parameter: IRLMNM

3. **RESOURCE TIMEOUT**

Acceptable values:	1-3600
Default:	60
Update:	see note B on 182
DSNZP*xxx*:	DSN6SPRM IRLMRWT

Specify the number of seconds before a time-out is detected. *Timeout* means that a lock request has waited for a resource (or for claims on a resource for a particular claim class to be released) longer than the number of seconds specified on this option. The value that is specified for this option must be a multiple of the DEADLOCK TIME on installation panel DSNTIPJ because IRLM uses its deadlock timer to initiate time-out detection and deadlock detection. This value is rarely the actual time. For data sharing, the actual timeout period is longer than the time-out value. See *DB2 Data Sharing: Planning and Administration* for an explanation of time-outs.

For information about optimizing performance by managing DB2's use of locks, see Part 5 (Volume 2) of *DB2 Administration Guide*.

4. **AUTO START**

Acceptable values:	YES, NO
Default:	YES
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM IRLMAUT

Specify whether DB2 is to automatically start and stop the IRLM.

If you specify YES, when DB2 starts, it tries to start the IRLM if the IRLM is not already started. When DB2 stops, IRLM automatically stops if the IRLM was started by DB2. However, IRLM will not automatically terminate if it is defined with DISCONNECT IRLM = NO.

Recommendation: Use YES if you use the IRLM only for a single DB2 subsystem.

If you specify NO, DB2 terminates if the IRLM is not started when DB2 comes up.

When IRLM initializes, it is registered with the z/OS automatic restart manager (ARM). It deregisters from the ARM when the IRLM is shut down normally. When IRLM terminates, it sends DB2 the registration information so that DB2 can determine whether IRLM was terminated normally. DB2 then deregisters from the ARM to prevent unwanted restarts. Otherwise, IRLM might be restarted with DB2, if AUTO START is YES. See Part 4 (Volume 1) of *DB2 Administration Guide* for information on changing the IRLM policy definition in a non-data sharing environment. See *DB2 Data Sharing: Planning and Administration* for IRLM policy definitions within a data sharing environment.

5. **PROC NAME**

Acceptable values:	1 to 8 characters
Default:	IRLMPROC
Update:	see note B on 182
DSNZP*xxx*:	DSN6SPRM IRLMPRC

Specify the name of the IRLM procedure that z/OS invokes if field 4 is YES. The name **cannot** be the same as the subsystem name in field 2.

The procedure is created during installation or migration by job DSNTIJMV and is placed in SYS1.PROCLIB. You can review it by examining DSNTIJMV.

6. **TIME TO AUTOSTART**

Acceptable values:	1 to 3600
Default:	300
Update:	see note B on 182
DSNZP*xxx*:	DSN6SPRM IRLMSWT

Specify the IRLM wait time in seconds. This is the time that DB2 waits for the IRLM to start during autostart. If the time expires, DB2 abends.

7. **UTILITY TIMEOUT**

Acceptable values:	1 to 254
Default:	6
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM UTIMOUT

Specify the number of resource time-out values (field 3) that a utility or utility command is to wait for a lock or for all claims on a resource of a particular claim class to be released. For example, if you use the default value, a utility can wait six times longer than an SQL application for a resource. This option allows utilities to wait longer than SQL applications to access a resource. For more information, see Part 5 (Volume 2) of *DB2 Administration Guide*.

8. **U LOCK FOR RR/RS:**

	Acceptable values:	YES, NO
	Default:	NO
|	Update:	option 17 on panel DSNTIPB
	DSNZP*xxx*:	DSN6SPRM RRULOCK

Specify whether to use the U (UPDATE) lock when using repeatable read (RR) or read stability (RS) isolation to access a table. If you specify NO, the lock mode for operations with RR or RS is S (SHARE). If the cursor in your applications includes the FOR UPDATE OF clause, but updates are infrequent, S-locks generally provide better performance.

If you specify YES, the lock mode for operations with RR or RS is U. If your applications make frequent updates with repeatable-read isolation, the U-lock might provide greater concurrency than the S-lock. However, applications that require high concurrency are almost always more efficient if they use cursor stability (CS) isolation.

9. **X LOCK FOR SEARCHED U/D:**

#	Acceptable values:	YES, NO, TARGET
	Default:	NO
|	Update:	option 17 on panel DSNTIPB
	DSNZP*xxx*:	DSN6SPRM XLKUPDLT

Specify the locking method that is to be used when performing a searched update or delete. A value of NO means DB2 uses an S- or U-lock when scanning for qualifying rows. For any qualifying rows or pages, the lock is upgraded to an X-lock before performing the update or delete. For non-qualifying rows or pages the lock is released if ISOLATION(CS) is used. For ISOLATION(RS) or ISOLATION(RR), an S-lock is retained on the rows or pages until the next commit point. Use this option to achieve higher rates of concurrency.

A value of YES means DB2 uses an X-lock on qualifying rows or pages. For ISOLATION(CS), the lock is released if the rows or pages are not updated or deleted. For ISOLATION(RS) or ISOLATION(RR), an X-lock is retained until the next commit point. A value of YES is beneficial in a data sharing environment when most or all searched updates and deletes use an index. If YES is specified and searched updates or deletes result in a table space scan, the likelihood of time-outs and deadlocks greatly increases.

A value of TARGET means DB2 combines YES and NO behavior. DB2 uses an
X-lock on qualifying rows or pages of the specific table that is targeted by the
update or delete statement. DB2 uses an S- or U-lock when scanning for rows or
pages of other tables that are referenced by the query (for example, tables that are
referenced only in the WHERE clause of the query). For non-qualifying rows or
pages the lock is released if ISOLATION(CS) is used. For ISOLATION(RS) or
ISOLATION(RR), an S-lock is retained on the rows or pages until the next commit
point.

10. **START IRLM CTRACE**

Acceptable values:	NO or YES
Default:	NO
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	none

Specify whether the IRLM component traces should be activated when IRLM is started. The DB2-provided IRLM procedure in DSNTIJMV is tailored according to this value.

Specify NO if you want IRLM to start only the low-activity subtraces EXP, INT, and XIT.

Specify YES if you want IRLM to start with all its subtraces active. Starting all IRLM subtraces has a slight impact on performance demand for ECSA storage, but it improves serviceability.

11. **IMS BMP TIMEOUT**

Acceptable values:	1 to 254
Default:	4
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM BMPTOUT

Specify the number of resource time-out values (field 3) that an IMS BMP connection is to wait for a lock to be released. For example, if you use the default value, an IMS BMP connection can wait 4 times the resource time-out value for a resource. This option gives you flexibility in tuning your system to avoid time-outs. For more information, see ″Installation options for wait times″ in Part 5 (Volume 2) of *DB2 Administration Guide*.

12. **DL/I BATCH TIMEOUT**

Acceptable values:	1 to 254
Default:	6
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DLITOUT

Specify the number of resource time-out values (field 3) that a DL/I batch connection is to wait for a lock to be released. For example, if you use the default value, a DL/I batch application can wait six times the resource timeout value for a resource. This option gives you flexibility in tuning your system to avoid time-outs. For more information, see ″Installation options for wait times″ in Part 5 (Volume 2) of *DB2 Administration Guide*.

13. **RETAINED LOCK TIMEOUT**

Acceptable values:	0 to 254
Default:	0
Update:	option 17 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RETLWAIT

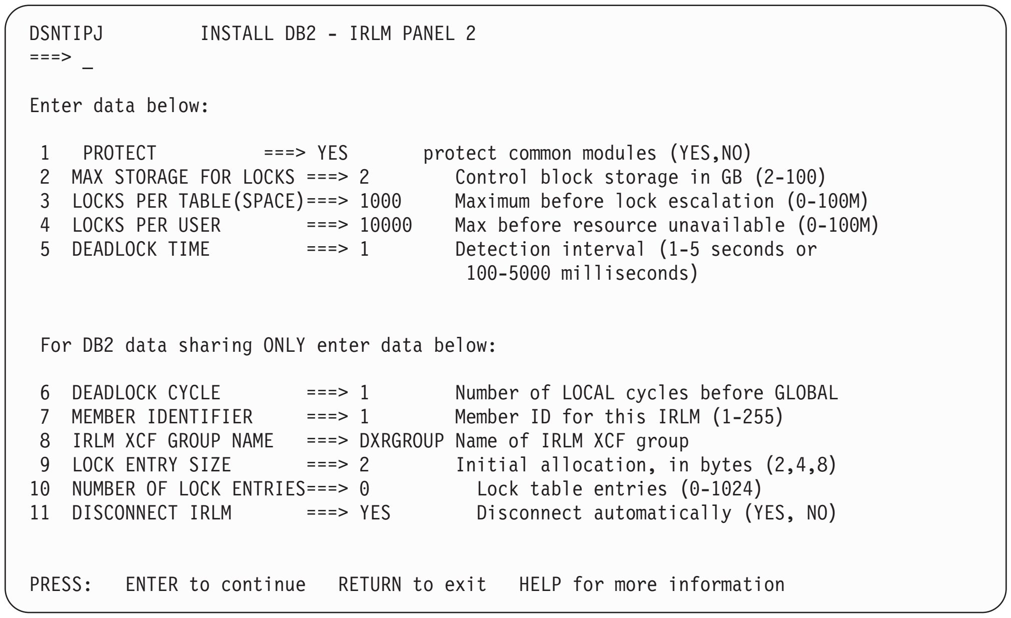
Indicate how long a transaction should wait for a lock on a resource if another DB2 in a data sharing group has failed and is holding an incompatible lock on that resource. This value is of importance only in a data sharing environment. Locks that are held by failed DB2 members are called *retained locks*. For more information, see *DB2 Data Sharing: Planning and Administration*.

The value that you use is a multiplier that is applied to the connection's normal time-out value. For example, if the retained lock multiplier is 2, the timeout period for a call attachment connection that is waiting for a retained lock is 1 * 2 (1 for the normal CAF timeout period, 2 for the additional time that is specified for retained locks).

If you use the default, 0, applications do not wait for incompatible retained locks, but instead the lock request is immediately rejected, and the application receives a resource unavailable SQLCODE.

IRLM panel 2: DSNTIPJ

The entries on this panel affect several of the characteristics of IRLM time-sharing fields and other locking options. The default values are adequate for most sites under ordinary conditions. DB2 and IRLM group names must start with a letter.



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

Enter data below:

 1   PROTECT          ===> YES       protect common modules (YES,NO)
 2  MAX STORAGE FOR LOCKS ===> 2        Control block storage in GB (2-100)
 3  LOCKS PER TABLE(SPACE)===> 1000     Maximum before lock escalation (0-100M)
 4  LOCKS PER USER        ===> 10000    Max before resource unavailable (0-100M)
 5  DEADLOCK TIME         ===> 1        Detection interval (1-5 seconds or
                                         100-5000 milliseconds)


 For DB2 data sharing ONLY enter data below:

 6  DEADLOCK CYCLE        ===> 1        Number of LOCAL cycles before GLOBAL
 7  MEMBER IDENTIFIER     ===> 1        Member ID for this IRLM (1-255)
 8  IRLM XCF GROUP NAME   ===> DXRGROUP Name of IRLM XCF group
 9  LOCK ENTRY SIZE       ===> 2        Initial allocation, in bytes (2,4,8)
10  NUMBER OF LOCK ENTRIES===> 0          Lock table entries (0-1024)
11  DISCONNECT IRLM       ===> YES        Disconnect automatically (YES, NO)


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

Figure 26. IRLM panel 2: DSNTIPJ

1. **PROTECT**

Acceptable values:	NO, YES
Default:	YES
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify whether IRLM loads its common storage modules into page-protected storage.

YES, the default, indicates that modules located in common storage are to be loaded into page-protected storage to prevent programs from overlaying the instructions. YES is recommended because it requires no additional overhead after the modules are loaded, and the protection can prevent code-overlay failures.

NO indicates that common storage modules are to be loaded into CSA or ECSA without first protecting that memory.

IRLM PROC parameter: PGPROT

2. **MAX STORAGE FOR LOCKS**

Acceptable values:	2 to 100
Default:	2
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify, in gigabytes, the maximum amount of private storage available above the 2–GB bar that the IRLM for this DB2 uses for its lock control block structure. This value becomes the setting of the MLMT parameter for the IRLM address space procedure. The IRLM address space procedure sets the z/OS MEMLIMIT value for the address space.

Ensure that you set this value high enough so that IRLM does not reach the limit. The value that you choose should provide space for possible retained locks. IRLM only gets storage as it needs it, so choose a large value. You can also change the value dynamically by using the z/OS command MODIFY irlmproc,SET,PVT. See "Common service area" on page 30 for more information about how IRLM uses storage. For data sharing settings, see *DB2 Data Sharing: Planning and Administration*.

IRLM PROC parameter: MLMT

3. **LOCKS PER TABLE(SPACE)**

Acceptable values:	0 to 104857600
Default:	1000
Update:	option 18 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM NUMLKTS

Specify the default value for the maximum number of page, row, or LOB locks that a single application can hold simultaneously in a single table or tablespace before lock escalation occurs. You can enter the value in bytes, or use the abbreviations K for kilobytes and M for megabytes. The value that you specify for this field must be less than the value specified for LOCKS PER USER (except when LOCKS PER USER is set to 0).

This value becomes the default value (SYSTEM) for the LOCKMAX clause of the SQL statements CREATE TABLESPACE and ALTER TABLESPACE. A value of 0 indicates that there is no limit to the number of data and row locks that a program can acquire

Recommendation: Do not set the value to 0, because it can cause the IRLM to experience storage shortages.

For more information about this parameter, see Part 5 (Volume 2) of *DB2 Administration Guide*.

4. **LOCKS PER USER**

Acceptable values:	0 to 104857600
Default:	10000
Update:	option 18 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM NUMLKUS

Specify the maximum number of page, row, or LOB locks that a single application can hold concurrently for all table spaces. You can enter the value in bytes, or use the abbreviations K for kilobytes and M for megabytes. The maximum number includes locks on data pages, index pages, subpages, and rows that the program acquires when it accesses table spaces. The limit applies to all table spaces that are defined with the LOCKSIZE PAGE, LOCKSIZE ROW, or LOCKSIZE ANY options. A value of 0 indicates that there is no limit to the number of data and row locks that a program can acquire.

DB2 assumes that each lock requires 540 bytes of storage. If you define referential constraints between values, you might want to select a higher value for this field.

To avoid exhausting the IRLM's storage for locks, follow these guidelines:

- Do not specify 0 or a very large value unless it is specifically required to run an application.
- Consider the design of your applications. Long-running applications, particularly those that perform row-level locking, have few or infrequent commit points, or use repeatable-read insolation may use substantial amounts of lock storage. You should perform frequent commits to release locks.

Important: These values are constraints for a single application. Each concurrent application can hold the maximum number of locks specified here.

Check panel DSNTIPC to ensure that the required storage for the IRLM does not exceed the available region size for the IRLM.

For more information about the maximum number of locks that can be held, see Part 5 (Volume 2) of *DB2 Administration Guide*.

5. **DEADLOCK TIME**

Acceptable values:	1 to 5, 100 to 5000
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify the time, in seconds or milliseconds, of the local deadlock detection cycle. DB2 interprets values between 1 and 5 as seconds and values between 100 and 5000 as milliseconds. Depending on the value that you enter, IRLM might substitute a smaller maximum value. A *deadlock* is a situation where two or more requesters are waiting for resources that are held by another requester. *Deadlock detection* is the procedure by which a deadlock and its participants are identified.

IRLM PROC parameter: DEADLOK

6. **DEADLOCK CYCLE**

Acceptable values:	1
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify the number of local deadlock cycles that must expire before the IRLM performs global deadlock detection processing. This option is used only for DB2 data sharing.

IRLM PROC parameter: DEADLOK

7. **MEMBER IDENTIFIER**

Acceptable values:	1 to 255
Default:	1
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify an ID number that uniquely names this IRLM member within an IRLM data sharing group.

Recommendation: Correlate the IRLM member ID with the DB2 member name. For example, for DB2 member DSN1, specify an IRLM member ID of 1.

The IRLM ID that you specify does not relate directly to the limit of IRLM members that can be in the data sharing group. That limit is determined by the current hardware limits (32). If you edit the IRLMPROC directly, you can specify a value from 1 to 255. See *DB2 Command Reference* for information about the IRLMPROC command.

This option is used only for DB2 data sharing.

IRLM PROC parameter: IRLMID

8. **IRLM XCF GROUP NAME**

Acceptable values:	1 to 8 characters
Default:	DXRGROUP
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify the name of the IRLM group. This name must be different from the DB2 group name.

Recommendation: Begin this name with DXR. All members in the DB2 group must have the same IRLM XCF group name.

This option is used only for DB2 data sharing.

To avoid names that IBM uses for its cross-system coupling facility (XCF) groups, the first character must be an uppercase letter J-Z unless the name begins with DXR. Do not use SYS as the first three characters, and do not use UNDESIG as the group name.

IRLM PROC parameter: IRLMGRP

9. **LOCK ENTRY SIZE**

Acceptable values:	2, 4, 8
Default:	2
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify the initial size, in bytes, of individual lock entries in the lock table portion of the lock structure.

Recommendation: If you have less than seven members in your data sharing group, use the default value for most efficient use of coupling lock structure space, use the default value. If you have more than seven members, use a value of 4 to avoid an automatic rebuild for maximum connections.

DB2 converts the value for LOCK ENTRY SIZE to a corresponding value for the IRLM parameter MAXUSRS as shown in Table 42.

Table 42. Converting lock entry size to MAXUSRS values

LOCK ENTRY SIZE	MAXUSRS value
2	7
4	23
8	32

10. **NUMBER OF LOCK ENTRIES**

Acceptable values:	1 to 1024
Default:	0
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify the number of lock table entries that you want in the coupling facility lock structure. This value must be a power of 2 in the range of 1 to 1024, with each increment representing 1 048 576 lock table entries. The default value, 0, indicates that IRLM is to determine the number of lock table entries based on the lock structure size that is specified in CFRM policy and the number of users (MAXUSRS). If you specify a value for lock structure size in CFRM policy that is greater than 1024 megabytes, IRLM limits the number of lock table entries to a maximum of 1024 megabytes. If you want to control the number of lock table entries, enter a non-zero value within the accepted range.

The number of lock table entries has a direct affect on cross-system extended services (XES) contention. You should therefore monitor this number to find the optimum values for your installation.

IRLM PROC parameter: LTE

11. **DISCONNECT IRLM**

Acceptable values:	YES, NO
Default:	YES
Update:	edit IRLM start procedure
DSNZP*xxx*:	none

Specify whether IRLM should disconnect automatically from the data sharing group when DB2 is not identified to it. The default, YES, causes IRLM to disconnect from the data sharing group when DB2 is stopped normally or stops as the result of a DB2 failure.

When you specify YES is conjunction with AUTOSTART YES (on panel DSNTIPI), manual intervention is not required to stop IRLM.

If you specify NO, IRLM remains connected to the data sharing group even when DB2 is stopped. In this case, you must explicitly stop IRLM to bring it down.

NODISCON has less impact on other systems when a DB2 fails because z/OS is not required to perform certain recovery actions that it normally performs when IRLM comes down. NODISCON can also mean that DB2 restarts more quickly after a DB2 normal or abnormal termination because it does not have to wait for IRLM to rejoin the IRLM data sharing group.

IRLM PROC parameter: SCOPE

Protection panel: DSNTIPP

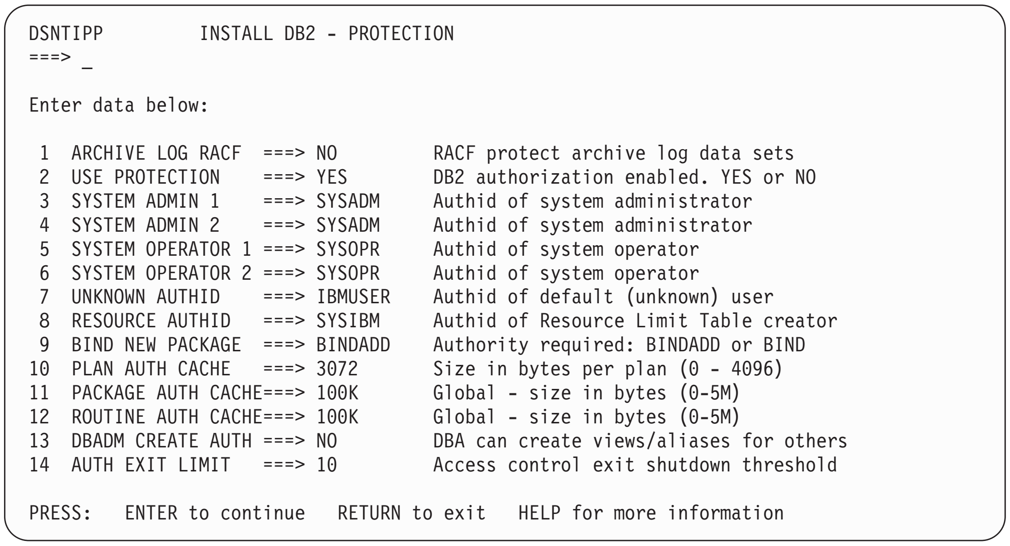
The entries on this panel are related to security matters. You should protect data sets with a security subsystem, such as Resource Access Control Facility (RACF), rather than with passwords.

If the data sets are managed by Data Facility Storage Management Subsystem (DFSMS), the password does not apply; data sets that are defined to DFSMS should be protected by RACF or some similar external security system.

Updating the parameters: If you are migrating, DB2 UDB for z/OS Version 8 uses your Version 7 catalog, directory, work file databases, BSDS, active logs, and archive logs. Consequently, you cannot change the passwords for those objects during migration.

Change passwords on panel DSNTIPP with the ALTER command of access method services. Then run the change log inventory utility, DSNJU003, to tell DB2 the new passwords.

You can change other entries by following an update process after migration. See "Main panel: DSNTIPA1" on page 94.



```
DSNTIPP         INSTALL DB2 - PROTECTION
===> _

Enter data below:

 1  ARCHIVE LOG RACF  ===> NO         RACF protect archive log data sets
 2  USE PROTECTION    ===> YES        DB2 authorization enabled. YES or NO
 3  SYSTEM ADMIN 1    ===> SYSADM     Authid of system administrator
 4  SYSTEM ADMIN 2    ===> SYSADM     Authid of system administrator
 5  SYSTEM OPERATOR 1 ===> SYSOPR     Authid of system operator
 6  SYSTEM OPERATOR 2 ===> SYSOPR     Authid of system operator
 7  UNKNOWN AUTHID    ===> IBMUSER    Authid of default (unknown) user
 8  RESOURCE AUTHID   ===> SYSIBM     Authid of Resource Limit Table creator
 9  BIND NEW PACKAGE  ===> BINDADD    Authority required: BINDADD or BIND
10  PLAN AUTH CACHE   ===> 3072       Size in bytes per plan (0 - 4096)
11  PACKAGE AUTH CACHE===> 100K       Global - size in bytes (0-5M)
12  ROUTINE AUTH CACHE===> 100K       Global - size in bytes (0-5M)
13  DBADM CREATE AUTH ===> NO         DBA can create views/aliases for others
14  AUTH EXIT LIMIT   ===> 10         Access control exit shutdown threshold

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

Figure 27. Protection panel: DSNTIPP

1. **ARCHIVE LOG RACF**

Acceptable values:	YES, NO
Default:	NO
Update:	see 194; not during migration
DSNZP*xxx*:	DSN6ARVP PROTECT

Specify whether archive log data sets are to be protected with individual profiles with RACF when they are created. If you specify YES, RACF protection must be active for DB2. However, a value of YES also means that you cannot use RACF generic profiles for archive log data sets. In addition, RACF class TAPEVOL must be active if your archive log is on tape. Otherwise, the offload fails. For information about using RACF, see Part 3 (Volume 1) of *DB2 Administration Guide*.

2. **USE PROTECTION**

Acceptable values:	YES, NO
Default:	YES
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AUTH

Specify whether DB2 is to perform authorization checking. If you specified YES for the value of CACHE DYNAMIC SQL on panel DSNTIP4, you must also specify YES for this value.

Recommendation: Specify YES. Specifying NO disables all authorization checking in DB2 and disables the GRANT statement. In this case, every privilege is granted to PUBLIC.

3. **SYSTEM ADMIN 1**

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSADM

Specify the first of two authorization IDs with installation SYSADM authority. The two users with installation SYSADM authority are permitted access to DB2 in all cases. For the implications of this authority, and to understand REVOKE implications when changing this field, see Part 3 (Volume 1) of *DB2 Administration Guide*.

4. **SYSTEM ADMIN 2**

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSADM
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSADM2

Specify the second of two authorization IDs with installation SYSADM authority; see field 8.

If you leave this field blank, the value is set to the value of field 3.

5. **SYSTEM OPERATOR 1**

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSOPR
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSOPR1

Specify the first of two authorization IDs with installation SYSOPR authority. The two users with installation SYSOPR authority are permitted access to DB2 even if the DB2 catalog is unavailable. For the implications of this authority, see Part 3 (Volume 1) of *DB2 Administration Guide*.

If blank, the value is set to the value of field 3.

Recommendation: Set the value of this field or the value of the SYSTEM
OPERATOR 2 field to SYSOPR. Doing so ensures that DB2 commands that are
issued from the console can be processed correctly when the DB2 catalog is
unavailable.

6. **SYSTEM OPERATOR 2**

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	SYSOPR
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SYSOPR2

Specify the second of two system operators with installation SYSOPR authority; see field 10.

If blank, the value is set to the value of field 5.

Recommendation: Set the value of this field or the value of the SYSTEM
OPERATOR 1 field to SYSOPR. Doing so ensures that DB2 commands that are
issued from the console can be processed correctly when the DB2 catalog is
unavailable.

7. **UNKNOWN AUTHID**

Acceptable values:	1 to 8 characters, the first of which must be an alphabetic character
Default:	IBMUSER
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DEFLTID

Specify the authorization ID that is to be used if RACF is not available for batch access and USER= is not specified in the job statement.

8. **RESOURCE AUTHID**

Acceptable values:	1 to 8 characters
Default:	SYSIBM
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP RLFAUTH

Specify the authorization ID used if you plan to use the resource limit facility (governor).

9. **BIND NEW PACKAGE**

Acceptable values:	BINDADD, BIND
Default:	BINDADD
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM BINDNV

Specify whether BIND or BINDADD authority is required to bind a new version of an existing package. If you accept the default, BINDADD, you allow only users with the BINDADD system privilege to create a new package. If you specify BIND, you allow users with the BIND privilege on a package or collection to create a new version of an existing package when they bind it. You also allow users with PACKADM authority to add a new package or a new version of a package to a collection.

See Part 3 (Volume 1) of *DB2 Administration Guide* for a full description of the privileges that are needed to bind a new package.

10. **PLAN AUTH CACHE**

Acceptable values:	0 to 4096 in multiples of 256
Default:	3072
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AUTHCACH

Specify the size of the authorization cache that is to be used if no CACHESIZE is specified on the BIND PLAN subcommand. Choose 0 if you do not want to use an authorization cache. For an authorization cache, you need 32 bytes of overhead plus (8 bytes of storage * number of concurrent users). See Part 3 (Volume 1) of *DB2 Administration Guide* for more information about cache size.

11. **PACKAGE AUTH CACHE**

Acceptable values:	0 to 5M
Default:	100K
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHEPAC

Specify how much storage to allocate for the caching of package authorization information for all packages on this DB2 member. The cache is stored in the DSN1DBM1 address space.

12. **ROUTINE AUTH CACHE**

Acceptable values:	0-5M
Default:	100K
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM CACHERAC

Specify how much storage to allocate for the caching of routine authorization information for all routines on this DB2 member. Routines include stored procedures and user-defined functions.

13. **DBADM CREATE AUTH**

Acceptable values:	NO, YES
Default:	NO
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM DBACRVW

Specify whether an authorization ID with DBADM authority on a database can:

- Create a view for another authorization ID on tables in that database.
- Create a materialized query table or alter a table to become a materialized query table for another authorization ID, assuming that DBADM authority is held on the database in which the tables of the fullselect reside and that the authorization ID has DBADM authority on the database in which the materialized query table is to reside.
- Create an alias for itself or another authorization ID for a table in that database.

If you specify YES, an authorization ID with DBCTRL authority on a database can also create an alias for itself or for another authorization ID for a table in that database.

Specifying YES results in less need for SYSADM authority on a database. However, users that need full authority may still need to have SYSADM authority. Specifying YES does not allow an authorization ID with DBADM authority to grant authority on that view.

14. **AUTH EXIT LIMIT**

Acceptable values:	0 to 32767
Default:	10
Update:	option 19 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM AEXITLIM

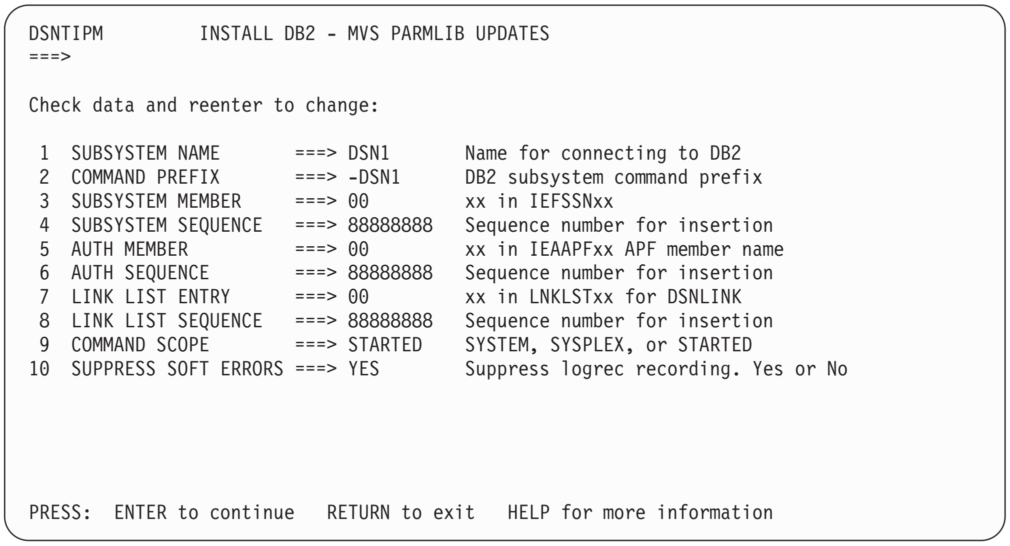
Specify the number of abends of the DB2 access control authorization exit routine that will be tolerated before it is shut down. After the exit routine shuts down, it can be reactivated only by restarting DB2. A very low setting may cause the exit routine to shut down in response to routine abends such as time-outs. A very high setting, on the other hand, may mask a problem with the exit routine environment that can result in degraded DB2 or system performance.

MVS PARMLIB updates panel: DSNTIPM

The entries on this panel produce the DSNTIJMV job that defines DB2 to z/OS and updates the following PARMLIB members:

- IEFSSN*xx*, to define DB2 and IRLM as formal z/OS subsystems
- IEAAPF*xx*, to authorize the *prefix*.SDSNLOAD, *prefix*.SDSNLINK, and *prefix*.SDSNEXIT libraries
- LNKLST*xx*, to include the *prefix*.SDSNLINK library.

Updating the parameters: Different sites have different requirements for identifying DB2 to z/OS; as a result, the updates that DSNTIJMV makes to z/OS PARMLIB members might be incomplete. To ensure that the updates are complete, it is recommended that you edit the z/OS PARMLIB members directly when you install or migrate DB2. This is substantially easier than editing DSNTIJMV.



```
DSNTIPM         INSTALL DB2 - MVS PARMLIB UPDATES
===>

Check data and reenter to change:

 1  SUBSYSTEM NAME       ===> DSN1       Name for connecting to DB2
 2  COMMAND PREFIX       ===> -DSN1      DB2 subsystem command prefix
 3  SUBSYSTEM MEMBER     ===> 00         xx in IEFSSNxx
 4  SUBSYSTEM SEQUENCE   ===> 88888888   Sequence number for insertion
 5  AUTH MEMBER          ===> 00         xx in IEAAPFxx APF member name
 6  AUTH SEQUENCE        ===> 88888888   Sequence number for insertion
 7  LINK LIST ENTRY      ===> 00         xx in LNKLSTxx for DSNLINK
 8  LINK LIST SEQUENCE   ===> 88888888   Sequence number for insertion
 9  COMMAND SCOPE        ===> STARTED    SYSTEM, SYSPLEX, or STARTED
10  SUPPRESS SOFT ERRORS ===> YES        Suppress logrec recording. Yes or No




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

Figure 28. MVS PARMLIB updates panel: DSNTIPM

1. **SUBSYSTEM NAME**

#	Acceptable values:	1 to 4 characters, the first of which must be A-Z, #, $, or @.
#		Others must be A-Z, 0-9, #, $, or @.
	Default:	DSN1
	Update:	see below
	DSNHDECP:	SSID

Specify the z/OS subsystem name for DB2. The name is used in member IEFSSN*xx* of SYS1.PARMLIB.

If you specified a group attachment name in the GROUP ATTACH field on panel DSNTIPK, DSNHDECP.SSID is set using that value.

Updating the parameters: Different sites have different requirements for identifying DB2 to z/OS; as a result, the updates that DSNTIJMV makes to z/OS PARMLIB members might be incomplete. To ensure that the updates are complete, it is recommended that you edit the z/OS PARMLIB members directly when you install or migrate DB2. This is substantially easier than editing DSNTIJMV.

2. **COMMAND PREFIX**

Acceptable values:	1 to 8 characters, the first of which must be a non-alphanumeric character.
Default:	-DSN1 (hyphen, concatenated with subsystem name)
Update:	see 199
DSNZP*xxx*:	none

Specify the DB2 command prefix. When the prefix appears at the beginning of a command that is entered at an z/OS operator's console, z/OS passes the command to DB2 for processing. The command prefix is used in the DB2 entry of member IEFSSN*xx* of SYS1.PARMLIB.

The first character of the command prefix must be a character in Table 43. The remaining characters of the command prefix must be from Table 43, A-Z, or 0-9.

Table 43. Allowable special characters for the command prefix

Name	Character	Hexadecimal representation
cent sign	¢	X'4A'
period	.	X'4B'
less-than sign	<	X'4C'
plus sign	+	X'4E'
vertical bar	|	X'4F'
ampersand [1]	&	X'50'
exclamation point	!	X'5A'
dollar sign	$	X'5B'
asterisk	*	X'5C'
right parenthesis	)	X'5D'
semi-colon	;	X'5E'
hyphen	-	X'60'
slash	/	X'61'
percent sign	%	X'6C'
underscore	_	X'6D'
question mark	?	X'6F'
colon	:	X'7A'
number sign	#	X'7B'
at sign	@	X'7C'
apostrophe [2]	'	X'7D'
equal sign[3]	=	X'7E'
quotation marks	"	X'7F'

1. To use the ampersand (&), accept the default in this field, and then edit job DSNTIJMV to specify the ampersand as the command prefix.
2. To use the apostrophe ('), you must code two consecutive apostrophes in your IEFSSNxx member. For example, the entry for subsystem DB2A with a command prefix of 'DB2A and a scope of started looks like this:
   ```
   DB2A,DSN3INI,'DSN3EPX,''DB2A,'
   ```
3. To use the equal sign (=), accept the default command prefix, and then edit job DSNTIJMV to replace the dash (–) with the equal sign as the first character of the command prefix.

Do not use the JES2 backspace character as a command prefix character. Do not assign a command prefix that is used by another subsystem or that can be interpreted as belonging to more than one subsystem or z/OS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix beginning from the first character. For example, you cannot assign '-' to one subsystem and '-DB2A' to another. Similarly, you cannot assign '?DB2' to one subsystem and '?DB2A' to another. However, you can assign ′-DB2A′ and ′-DB2B′ to different DB2 subsystems.

To use multiple-character command prefixes, have the system programmer update the IEFSSN*xx* subsystem definition statements in SYS1.PARMLIB. For more information about the SYS1.PARMLIB IEFSSN*xx* statement, see "DSNTIJMV updates to SYS1.PARMLIB" on page 253.

3. **SUBSYSTEM MEMBER**

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see 199
DSNZP*xxx*:	none

Specify the last two characters (*xx*) of the name of member IEFSSN*xx* of SYS1.PARMLIB. The subsystem member name indicates the available z/OS subsystems, including DB2 and IRLM.

4. **SUBSYSTEM SEQUENCE**

Acceptable values:	1 to 99999995
Default:	88888888
Update:	see 199
DSNZP*xxx*:	none

Specify any number that is greater than the highest sequence number that is already used in the IEFSSN*xx* PARMLIB member.

5. **AUTH MEMBER**

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see 199
DSNZP*xxx*:	none

Specify the last two characters (*xx*) of the name of member IEAAPF*xx* of SYS1.PARMLIB. This member is used for authorized program facility (APF) authorization of the *prefix*.SDSNLOAD, *prefix*.SDSNLINK, and *prefix*.SDSNEXIT libraries. This data set must be APF-authorized. The member name must currently exist for the z/OS update job DSNTIJMV to work correctly.

You can use the PROG*xx* member instead of the IEAAPF*xx* member. In this case, you must manually name the PROG*xx* member because job DSNTIJMV does not do it for you.

6. **AUTH SEQUENCE**

Acceptable values:	1-99999995
Default:	88888888
Update:	see 199
DSNZP*xxx*:	none

Specify any number that is greater than the highest sequence number that is already used in the IEAAPF*xx* PARMLIB member.

7. **LINK LIST ENTRY**

Acceptable values:	2 alphanumeric characters
Default:	00
Update:	see 199
DSNZP*xxx*:	none

Specify the last two characters of LNKLST*xx* as needed to include the *prefix*.SDSNLINK library.

8. **LINK LIST SEQUENCE**

Acceptable values:	1 to 99999999
Default:	88888888
Update:	see 199
DSNZP*xxx*:	none

Specify any number that is greater than the highest sequence number that is already used in the LNKLST*xx* PARMLIB member.

9. **COMMAND SCOPE**

Acceptable values:	SYSTEM, SYSPLEX, STARTED
Default:	STARTED
Update	see "Installation step 18: Bind the packages for DB2 REXX Language Support: DSNTIJRX (optional)" on page 278
DSNZP*xxx*:	none

Specify the scope of the command prefix.

SYSTEM The scope of commands is for one z/OS system. The command prefix is registered at z/OS IPL.

SYSPLEX The scope of commands is for the entire Sysplex. The command prefix is registered at z/OS IPL.

STARTED The scope of commands is for the entire Sysplex. The command prefix is registered at DB2 startup and deregistered when DB2 stops.

Although STARTED specifies a Sysplex scope, you can also use it for a DB2 subsystem in a non-data-sharing environment. Use STARTED if you intend to use the z/OS automatic restart manager, or if you might move this DB2 into a data sharing group.

10. **SUPPRESS SOFT ERRORS**

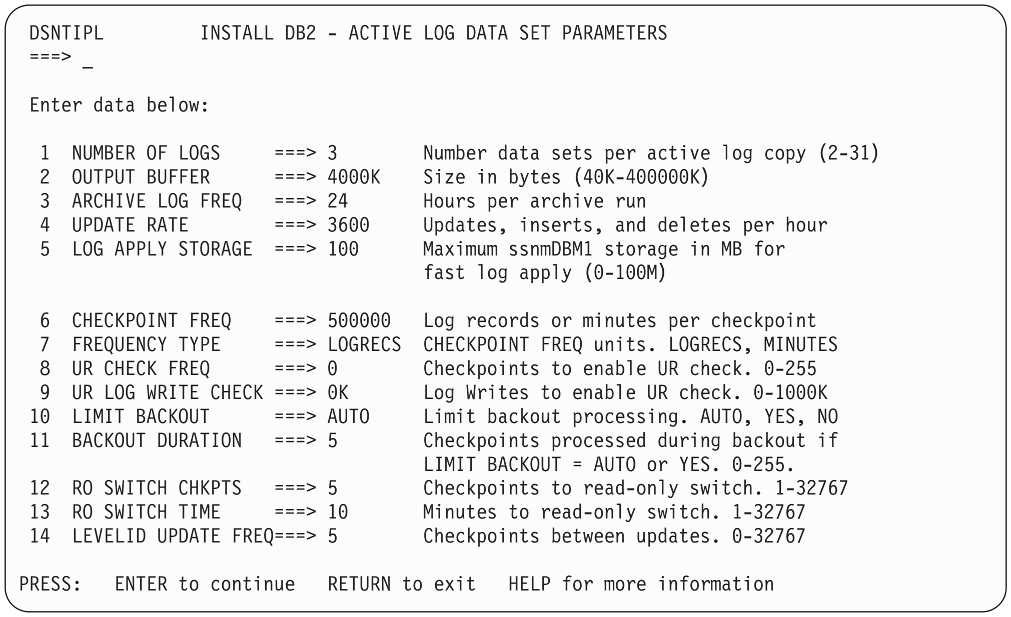
Acceptable values:	YES, NO
Default:	Yes
Update:	option 20 on panel DSNTIPB
DSNZP*xxx*:	SUPERRS

Specify whether to record errors such as invalid decimal data and arithmetic exceptions. DB2 catches these errors and issues SQLCODEs for them. This option enables or disables the recording of these errors in the operating system data set, SYS1.LOGREC.

Active log data set parameters: DSNTIPL

The entries on this panel define characteristics of active log data sets.

Performance note: Several fields on this panel affect the DB2 use of logging. Be careful when determining the values that are associated with fields on this panel. These values can greatly affect the performance of your DB2 subsystem. For a description of DB2 logging, see Part 4 (Volume 1) of *DB2 Administration Guide*.



```
DSNTIPL         INSTALL DB2 - ACTIVE LOG DATA SET PARAMETERS
===> _

Enter data below:

 1  NUMBER OF LOGS     ===> 3        Number data sets per active log copy (2-31)
 2  OUTPUT BUFFER      ===> 4000K    Size in bytes (40K-400000K)
 3  ARCHIVE LOG FREQ   ===> 24       Hours per archive run
 4  UPDATE RATE        ===> 3600     Updates, inserts, and deletes per hour
 5  LOG APPLY STORAGE  ===> 100      Maximum ssnmDBM1 storage in MB for
                                     fast log apply (0-100M)

 6  CHECKPOINT FREQ    ===> 500000   Log records or minutes per checkpoint
 7  FREQUENCY TYPE     ===> LOGRECS  CHECKPOINT FREQ units. LOGRECS, MINUTES
 8  UR CHECK FREQ      ===> 0        Checkpoints to enable UR check. 0-255
 9  UR LOG WRITE CHECK ===> 0K       Log Writes to enable UR check. 0-1000K
10  LIMIT BACKOUT      ===> AUTO     Limit backout processing. AUTO, YES, NO
11  BACKOUT DURATION   ===> 5        Checkpoints processed during backout if
                                     LIMIT BACKOUT = AUTO or YES. 0-255.
12  RO SWITCH CHKPTS   ===> 5        Checkpoints to read-only switch. 1-32767
13  RO SWITCH TIME     ===> 10       Minutes to read-only switch. 1-32767
14  LEVELID UPDATE FREQ===> 5        Checkpoints between updates. 0-32767

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

Figure 29. Log data sets panel: DSNTIPL

1. **NUMBER OF LOGS**

Acceptable values:	2 to 31 (2 to 93 if the BSDS has been converted)
Default:	3
Update:	cannot change during update or migration
DSNZP*xxx*:	none

| Specify the number of data sets for each copy of the active log. This value specifies
| the number of active log data sets that are established at installation time. You can
| later use the DSNJU003 utility to change the number of log data sets, and if you
| run the BSDS conversion utility, you can specify up to 93 data sets. If you use the
| DSNJU003 utility to modify the number of logs, your modified value is not
| reflected on this panel. See "Updating other parameters" on page 249 for details.
| For more information about running the BSDS conversion utility, see
| "Considerations for the BSDS" on page 361.

2. **OUTPUT BUFFER**

Acceptable values:	40K to 400000K
Default:	4000K
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP OUTBUFF

Specify the size of the output buffer that is used for writing active log data sets. You can enter the value in bytes (for example, 40960) or use the abbreviation K for kilobytes (for example, 40K). The larger the output buffer, the more likely a requested RBA can be found without a read request.

3. **ARCHIVE LOG FREQ**

Acceptable values:	1 to 200
Default:	24
Update:	cannot change during update
DSNZP*xxx*:	none

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

4. **UPDATE RATE**

Acceptable values:	1 to 16 777 216
Default:	3600
Update:	cannot change during update
DSNZP*xxx*:	none

Estimate the average number of inserts, updates, and deletes that are expected per hour in your subsystem.

You can use K (as in 32K) for multiples of 1024 bytes and M (as in 16M) for multiples of 1 048 576 bytes.

The size calculations in the DSNTINST CLIST assume that about 400 bytes of data are logged for each insert, update, and delete. The amount of data that is logged for these changes might be different at your site. Therefore, consider changing the size of the log data sets after you gain some experience with DB2 and have a better idea of how many bytes of data are logged for each change. Generally, if you have a subsystem that is tuned for maximum efficiency, you can expect to log about 10 GB of data per hour while processing several millions of updates and inserts.

Together, the UPDATE RATE and the ARCHIVE LOG FREQ (field 3) determine the size of the active logs.

5. **LOG APPLY STORAGE**

	Acceptable values:	0 to 100M
|	Default:	100M
|	Update:	option 21 on panel DSNTIPB
	DSNZP*xxx*:	DSN6SYSP LOGAPSTG

The value in this field represents the maximum *ssnm*DBM1 storage that can be used by the fast log-apply process. If you specify 0, the fast log-apply process is disabled except during DB2 restart. During DB2 restart, the fast log-apply process is always enabled.

Recommendation: Specify 10 MB of log apply storage for each concurrent RECOVER job that you want to have faster log apply processing. The default value of 100 MB provides log apply storage for 10 concurrent RECOVER jobs.

6. **CHECKPOINT FREQ**

Acceptable values:	200 to 16 000 000 (log records) or 1 to 60 (minutes)
Default:	500 000
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP CHKFREQ

Specify the system checkpoint frequency in minutes or in number of log records. If you have widely variable logging rates, maximize system performance by specifying the checkpoint frequency in time. DB2 starts a new checkpoint at the interval you specify, either in minutes, or in the number of log records.

You can use the SET LOG command to dynamically change the number of log records between checkpoints.

Recommendation: If your primary concern is DB2 restart time, use a checkpoint frequency between 50 000 and 1 000 000 log records. Otherwise, use a checkpoint frequency of 2 to 5 minutes.

7. **FREQUENCY TYPE**

Acceptable values:	LOGRECS or MINUTES
Default:	LOGRECS
Update:	cannot change during update
DSNZP*xxx*:	none

Specify whether the units for checkpoint frequency are in log records or time. If you choose LOGRECS, specify the number of records in the CHECKPOINT FREQ field. If you choose MINUTES, specify the number of minutes.

8. **UR CHECK FREQ**

Acceptable values:	0 to 255
Default:	0
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP URCHKTH

Specify the number of checkpoint cycles that are to complete before DB2 issues a warning message to the console and instrumentation for an uncommitted unit of recovery (UR).

Accept the default to disable this option. This option does not affect performance. If you use this option, specify a value that is based on how often a checkpoint occurs in your system and how much time you can allow for a restart or shutdown. For example, if your site's checkpoint interval is 5 minutes and the standard limit for issuing commits with units of recovery is 20 minutes, divide 20 by 5 to determine the best value for your system.

9. **UR LOG WRITE CHECK**

Acceptable values:	0 to 1000K
Default:	0
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP URLGWTH

Specify the number of log records that are to be written by an uncommitted unit of recovery (UR) before DB2 issues a warning message to the console. The purpose of this option is to provide notification of a long-running UR. Long-running URs might result in a lengthy DB2 restart or a lengthy recovery situation for critical tables. Specify the value in 1-K (1000 log records) increments. A value of 0 indicates that no write check is to be performed.

10. **LIMIT BACKOUT**

Acceptable values:	AUTO, YES, NO
Default:	AUTO
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP LBACKOUT

Specify whether to postpone some backward-log processing.

NO specifies that DB2 backward-log processing should process all inflight and in-abort units of recovery (URs). YES postpones back-out processing for some units of work until you issue the command RECOVER POSTPONED. AUTO postpones some backout processing, but automatically starts the back-out processing when DB2 restarts and begins acceptance of new work. With YES or AUTO, back-out processing runs concurrently with new work. sets or partitions with pending back-out work are unavailable until their back-out work is complete.

See the description of the BACKOUT DURATION field for information on how you specify the amount of backward-log processing that is allowed to complete during restart (when LIMIT BACKOUT is YES or AUTO).

11. **BACKOUT DURATION**

Acceptable values:	0 to 255
Default:	5
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP BACKODUR

Specify a multiplier to indicate how much log to process for backout when LIMIT BACKOUT=YES or AUTO.

During restart, backward-log processing continues until both of the following events occur:

- All inflight and in-abort URs with update activity against the catalog or directory are backed out.
- The number of log records processed is equal to the number you specify in BACKOUT DURATION multiplied by the number of log records per checkpoint, which you specified in field CHECKPOINT FREQ. If the checkpoint frequency type is minutes, the default value of 50 000 log records is used to calculate the number of log records to process.

Inflight and in-abort URs that are not completely backed out during restart are converted to postponed-abort status. sets or partitions with postponed backout work are put into restart-pending (RESTP). This state blocks all access to the object other than access by the RECOVER POSTPONED command state or by automatic back-out processing that is performed by DB2 when LIMITED BACKOU =AUTO.

A table space might be in restart-pending mode, without the associated index spaces also in restart-pending mode. This happens if a postponed-abort UR makes updates only to non-indexed fields of a table in a table space. In this case, the indexes are accessible to SQL (for index-only queries), even though the table space is inaccessible.

12. **RO SWITCH CHKPTS**

Acceptable values:	1 to 32767
Default:	5 (checkpoints)
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP PCLOSEN

Indicates the number of consecutive DB2 checkpoints since a set or partition was last updated, after which DB2 converts the set or partition from read-write to read-only. This value is used in conjunction with RO SWITCH TIME. If the condition for RO SWITCH TIME or RO SWITCH CHKPTS is met, the set or partition is converted from read-write to read-only.

Having DB2 switch an infrequently updated set from read-write to read-only can be a performance benefit for recovery, logging, and for data sharing processing. See Part 5 (Volume 2) of *DB2 Administration Guide* and *DB2 Data Sharing: Planning and Administration* for more information about read-only switching.

13. **RO SWITCH TIME**

Acceptable values:	1 to 32767
Default:	10 (minutes)
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP PCLOSET

Indicates the number of minutes since a set or partition was last updated, after which DB2 converts the set or partition from read-write to read-only. This value is used in conjunction with RO SWITCH CHKPTS. If the condition for RO SWITCH CHKPTS or RO SWITCH TIME is met, the set or partition is converted from read-write to read-only.

Having DB2 switch an infrequently updated set from read-write to read-only can be a performance benefit for recovery, logging, and for data sharing processing. See Part 5 (Volume 2) of *DB2 Administration Guide* and *DB2 Data Sharing: Planning and Administration* for more information about read-only switching.

14. **LEVELID UPDATE FREQ**

Acceptable values:	0 to 32767
Default:	5 (checkpoints)
Update:	option 21 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DLDFREQ

Controls how often, in the number of checkpoints, the level ID of a set or partition
# is to be updated. A zero (0) disables down-level detection. The level ID frequency
# also controls how often the value that the RECOVER LOGONLY utility uses as the
# starting point for log apply is updated. This value is updated at the same
# frequency (in checkpoints) as the level ID update frequency that you specify in this
# field.

Consider the following questions when you choose a value for the frequency of level ID updates:

- ***How often do you use backup and restore methods outside of DB2's control?*** (such as DSN1COPY or DFDSS dump and restore)? If you rarely use such methods, you do not need to update the level ID frequently.
- ***How many sets are open for update at the same time?*** If DB2 updates level IDs frequently, you have extra protection against down-level sets. However, if the level IDs for many sets must be set at every checkpoint, you might experience a performance degradation.
- ***How often does the subsystem take checkpoints?*** If your DB2 subsystem takes frequent system checkpoints, you can set the level ID frequency to a higher number.

Archive log data set parameters panel: DSNTIPA

The entries on this panel define the characteristics of archive log data sets.

Updating the parameters: You can update all the parameters on this panel by using their subsystem parameter name.

```
 DSNTIPA         INSTALL DB2 - ARCHIVE LOG DATA SET PARAMETERS
 ===> _
 Enter data below:

  1  ALLOCATION UNITS ===> BLK       Blk, Trk, or Cyl
  2  PRIMARY QUANTITY ===>           Primary space allocation
  3  SECONDARY QTY. ===>           Secondary space allocation
  4  CATALOG DATA     ===> NO        YES or NO to catalog archive data sets
  5  DEVICE TYPE 1    ===> TAPE      Unit name for COPY1 archive logs
  6  DEVICE TYPE 2    ===>           Unit name for COPY2 archive logs
  7  BLOCK SIZE       ===> 24576     Rounded up to 4096 multiple
  8  READ TAPE UNITS  ===> 2         Maximum allocated read tape units
  9  DEALLOC PERIOD   ===> 0         Time interval to deallocate tape units
 10  RECORDING MAX    ===> 1000      Number of data sets recorded in BSDS
 11  WRITE TO OPER    ===> YES       Issue WTOR before mount for archive
 12  WTOR ROUTE CODE  ===> 1,3,4
                                     Routing codes for archive WTORs
 13  RETENTION PERIOD ===> 9999      Days to retain archive log data sets
 14  QUIESCE PERIOD   ===> 5         Maximum quiesce interval (1-999)
 15  COMPACT DATA     ===> NO        YES or NO for data compaction
 16  SINGLE VOLUME    ===> NO        Single volume for disk archives. NO or YES
PRESS:   ENTER to continue   RETURN to exit   HELP for more information
```

Figure 30. Archive log data sets panel: DSNTIPA

1. **ALLOCATION UNITS**

Acceptable values:	BLK, TRK, or CYL
Default:	BLK
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ALCUNIT

Specify the units in which primary and secondary space allocations are to be obtained.

2. **PRIMARY QUANTITY**

Acceptable values:	blank or 1 to 999999
Default:	blank
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP PRIQTY

Specify the primary space allocation for a disk data set, in units of ALCUNIT. If you use the default, a blank, the CLIST calculates this space using block size and size of the log. DFSMS Direct Access Device Space Management (DADSM) limits the space allocation on a single volume to less than 64000 tracks. Therefore, if the archive log data set size can be greater than or equal to 64000 tracks, you need to specify a primary space quantity of less than 64000 tracks. This forces the archive log data set to extend to a second volume. For more information about estimating the archive log data set size, see "Archive log data sets storage requirements" on page 27.

3. **SECONDARY QTY.**

Acceptable values:	blank or 1 to 999999
Default:	blank
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP SECQTY

Specify the secondary space allocation for a disk data set, in units of ALCUNIT. If you use the default, a blank, the CLIST calculates this space using block size and size of the log.

4. **CATALOG DATA**

Acceptable values:	YES, NO
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP CATALOG

Specify whether archive log data sets are to be cataloged in the primary catalog of the ICF. This option is only meaningful if you specify tape for the DEVICE TYPE 1 or DEVICE TYPE 2 fields on this panel because DB2 requires that all archive log data sets allocated on disk be cataloged. If you choose to archive to disk, the catalog option must be set to YES. If the catalog option is set to NO and you decide to place your archive log data sets on disk, you receive message DSNJ072E each time an archive log data set is allocated, and the DB2 subsystem catalogs the data set.

5. **DEVICE TYPE 1**

Acceptable values:	device type or unit name
Default:	TAPE
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP UNIT

Specify the device type or unit name for storing archive log data sets. The value can be any alphanumeric string. If you choose to archive to disk, you can specify a generic device type with a limited volume range. DB2 requires that all archive log data sets allocated on disk be cataloged. If the device type specifies disk, set field 4 (CATALOG DATA) to YES.

If the unit name specifies disk, the archive log data sets can extend to a maximum of 15 volumes. If the unit name specifies a tape device, DB2 can extend to a maximum of 20 volumes. If you chose to use disk, make the primary and secondary space allocations (fields 2 and 3) large enough to contain all of the data that comes from the active log data sets without extending beyond 15 volumes.

When archiving to disk, DB2 uses the number of online storage volumes for the specified UNIT name to determine a count of candidate volumes, up to a maximum of 15 volumes. If the archives are to be managed by SMS, do not use a storage class with the Guaranteed Space attribute. SMS attempts to allocate a primary extent on every candidate volume. This can result in allocation failures or unused space because the primary extent on each unused volume is not released when the archive data set is closed.

For information about single volume archives, see “Installation step 4: Define DB2 initialization parameters: DSNTIJUZ” on page 259. For information about using SMS to archive log data sets, see *DB2 Administration Guide*.

6. **DEVICE TYPE 2**

Acceptable values:	device type or unit name
Default:	none
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP UNIT2

Specify the device type or unit name for storing the second copy of archive log data sets (COPY2 data sets), as for field 5.

7. **BLOCK SIZE**

Acceptable values:	8192 to 28672
Default:	24576
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP BLKSIZE

Specify the block size of the archive log data set. The block size must be compatible with the device type you use for archive logs. The value is rounded up to the next multiple of 4096 bytes. You can also enter the value with a K; for example, 28K.

If the archive log is written to tape, using the largest possible block size improves the speed of reading the archive logs. Use Table 44 as a guide.

Table 44. Recommended block size values

Archive log device	Block size
Tape	28672
3380	20480
3390 or RAMAC	24576

8. **READ TAPE UNITS**

Acceptable values:	1 to 99
Default:	2
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP MAXRTU

Specify the maximum number of dedicated tape units that can be allocated to read archive log tape volumes concurrently. This installation option, along with DEALLOC PERIOD, allows DB2 to optimize archive log reading from tape devices.

In a data sharing environment, the archive tape is not available to other members of the group until the deallocation period expires. You might not want to use this option in a data sharing environment unless all recover jobs are submitted from the same member.

Recommendation: Set the READ TAPE UNITS value to be at least one less than the number of tape units available to DB2. If you do otherwise, the OFFLOAD process could be delayed, which would affect the performance of your DB2 subsystem. For maximum throughput during archive log processing, specify the largest value possible for this option, remembering that you need at least one tape unit for offload processing. You can override this value by using the SET ARCHIVE command.

9. **DEALLOC PERIOD**

Acceptable values:	see below
Default:	0
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP DEALLCT

Specify the length of time an archive read tape unit is allowed to remain unused before it is deallocated. You can specify:

- minutes, seconds (blank or 0 to 59, blank or 1-1439)
- 1440 (minutes)
- NOLIMIT

Specifying NOLIMIT allows maximum optimization opportunities.

Example: Assume that you want the deallocation period to be 30 seconds. Enter `0,30`.

Example: Assume that you want the deallocation period to be 23 minutes and 47 seconds. Enter `23,47`.

Recommendation: If your archive log data is on tape, set this value high enough to allow DB2 to optimize tape handling for multiple read applications. When all tape reading is complete, you can update this option with the SET ARCHIVE command.

10. **RECORDING MAX**

Acceptable values:	10 to 10000 (see restriction below if BSDS is not converted)
Default:	1000
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6LOGP MAXARCH

Specify the maximum number of archive log volumes that are to be recorded in the BSDS. When this number is exceeded, recording resumes at the beginning of the BSDS. The maximum number of volumes is dependent on the conversion of the BSDS. The BSDS conversion process is run after DB2 is in new-function mode. If you have converted the BSDS, the maximum is 10000 volumes. If you have not converted the BSDS, and a value greater than 1000 is entered in this field, DB2 resets the value to 1000 and issues message DSNJ155I at DB2 startup. For more information about converting the BSDS, see “Considerations for the BSDS” on page 361.

You must create image copies of all DB2 objects, probably several times, before the archive log data sets are discarded. If you fail to retain an adequate number of archive log data sets for all the image copies, you might need to cold start or reinstall DB2. In both cases, data is lost.

For information about managing the log and determining how long to keep the logs, refer to Part 4 (Volume 1) of *DB2 Administration Guide*.

11. **WRITE TO OPER**

Acceptable values:	YES, NO
Default:	YES
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCWTOR

Specify whether to send a message to the operator and wait for an answer before attempting to mount an archive log data set. Other DB2 users can be forced to wait while the mount is pending. They are not affected while DB2 is waiting for a response to the message.

Specify NO if you use a device that does not have long delays for mounts. Specify YES if you use a device for storing archive log data sets, such as tape, that requires long delays for mounts. Field 5 (DEVICE TYPE 1) specifies the device type or unit name.

12. **WTOR ROUTE CODE**

Acceptable values:	see below
Default:	1,3,4
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCWRTC

Specify the list of route codes from messages from the archive log data sets to the operator. You can specify from 1 to 16 route codes. Separate numbers in the list by commas only, not by blanks.

For descriptions of the routing codes, see *z/OS MVS System Codes*. The routing codes are also discussed in the description of the WTO macro in *z/OS MVS Programming: Assembler Services Reference, Volumes 1 and 2*.

13. **RETENTION PERIOD**

Acceptable values:	0 to 9999
Default:	9999
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP ARCRETN

Specify the number of days that DB2 is to retain archive log data sets. The retention period is often used in tape management systems to control the reuse and scratching of data sets and tapes. DB2 uses the value as the value for the dynamic allocation parameter DALRETPD when archive log data sets are created.

The retention period set by the DFSMSdfp storage management subsystem (SMS) can be overridden by this DB2 parameter. Typically, the retention period is set to the smaller value that is specified by either DB2 or SMS. The storage administrator and database administrator should agree on a retention period value that is appropriate for DB2.

The retention period is added to the current date to calculate the expiration date.

Important: Due to the wide variety of tape management systems and the opportunity for external manual overrides of retention periods, DB2 does not have an automated method to delete the archive log data sets from the BSDS inventory of archive log data sets. Therefore, the information about an archive log data set might be in the BSDS long after it has been scratched by a tape management system, after its retention period expired. Conversely, the maximum number of archive log data sets can have been exceeded (see field 8), and the data from the BSDS can be dropped long before the data set has reached its expiration data.

For information about the archive log data sets and how they can be managed by using the change log inventory utility (DSNJU003), refer to Part 4 (Volume 1) of *DB2 Administration Guide*.

14. **QUIESCE PERIOD**

Acceptable values:	1 to 999
Default:	5
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP QUIESCE

Specify the maximum amount of time, in seconds, that DB2 is allowed to attempt a full system quiesce.

This parameter requires some tuning. If you specify too short an interval, the quiesce period expires before a full quiesce is accomplished. If you specify too long an interval, the quiesce period might cause unnecessary DB2 lock contention and time-outs. For more information about the quiesce mode of the ARCHIVE LOG command, see Part 4 (Volume 1) of *DB2 Administration Guide*.

15. **COMPACT DATA**

Acceptable values:	YES or NO
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP COMPACT

Specify whether data that is written to archive logs should be compacted. This option only applies to data written to a 3480 or 3490 device that has the improved data recording capability (IDRC) feature. When this feature is turned on, hardware in the IDRC-compliant tape control unit writes data at a much higher density than normal, allowing for more data on a volume. Specify NO if you don't want your data compacted or do not use a 3480 or 3490 device with the IDRC feature. Specify YES if you use a 3480 or 3490 device with the IDRC feature and you want the data to be compacted.

If you use compression or auto-blocking on the tape unit, you need to ensure that you do not read backwards on the tape unit. You can do this by increasing the size and number of active log data sets and by monitoring long-running units of recovery with the UR CHECK FREQ (panel DSNTIPN) or another monitor. The alternative to monitoring the units of work and increasing active log space is archiving to disk and then using another facility, such as DFSMShsm to archive the archive log from disk to tape. Be aware that data that is compressed to tape can only be read with a device that supports the IDRC feature. This could be a concern when you send archive tapes to another site for remote recovery.

16. **SINGLE VOLUME**

Acceptable values:	YES or NO
Default:	NO
Update:	option 22 on panel DSNTIPB
DSNZP*xxx*:	DSN6ARVP SVOLARC

When archiving to disk, DB2 will use the number of online storage volumes for the specified UNIT name to determine a count of candidate volumes, up to a maximum of 15 volumes. Specify YES if you want DB2 to specify a unit and volume count of 1 when allocating a new archive log data set on disk. You may want to do this when SMS manages the archives.

Databases and spaces to start automatically panel: DSNTIPS

The entries on this panel name the databases, table spaces, and index spaces to restart automatically when you start DB2.

Updating the parameters: You can update all parameters on this panel by using their subsystem parameter name.

```
DSNTIPS         INSTALL DB2 - DATABASES AND SPACES TO START AUTOMATICALLY
===> _

Enter data below:

 1  ===> RESTART       RESTART or DEFER the objects named below.
    The objects to restart or defer can be ALL in item 2, a database
    name, or database name.space name.

 2  ==> ALL              14  ==>                26  ==>
 3  ==>                  15  ==>                27  ==>
 4  ==>                  16  ==>                28  ==>
 5  ==>                  17  ==>                29  ==>
 6  ==>                  18  ==>                30  ==>
 7  ==>                  19  ==>                31  ==>
 8  ==>                  20  ==>                32  ==>
 9  ==>                  21  ==>                33  ==>
10  ==>                  22  ==>                34  ==>
11  ==>                  23  ==>                35  ==>
12  ==>                  24  ==>                36  ==>
13  ==>                  25  ==>                37  ==>

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

Figure 31. Databases and spaces to start automatically panel: DSNTIPS

1. **RESTART OR DEFER**

Acceptable values:	RESTART, DEFER
Default:	RESTART
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RESTART

Specify whether DB2 is to restart or defer processing for the objects that are listed in fields 2 through 37 when DB2 is started. RESTART causes DB2 to perform restart processing for the listed objects. DEFER causes DB2 not to perform restart processing for the objects.

2 - 37. **START NAMES**

Acceptable values:	ALL, space names
Default:	ALL
Update:	option 23 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM ALL

Specify the names of the databases, table spaces, and index 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 databases and spaces. This is the default.

DEFER ALL defers recovery of all objects, including DB2 catalog objects. For information about the restart process and the implications of deferring objects at restart time, see Part 4 (Volume 1) of *DB2 Administration Guide*.

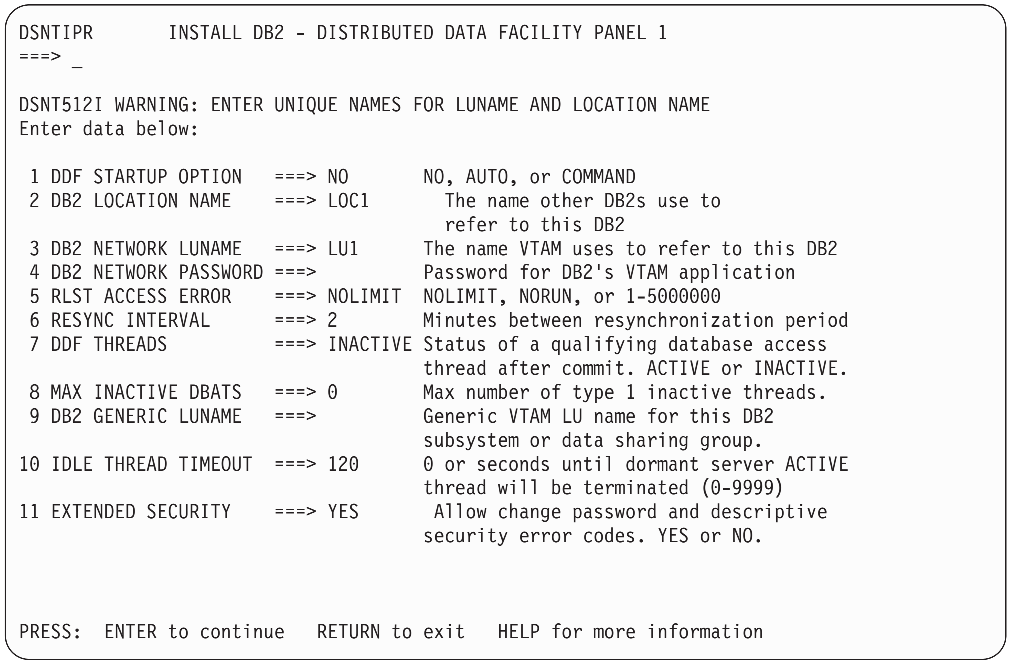
- Database name to restart or defer all spaces in that database.
- Table space or index space name in the format `database-name.space-name` to restart or defer the individual table or index space.

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.

Distributed data facility panel 1: DSNTIPR

The entries on this panel control the starting of the distributed data facility (DDF) and specify names that are used to connect another DB2 subsystem.

To use DDF, you must have VTAM installed, even if you are using only TCP/IP connections. If you do not have VTAM installed, see “Installing support for a communications network” on page 279 for instructions on installing VTAM.



```
DSNTIPR       INSTALL DB2 - DISTRIBUTED DATA FACILITY PANEL 1
===> _

DSNT512I WARNING: ENTER UNIQUE NAMES FOR LUNAME AND LOCATION NAME
Enter data below:

 1 DDF STARTUP OPTION   ===> NO       NO, AUTO, or COMMAND
 2 DB2 LOCATION NAME    ===> LOC1       The name other DB2s use to
                                        refer to this DB2
 3 DB2 NETWORK LUNAME   ===> LU1      The name VTAM uses to refer to this DB2
 4 DB2 NETWORK PASSWORD ===>          Password for DB2's VTAM application
 5 RLST ACCESS ERROR    ===> NOLIMIT  NOLIMIT, NORUN, or 1-5000000
 6 RESYNC INTERVAL      ===> 2        Minutes between resynchronization period
 7 DDF THREADS          ===> INACTIVE Status of a qualifying database access
                                      thread after commit. ACTIVE or INACTIVE.
 8 MAX INACTIVE DBATS   ===> 0        Max number of type 1 inactive threads.
 9 DB2 GENERIC LUNAME   ===>          Generic VTAM LU name for this DB2
                                      subsystem or data sharing group.
10 IDLE THREAD TIMEOUT  ===> 120      0 or seconds until dormant server ACTIVE
                                      thread will be terminated (0-9999)
11 EXTENDED SECURITY    ===> YES       Allow change password and descriptive
                                      security error codes. YES or NO.



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

Figure 32. Distributed data facility panel: DSNTIPR

1. **DDF STARTUP OPTION**

Acceptable values:	NO, AUTO, COMMAND
Default:	NO
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC DDF

Specify whether to load DDF and, if DDF is loaded, how to start it.

NO signifies that you do not want the DDF to be loaded at DB2 startup and that it cannot be started with a command. If you specify NO, the remaining fields on this panel are ignored and the stored procedures sample application and DDF sample jobs (DSNTEJ6S, DSNTEJ6P, DSNTEJ6, DSNTEJ6D, DSNTEJ6T, DSNTEJ63, DSNTEJ64, and DSNTEJ65) are not edited. You must enter values in the remaining fields, so you can accept the defaults.

AUTO specifies that this facility is automatically initialized and started when the DB2 subsystem is started.

COMMAND specifies that the facility is to be initialized at DB2 startup and is prepared to receive the DSN1 START DDF command.

If AUTO or COMMAND is specified, the DDF address space is started as part of DDF initialization. The remaining fields on this panel are mandatory. The repository for the field names (LOCATION, LUNAME, and PASSWORD) is the bootstrap data set (BSDS). The BSDS is updated by the change log inventory utility in step DSNTLOG of install job DSNTIJUZ.

2. **DB2 LOCATION NAME**

Acceptable values:	1 to 16 alphanumeric characters
Default:	LOC1
Update:	see 249 ("The update process")
DSNZP*xxx*:	none

Specify the unique name that requesters use to connect to this DB2 subsystem. The name must begin with a letter and must not contain special characters. Acceptable characters are A-Z, 0-9, and underscore.

You must specify a value, even if you do not use DDF.

3. **DB2 NETWORK LUNAME**

Acceptable values:	1 to 8 alphanumeric characters
Default:	LU1
Update:	see 249 ("The update process")
DSNZP*xxx*:	none

Specify the logical unit name (LU name) for this DB2 subsystem. This name uniquely identifies this DB2 subsystem to VTAM. It is also used to uniquely identify logical units of work within DB2 trace records. The name must begin with a letter and must not contain special characters.

You must specify a value.

4. **DB2 NETWORK PASSWORD**

Acceptable values:	1 to 8 alphanumeric characters
Default:	none
Update:	see Update on 249
DSNZP*xxx*:	none

This optional field specifies the password that VTAM uses to recognize this DB2 subsystem. This password must also be supplied to VTAM on the VTAM APPL definition statement. The password must begin with a letter and must not contain special characters.

5. **RLST ACCESS ERROR**

Acceptable values:	NOLIMIT, NORUN, 1 to 5000000
Default:	NOLIMIT
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC RLFERRD

Specify what action DB2 is to take if the governor encounters a condition that prevents it from accessing the resource limit specification table, or if DB2 cannot find a row in the table that applies to the authorization ID, the plan or package name, and the logical unit of work name of the query user.

NOLIMIT allows all dynamic SQL statements to run without limit.

NORUN terminates all dynamic SQL statements immediately with an SQL error code.

A number from 1 to 5000000 is the default limit; if the limit is exceeded, the SQL statement is terminated. For guidelines in choosing the default limit, see Part 5 (Volume 2) of *DB2 Administration Guide*.

For more information about using the governor for remote queries, see Part 5 (Volume 2) of *DB2 Administration Guide*.

6. **RESYNC INTERVAL**

Acceptable values:	1 to 99
Default:	2
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC RESYNC

Specify the time interval, in minutes, between resynchronization periods. A *resynchronization period* is the time during which indoubt logical units of work that involve this DB2 subsystem and partner logical units are processed.

7. **DDF THREADS**

Acceptable values:	ACTIVE, INACTIVE
Default:	INACTIVE
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC CMTSTAT

Specify whether to make a thread active or inactive after it successfully commits or rolls back and holds no cursors. A thread can become inactive only if it holds no cursors, has no temporary tables defined, and executes no statements from the dynamic statement cache.

If you specify ACTIVE, the thread remains active. This provides the best performance but consumes system resources. If your installation must support a large number of connections, specify INACTIVE.

If you specify INACTIVE, DB2 supports two different types of inactive concepts:

- An inactive DBAT (previously called a type 1 inactive thread), which has the same characteristics as inactive threads that were available in releases prior to DB2 UDB for z/OS Version 8. In this case, the thread remains associated with the connections, but the thread's storage utilization is reduced as much as possible. However, this still potentially requires a large number of threads to support a large number of connections.
- An inactive connection (previously called a type 2 inactive thread), which uses less storage than an inactive DBAT. In this case, the connections are disassociated from the thread. The thread is allowed to be pooled and reused for

other connections, new or inactive. This provides better resource utilization because there are typically a small number of threads that can be used to service a large number of connections.

Because they use less storage, inactive connections are preferable. However, not all threads can become inactive connections. Table 45 summarizes the conditions under which you can have an inactive DBAT or an inactive connection. If a thread is to become inactive, DB2 tries to make it an inactive connection. If DB2 cannot make it an inactive connection, it tries to make it a inactive DBAT. If neither attempt is successful, the thread remains active.

Table 45. Requirements for inactive DBATs and inactive connections

If the event is...	**Inactive connection can be created**	**Inactive DBAT can be created**
A hop to another location	Yes	Yes
A connection using DB2 private protocols	No	Yes
A package bound with RELEASE(COMMIT)	Yes	Yes
A package bound with RELEASE(DEALLOCATE)	Yes	No
A held cursor, a held LOB locator, or a package bound with KEEPDYNAMIC(YES)	No	No

IBM Systems Virtualization Engine Enterprise Workload Manager (EWLM) is a robust performance management tool that allows you to monitor and manage work that runs within your environment. If you use EWLM with DB2, specify INACTIVE for DDF THREADS. Otherwise, DDF ignores all EWLM correlators that are passed to it. For more information about using EWLM with DB2, see *DB2 Administration Guide*. For more information about EWLM, see the IBM Systems Software Information Center at the following Web site:

`http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp`

See *DB2 Administration Guide* for more information about active and inactive threads.

8. **MAX INACTIVE DBATS**

Acceptable values:	0 to the value of MAX REMOTE CONNECTED
Default:	0
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC MAXTYPE1

Specify the number of inactive DBATs that DB2 is to allow. This limit is defined because a large number of inactive DBATs might adversely affect system performance. Inactive DBATs are used for private protocol. DRDA uses inactive connections.

A value of 0 indicates that inactive DBATs are not allowed. If a thread meets the requirement of an inactive DBATs, and MAX INACTIVE DBATS is 0, the thread remains active.

A value of greater than 0 indicates that inactive DBATs are allowed, but they are limited to the specified number. When a thread meets the requirement of an inactive DBAT, and MAX INACTIVE DBATS is reached, the remote connection is terminated.

If you want to allow inactive DBATs, set this value to the maximum number of concurrent connections that you want to allow to go inactive that access another remote location with three-part names.

A value that is equal to the value in the MAX REMOTE CONNECTED field from panel DSNTIPE allows all remote threads to become type 1 inactive threads.

9. **DB2 GENERIC LUNAME**

Acceptable values:	1 to 8 alphanumeric characters
Default:	none
DSNZP*xxx*:	none

Specify a generic LUNAME to identify this DB2 subsystem or data sharing group in a network. You can use a generic LU name only if DB2 is running as part of a z/OS Sysplex. Using a generic LUNAME helps you control the distributed workload among the servers in a data sharing group. Previously, you could associate only one LUNAME with a LOCATION name. Now, you can associate multiple NETID.LUNAME values with a single LOCATION name. When an application requests access to a particular location, DB2 uses the SYSIBM.LOCATIONS and SYSIBM.LULIST tables to find the available network destinations (LUNAMEs) for that location. See Chapter 12, "Connecting distributed database systems," on page 449 for more information about setting up a generic LU name.

10. **IDLE THREAD TIMEOUT**

Acceptable values:	0 to 9999
Default:	120
Update:	option 24 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC IDTHTOIN

Specify the approximate time, in seconds, that an active server thread should be allowed to remain idle before it is canceled. The thread is canceled after the timeout value expires; its locks and cursors are released. Inactive and indoubt threads are not subject to time-out. The value that you specify for DDF THREADS determines whether a thread can become inactive, and thus not subject to time-out.

Threads are checked every two minutes to see if they have exceeded the time-out value. If the time-out value is less than two minutes, the thread might not be canceled if it has been inactive for more than the time-out value but less than two minutes.

Specifying 0 disables time-out processing. If time-out processing is disabled, idle server threads remain in the system and continue to hold their resources, if any.

11. **EXTENDED SECURITY**

Acceptable values:	YES, NO
Default:	YES

|	Update:	option 24 on panel DSNTIPB
	DSNZP*xxx*:	DSN6SYSP EXTSEC

Specify how you want to set up two related security options.

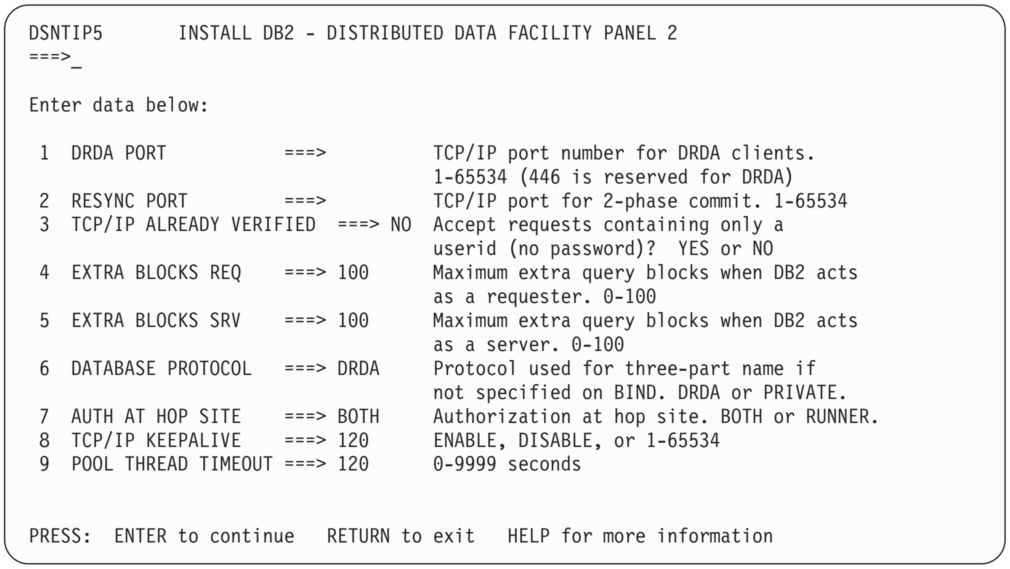
If you specify YES:

- Detailed reason codes are returned to a DRDA level 3 client when a DDF connection request fails because of security errors. When using SNA protocols, the requester must have included a product that supports the extended security sense codes. One such product is DB2 Connect.
- RACF users can change their passwords by using the DRDA change password function. This support is only for DRDA requesters that have implemented support for changing passwords.

Specifying NO returns generic error codes to the clients and prevents RACF users
from changing their passwords.

Recommendation: Specify a value of YES. This allows properly enabled DRDA clients to determine the cause of security failures without requiring DB2 operator support. A value of YES also allows RACF users on properly enabled DB2 clients to change their passwords.

Distributed data facility panel 2: DSNTIP5



```
DSNTIP5       INSTALL DB2 - DISTRIBUTED DATA FACILITY PANEL 2
===>_

Enter data below:

 1  DRDA PORT           ===>          TCP/IP port number for DRDA clients.
                                      1-65534 (446 is reserved for DRDA)
 2  RESYNC PORT         ===>          TCP/IP port for 2-phase commit. 1-65534
 3  TCP/IP ALREADY VERIFIED  ===> NO  Accept requests containing only a
                                      userid (no password)?  YES or NO
 4  EXTRA BLOCKS REQ    ===> 100      Maximum extra query blocks when DB2 acts
                                      as a requester. 0-100
 5  EXTRA BLOCKS SRV    ===> 100      Maximum extra query blocks when DB2 acts
                                      as a server. 0-100
 6  DATABASE PROTOCOL   ===> DRDA     Protocol used for three-part name if
                                      not specified on BIND. DRDA or PRIVATE.
 7  AUTH AT HOP SITE    ===> BOTH     Authorization at hop site. BOTH or RUNNER.
 8  TCP/IP KEEPALIVE    ===> 120      ENABLE, DISABLE, or 1-65534
 9  POOL THREAD TIMEOUT ===> 120      0-9999 seconds


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

Figure 33. Distributed data facility panel: DSNTIP5

1. **DRDA PORT**

Acceptable values:	1 to 65534
Default:	none
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the TCP/IP port number that is to be used for accepting TCP/IP connection requests from remote DRDA clients. If you are enabling data sharing, each member must have the same DRDA port number. You must specify a value if you plan to use TCP/IP. Leaving this field blank means that you are not using TCP/IP. A blank field is equivalent to using 0 in the change log inventory (DSNJU003) utility. See Section 3 of *DB2 Utility Guide and Reference* for more information about the change log inventory utility.

2. **RESYNC PORT**

Acceptable values:	1 to 65534
Default:	none
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	none

Specify the TCP/IP port number that is to be used to process requests for two-phase commit resynchronization. This value must be different than the value that is specified for DRDA PORT. If you are enabling data sharing, each member must have a unique resynchronization port. Leaving this field blank means that you are not using TCP/IP. A blank field is equivalent to using 0 in the change log inventory (DSNJU003) utility. See Section 3 of *DB2 Utility Guide and Reference* for more information about the change log inventory utility.

3. **TCP/IP ALREADY VERIFIED**

Acceptable values:	YES, NO
Default:	NO
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC TCPALVER

Specify whether TCP/IP connection requests that contains only a user ID (no password, RACF PassTicket, or Kerberos ticket) are to be accepted by DB2. YES means a connection request is accepted with a user ID only. This value must be the same for all members of a data sharing group. This option applies to all incoming requests that use TCP/IP regardless of the requesting location. See Part 3 (Volume 1) of *DB2 Administration Guide* for more information.

4. **EXTRA BLOCKS REQ**

Acceptable values:	0 to 100
Default:	100
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP EXTRAREQ

Specify an upper limit on the number of extra DRDA query blocks DB2 requests from a remote DRDA server. This does not limit the size of the SQL query answer set. It simply controls the total amount of data that can be transmitted on any given network exchange.

5. **EXTRA BLOCKS SRV**

Acceptable values:	0 to 100
Default:	100
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP EXTRASRV

Specify an upper limit on the number of extra DRDA query blocks DB2 returns to a DRDA client. This does not limit the size of the SQL query answer set. It simply controls the total amount of data that can be transmitted on any given network exchange.

6. **DATABASE PROTOCOL**

Acceptable values:	DRDA or PRIVATE
Default:	DRDA
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP DBPROTCL

Specify the default protocol (DRDA or PRIVATE) that is to be used when option DBPROTOCOL BIND is not explicitly specified for the bind of a plan or a package. The default value for DATABASE PROTOCOL is DRDA.

An application program might contain statements with three-part names or aliases that reference remote objects. At bind or rebind of a plan, a user can specify whether these statements flow to the remote site using DB2 private protocol access or DRDA protocol when communicating with the remote server.

Although DRDA is generally the recommended value, in some cases PRIVATE is needed. Choose a value of PRIVATE if you do not plan to move applications that use three-part names to DRDA access immediately. To use DRDA access for applications with three-part names, you must bind packages for those applications at each location that the applications access, then bind all packages into a plan. If you cannot perform this activity immediately, and you want your applications to continue to work, you should specify PRIVATE for DATABASE PROTOCOL. See *DB2 Application Programming and SQL Guide* for more information on moving applications from DB2 private protocol access to DRDA access.

The BIND commands for DB2-supplied applications are in job DSNTIJSG. For information on job DSNTIJSG, see “Migration step 23: Bind SPUFI and DCLGEN and user-maintained database activity: DSNTIJSG” on page 340.

An application that uses DB2 private protocol access cannot include SQL statements that were added to DB2 after Version 7.

7. **AUTH AT HOP SITE**

Acceptable values:	BOTH or RUNNER
Default:	BOTH
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM HOPAUTH

Indicate whose authorization is to be checked at a second server (sometimes called a “hop” site) when the request is from a requester that is not DB2 UDB for z/OS. This option applies only when private protocol access is used for the hop from the second to third site. See "Privileges exercised through a plan or package" in Part 3 of *DB2 Administration Guide* for more information about authorization for distributed processing.

BOTH, the default, means that the package owner’s authorization is checked for static SQL, and the runner’s authorization ID is checked for dynamic SQL.

RUNNER means that both static and dynamic SQL use the runner’s authorization.

8. **TCP/IP KEEPALIVE**

Acceptable values:	ENABLE, DISABLE, or 1 to 65534
Default:	120
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC TCPKPALV

In cases where the TCP/IP KeepAlive value in the TCP/IP configuration is not appropriate for the DB2 subsystem, you can use this field as an override. You can specify the following values:

- ENABLE: Do not override the TCP/IP KeepAlive configuration value.
- DISABLE: Disable KeepAlive probing for this subsystem.
- 1 to 65534: Override the TCP/IP KeepAlive configuration value with the entered number. This value is specified in seconds. Consider setting this value close to the IDLE THREAD TIMEOUT value on installation panel DSNTIPR or the IRLM RESOURCE TIMEOUT value on installation panel DSNTIPI.

Avoid using very small values. KeepAlive detection is accomplished by probing the network based on the time that is entered in the KeepAlive parameter. A small KeepAlive value can cause excessive network traffic and system resource consumption. Maintain a proper balance that allows network failures to be detected on a timely basis without a severe impact on system and network performance. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information.

9. **POOL THREAD TIMEOUT**

Acceptable values:	0 to 9999
Default:	120
Update:	option 25 on panel DSNTIPB
DSNZP*xxx*:	DSN6FAC POOLINAC

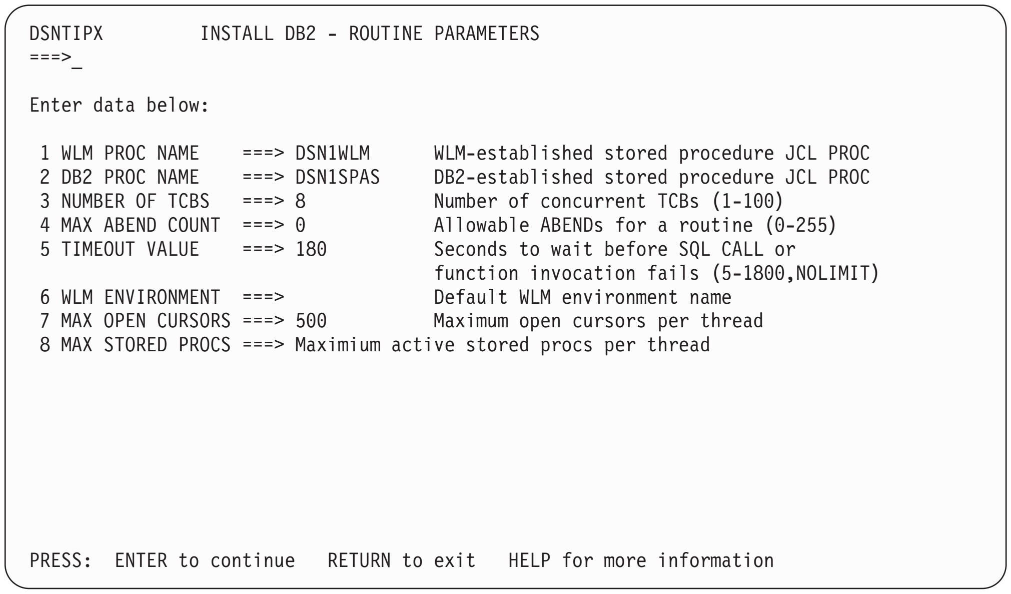
Specify the approximate time, in seconds that a database access thread (DBAT) can remain idle in the pool before it is terminated. A database access thread in the pool counts as an active thread against MAX REMOTE ACTIVE and can hold locks, but does not have any cursors.

Threads are checked every three minutes to see if they have exceeded the time-out value. If the time-out value is less than three minutes, the thread might not be cancelled if it has been inactive for more than the time-out value but less than three minutes.

Specifying 0 causes a DBAT to terminate rather than go into the pool if the pool has a sufficient number of threads to process the number of inactive DBATs (type 2 inactive threads) that currently exist.

Routine parameters panel: DSNTIPX

The entries on this panel are used to start the stored procedures address space to run stored procedures or user-defined functions. See “Enabling stored procedures after installation” on page 285 if you want to enable stored procedures outside of the standard installation steps.



```
DSNTIPX         INSTALL DB2 - ROUTINE PARAMETERS
===>_

Enter data below:

 1 WLM PROC NAME    ===> DSN1WLM      WLM-established stored procedure JCL PROC
 2 DB2 PROC NAME    ===> DSN1SPAS     DB2-established stored procedure JCL PROC
 3 NUMBER OF TCBS   ===> 8            Number of concurrent TCBs (1-100)
 4 MAX ABEND COUNT  ===> 0            Allowable ABENDs for a routine (0-255)
 5 TIMEOUT VALUE    ===> 180          Seconds to wait before SQL CALL or
                                      function invocation fails (5-1800,NOLIMIT)
 6 WLM ENVIRONMENT  ===>              Default WLM environment name
 7 MAX OPEN CURSORS ===> 500          Maximum open cursors per thread
 8 MAX STORED PROCS ===> Maximium active stored procs per thread









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

Figure 34. Routine parameters panel: DSNTIPX

1. **WLM PROC NAME**

Acceptable values:	1 to 8 alphanumeric characters
Default:	ssnWLM
DSNZP*xxx*:	none

Specify a name for the stored procedures JCL procedure that is generated during installation. This procedure is used for a WLM-established stored procedures address space.

If this field has a blank, the JCL procedure is still generated. In this case, the JCL procedure will be named by appending the string `WLM` to the DB2 subsystem name (specified on panel DSNTIPM in the field SUBSYSTEM NAME).

2. **DB2 PROC NAME**

Acceptable values:	1 to 8 alphanumeric characters
Default:	blank
DSNZP*xxx*:	DSN6SYSP STORPROC

Specify a name for the JCL procedure that is used to start the DB2-established address space. If you replace the default value with blanks, you cannot start the DB2-established stored procedures address space until you update the subsystem parameter.

In Version 8, support for DB2-established address spaces is deprecated. If you are installing DB2, this field is blank and cannot be updated. If you are migrating from

DB2 Version 7, you can change this field if required. Although existing stored procedures can still run in a DB2-established stored procedure address space, you should move your stored procedures to WLM environments as soon as possible. If you CREATE or ALTER an existing stored procedure, it cannot run in a DB2-established stored procedure address space. For more information about moving stored procedures, see Part 5 (Volume 2) of *DB2 Administration Guide*

3. **NUMBER OF TCBS**

Acceptable values:	1 to 100
Default:	8
DSNZP*xxx*:	none

Specify how many SQL CALL statements or an invocation of a user-defined function can be processed concurrently in one address space. The larger the value, the more stored procedures and user-defined functions you can run concurrently in one address space. This value is dependent on the z/OS Unix System Services MAXPROCUSER value. If this value is set above the z/OS Unix System Services MAXPROCUSER value, you may exceed the maximum number of processes for the user. For information about how this value affects storage below the 16-MB line, see Part 5 (Volume 2) of *DB2 Administration Guide*.

4. **MAX ABEND COUNT**

Acceptable values:	0 to 255
Default:	0
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STORMXAB

Specify the number of times a stored procedure or an invocation of a user-defined function is allowed to terminate abnormally, after which SQL CALL statements for the stored procedure or user-defined function are rejected. The default of 0 means that the first abend of a stored procedure or user defined function causes SQL CALLs to that procedure or function to be rejected. For production systems, you should accept the default.

5. **TIMEOUT VALUE**

Acceptable values:	5 to 1800
Default:	180
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP STORTIME

Specify the number of seconds before DB2 is to stop waiting for an SQL CALL or invocation of a user-defined function to be assigned to one of the task control blocks (TCBs) in a DB2 stored procedures address space. If the time interval expires, the SQL statement fails. The default is a reasonable waiting time for most sites. You might want to choose a higher value if your system has long queues. You might want to choose a lower value if you want to minimize the waiting time for end-user requests. The NOLIMIT value means that DB2 waits indefinitely for the SQL request to complete, while the thread is active.

Recommendation: Do not select the NOLIMIT value. If the stored procedure address space is down for some reason or the user-defined function does not complete, your SQL request hangs until the request is satisfied or the thread is canceled.

6. **WLM ENVIRONMENT**

Acceptable values:	Any valid name from 1 to 18 alphanumeric characters, including underscores
Default:	blank
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	DSN6SYSP WLMENV

Specify the name of the WLM_ENVIRONMENT to use for a user-defined function or stored procedure when a value is not given for the WLM_ENVIRONMENT option on the CREATE FUNCTION or CREATE PROCEDURE statements. The name can include the underscore character.

Changing this value does not change existing routines because the value is stored in the catalog when the function or procedure is created.

7. **MAX OPEN CURSORS**

Acceptable values:	0 to 99999
Default:	500
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAX_NUM_CUR

Specify the maximum number of cursors, including allocated cursors, that are open at a given DB2 site per thread. RDS will keep a total of currently open cursors. If an application attempts to open a thread after the maximum is reached, the statement will fail.

In a data sharing group, this parameter has member scope.

8. **MAX STORED PROCS**

Acceptable values:	0 to 99999
Default:	2000
Update:	option 26 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAX_ST_PROC

Specify the maximum number of stored procedures per thread. If an application attempts to call a stored procedure after the maximum is reached, the statement will fail.

In a data sharing group, this parameter has member scope.

Data definition control support panel: DSNTIPZ

The entries on this panel allow you to install and tailor data definition control support. Two SQL tables (application registration and object registration) are identified and created even if data definition control support is not installed. This simplifies future activation of the facility. Specified application identifiers (DB2 plans or collections of packages) can be registered in the application registration table, and, optionally, their associated DB2 object names can be registered in the object registration table. DB2 consults these two tables prior to accepting a given DDL statement to make sure that a particular application identifier and object name are registered. For guidance on data definition control support, see Part 3 (Volume 1) of *DB2 Administration Guide*.

```
DSNTIPZ      INSTALL DB2 - DATA DEFINITION CONTROL SUPPORT
===>

Enter data below:

 1 INSTALL DD CONTROL SUPT. ===> NO       YES - activate the support
                                          NO  - omit DD control support
 2 CONTROL ALL APPLICATIONS ===> NO       YES or NO
 3 REQUIRE FULL NAMES       ===> YES      YES or NO
 4 UNREGISTERED DDL DEFAULT ===> ACCEPT   Action for unregistered DDL:
                                          ACCEPT - allow it
                                          REJECT - prohibit it
                                          APPL   - consult ART
 5 ART/ORT ESCAPE CHARACTER ===>          Used in ART/ORT Searches
 6 REGISTRATION OWNER       ===> DSNRGCOL         Qualifier for ART and ORT
 7 REGISTRATION DATABASE    ===> DSNRGFDB         Database name
 8 APPL REGISTRATION TABLE  ===> DSN_REGISTER_APPL Table name
 9 OBJT REGISTRATION TABLE  ===> DSN_REGISTER_OBJT Table name

Note: ART = Application Registration Table
      ORT = Object      Registration Table

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

Figure 35. Data definition control support panel: DSNTIPZ

1. **INSTALL DD CONTROL SUPT.**

Acceptable values:	YES, NO
Default:	NO
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFINSTL

Specify whether to install data definition control support. If NO is specified, DDL statements are not validated by this support. The application registration table and object registration table are still created according to values that are entered in fields 5 through 8.

2. **CONTROL ALL APPLICATIONS**

Acceptable values:	YES, NO
Default:	NO
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFDEDPL

Specify whether the DB2 subsystem is completely controlled by a set of closed applications whose application identifiers are identified in the application registration table. Closed applications require their DB2 objects to be managed solely through the plans or packages of the closed application that is registered in the application registration table.

3. **REQUIRE FULL NAMES**

Acceptable values:	YES, NO
Default:	YES
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFFULLQ

Specify whether registered objects require fully qualified names.

4. **UNREGISTERED DDL DEFAULT**

Acceptable values:	ACCEPT, REJECT, APPL
Default:	ACCEPT
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFDEFLT

Specify what action is taken for DDL that names an unregistered object. If the ACCEPT option is specified, the DDL is accepted. If the REJECT option is specified, the DDL is rejected. If APPL is specified, the DDL is rejected if the current application is not registered.

5. **ART/ORT ESCAPE CHARACTER**

Acceptable values:	any non-alphanumeric character
Default:	none
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFESCP

Specify the escape character that is to be used in the application registration table (ART) or object registration table (ORT). Sets of names in the ART and ORT can be represented by patterns that use the underscore(_) and percent sign (%) characters in the same way as in an SQL LIKE predicate.

If you enter a character in this field, it can be used in those patterns in the same way as an escape character is used in an SQL LIKE predicate. See Part 3 (Volume 1) of *DB2 Administration Guide* for examples of using the percent and underscore characters and the escape character.

6. **REGISTRATION OWNER**

Acceptable values:	1 to 8 characters
Default:	DSNRGCOL
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFCOLID

Specify the owner of both the application registration table and the object registration table.

7. **REGISTRATION DATABASE**

Acceptable values:	1 to 8 characters
Default:	DSNRGFDB
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFDBNAM

Specify the name of the database that contains the registration tables.

8. **APPL REGISTRATION TABLE**

Acceptable values:	1 to 17 characters
Default:	DSN_REGISTER_APPL
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFNMPRT

Specify the name of the application registration table.

9. **OBJT REGISTRATION TABLE**

Acceptable values:	1-17 characters
Default:	DSN_REGISTER_OBJT
Update:	option 27 on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM RGFNMORT

Specify the name of the object registration table.

Job editing panel: DSNTIPY

The entries on this panel specify values and information about job statements for the installation and sample application jobs.

Establishing system affinity for installation jobs: You must ensure that the installation jobs run on the z/OS system where the appropriate DB2 subsystem is running. To do this, you can choose between these methods:

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

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

  In this statement, *cccc* is the JES2 name. You can specify an asterisk (SYSAFF=*) to indicate that the job should run on the system from which it was submitted.
- For JES3 systems, use the following JCL statement:

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

  In this statement, *main-name* is the JES3 name.

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

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

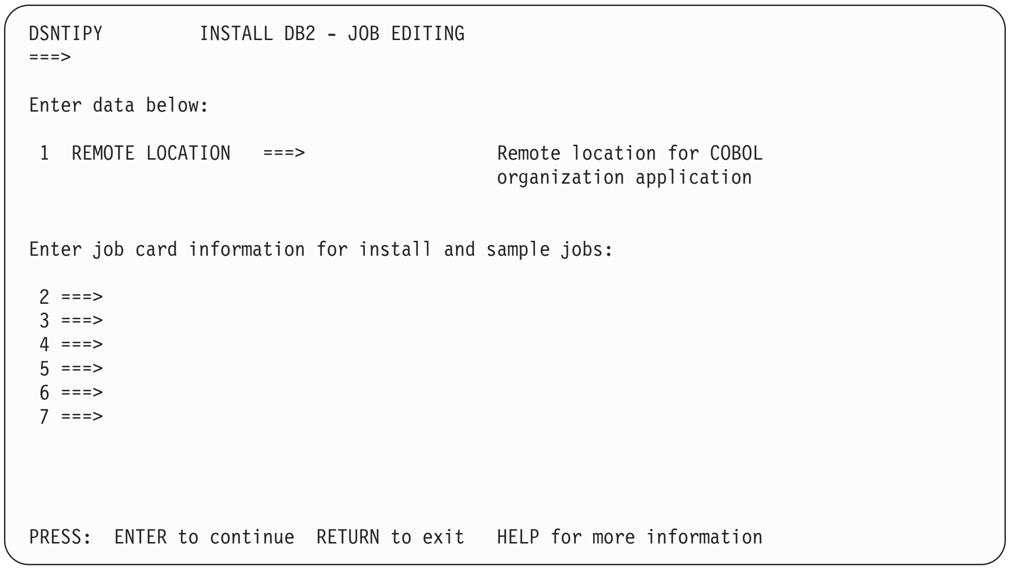
Ensuring that installation jobs access the right JCL procedures: If your z/OS system has more than one procedure library, you need to ensure that your installation jobs access the correct set of procedures. One way to do this is to use a JCLLIB statement to specify the order for procedure libraries.

The JCLLIB statement has the following form:

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

The JCLLIB statement must follow the JOB statement and precede the first EXEC statement in the job. If you enter this statement on panel DSNTIPY, DB2 inserts it into your JCL.

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



```
DSNTIPY         INSTALL DB2 - JOB EDITING
===>

Enter data below:

 1  REMOTE LOCATION   ===>                  Remote location for COBOL
                                            organization application


Enter job card information for install and sample jobs:

 2 ===>
 3 ===>
 4 ===>
 5 ===>
 6 ===>
 7 ===>




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

Figure 36. Job editing panel: DSNTIPY

1. **REMOTE LOCATION**

Acceptable values:	1 to 16 characters
Default:	none
Update:	See "The update process" on page 247
DSNZP*xxx*:	none

Specify the location name of another DB2 subsystem to be used by the COBOL preparation sample job (DSNTEJ3C), the DDF remote location update sample job (DSNTEJ6), and the stored procedures sample jobs (DSNTEJ6S, DSNTEJ6P, DSNTEJ6T, DSNTEJ6D, DSNTEJ63, DSNTEJ64, and DSNTEJ65). The name must begin with a letter and must not contain special characters. A remote location name is accepted only if you have also entered a DB2 location name for DB2 LOCATION NAME (field 2 on installation panel DSNTIPR).

2-7. **job card information**

Acceptable values:	see below
Default:	none
Update:	See "The update process" on page 247
DSNZP*xxx*:	none

Specify the job statements that are to be used in all the installation and sample application jobs.

Specify the job name in one of two ways:
- If the job name is *member*, the job name for each job is the same as its member name.
- If the job name is any value other than *member*, the name is truncated to seven characters, and one character is added to the end of the name identifying the run order for that job.

An example of job card information follows:

```
3====> //MEMBER JOB,
4====> // MSGLEVEL=(1,1),MSGCLASS=H,REGION=4096,CLASS=A
5====> // USER=SYSADM,PASSWORD=SYSADM,NOTIFY=SYSADM
6====>
```

CLIST calculations panel 1: DSNTIPC

This panel displays the messages produced by the installation CLIST indicating calculated storage sizes. Space estimates from these messages do not account for cylinder rounding. Base requirements can be 10 to 20% higher than the message indicates depending on the disk type. If you need more information about these messages, see Part 2 of *DB2 Messages*.

The messages 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).

During the tailoring session, a warning message is issued to the tailoring terminal. This message is always issued if you accept the default.

```
DSNT438I WARNING, IRLM LOCK MAXIMUM SPACE = irlmreg K, AVAILABLE = irlmav K
```

This message indicates that the IRLM could request a total amount of space that is larger than the available space, causing an abend. The message is based

- The maximum number of data or row locks per user specified on installation panel DSNTIPJ (LOCKS PER USER)
- The number of users specified on installation panel DSNTIPE for MAX USERS and MAX REMOTE ACTIVE during the tailoring session

The formula is:

```
(MAX USERS + MAX REMOTE ACTIVE) * LOCKS PER USER * 250 bytes per lock
```

| The CLIST assumes that the private region that is available for IRLM locks is
| estimated as 60000 KB, if extended private address space is used.

When using the default in the tailoring session you get:

```
70 * 10000 * 250 = 175000 KB
```

This amount is a high-end estimate. It is the amount of storage that is needed if the maximum number of users are connected and each user uses the maximum number of locks. Most users hold only a few locks.

```
DSNTIPC         INSTALL DB2 - CLIST CALCULATIONS - PANEL 1
===>

 You can update the DSMAX, EDMPOOL, EDMPOOL STATEMENT CACHE,
 EDM DBD CACHE, SORT POOL, and RID POOL sizes if necessary.


                                       Calculated   Override

 1  DSMAX - MAXIMUM OPEN DATA SETS    =     9960               (1-100000)
 2  DSNT485I  EDMPOOL STORAGE SIZE    =    32500  K           K
 3  DSNT485I  EDM STATEMENT CACHE     =   101562  K           K
 4  DSNT485I  EDM DBD CACHE           =   101562  K           K
 5  DSNT485I  BUFFER POOL SIZE        =      101  M
 6  DSNT485I  SORT POOL SIZE          =     2000  K           K
 7  DSNT485I  RID POOL SIZE           =     8000  K           K
 8  DSNT485I  DATA SET STORAGE SIZE   =    17982  K
 9  DSNT485I  CODE STORAGE SIZE       =    25000  K
10  DSNT485I  WORKING STORAGE SIZE    =    55800  K
11  DSNT486I  TOTAL MAIN STORAGE      =      239  M
12  DSNT487I  TOTAL STORAGE BELOW 16M =     1159  K WITH SWA ABOVE 16M LINE
13  DSNT438I  IRLM LOCK MAXIMUM SPACE =        2  G, AVAILABLE =   2 G

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

Figure 37. CLIST calculations panel 1: DSNTIPC

1. **DSMAX**

Acceptable values:	1 to 100 000
Default:	based on calculations
Update:	enter a value in the override column
DSNZP*xxx*:	DSN6SPRM DSMAX

Specify the maximum number of data sets that can be open at one time. Although the maximum number of data sets is 100 000, the practical limit may be less than 30 000, depending on available storage below the line. For z/OS Version 1 Release 6 or earlier, specify a value that is less than or equal to 65 041. The value that you enter can substantially influence the performance of DB2; see Part 5 (Volume 2) of *DB2 Administration Guide* for more information.

When a secondary index is nonpartitioned, the number of data sets that are required for the index is dependent on the total required space to contain the tree structure and on the size limit of each data set. When a secondary index is data-partitioned, the number of data sets that are required for the index is equal to the number of data partitions in the table space that contains the table. Unless a small piece size is used for nonpartitioned secondary indexes, partitioning generally results in an increase in the number of data sets for the index.

If the partitioning of secondary indexes causes the number of data sets to increase or decrease appreciably, you can modify the value of DSMAX. The default value for DSMAX is calculated by DB2 and does not count partitioned objects. Choose the value for DSMAX according to the impact partitioning secondary indexes have on the number of data sets for those objects. DB2 defers closing and deallocating table spaces or indexes until the number of open data sets reaches the operating system limit or 99% of the value that is specified in DSMAX.

2. **EDMPOOL STORAGE SIZE**

Acceptable values:	0 to 2097152
Default:	based on calculations

Update:	press ENTER twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMPOOL

Specify the size of the environmental descriptor manager (EDM) pool that is calculated by the CLIST in kilobytes. This value is used at DB2 start time as the minimum value. It can be increased and subsequently decreased with the SET SYSPARM command. The EDM pool is located below the 2-GB bar. You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC ; the CLIST calculates this value based on input from previous panels. If there is a value in the Override column, you must erase the override value in order to accept the calculated value.
- Typing your own value in the Override column of panel DSNTIPC.

For information on how DB2 calculates the EDM pool size, see "EDM pool size calculation" on page 36.

3. **EDM STATEMENT CACHE**

Acceptable values:	5000 to 1048576
Default:	see below
Update:	press ENTER on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMSTMTC

The default value for EDM STATEMENT CACHE is taken from the EDMPOOL STORAGE SIZE field on panel DSNTIPC, or 5000 K, whichever is larger.

Specify the size (in KB) of the statement cache that can be used by the EDM. This value is used at DB2 startup time as the minimum value. This value cannot be decreased below the value that is specified at DB2 startup. The CLIST calculates a statement cache size. This storage pool is located above the 2-GB bar. You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC; the CLIST calculates this value based on input from previous panels. If a value is in the Override column, you must erase the override value in order to accept the calculated value.
- Entering your own value in the Override column of panel DSNTIPC.

For information on how DB2 calculates the EDM data space pool size, see "EDM pool size calculation" on page 36.

4. **EDM DBD CACHE**

Acceptable values:	5000 to 2097152
Default:	below
Update:	press ENTER on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM EDMDBDC

The default value for EDM DBD CACHE is the larger of the EDM calculated value or 5000 K.

Specify the minimum size (in KB) of the DBD cache that can be used by the environmental descriptor manager (EDM). This value is used at DB2 startup time as the minimum value. This value cannot be decreased below the value that is specified at DB2 startup. This storage pool is located above the 2-GB bar. The CLIST calculates the DBD cache size.

You have a choice of:

- Accepting the value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the calculated value.
- Entering your own value in the Override column of panel DSNTIPC.

5. **BUFFER POOL SIZE**

Acceptable values:	none
Default:	based on calculations
Update:	run CLIST again
DSNZP*xxx*:	none

Specify the buffer pool size that is calculated by the CLIST. This field is protected and cannot be changed during update processing. If you want to change the size of a buffer pool, you must use the command ALTER BUFFERPOOL, as described in *DB2 Command Reference*.

6. **SORT POOL SIZE**

Acceptable values:	240K to 128000K
Default:	2000K
Update:	press Enter twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM SRTPOOL

Specify the amount of storage that is needed for the sort pool. You have a choice of:

- Accepting the default value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the default value.
- Typing your own value in the Override column of panel DSNTIPC.

If you decide to change this field, estimate the sort pool value by using the following formula:

```
32000 * (12 + sort key length + sort data length
+ 4 (if ESA hardware sort assist))
```

See Part 5 (Volume 2) of *DB2 Administration Guide* for detailed instructions on choosing sizes for optimal performance.

7. **RID POOL SIZE**

Acceptable values:	0, 128K to 10000000K
Default:	8000K
Update:	press Enter twice on panel DSNTIPB
DSNZP*xxx*:	DSN6SPRM MAXRBLK

Specify the amount of storage needed for the RID pool as calculated by the CLIST. You have a choice of:

- Accepting the default value in the Calculated column of panel DSNTIPC. If a value is in the Override column, you must erase the override value in order to accept the default value.
- Typing your own value in the Override column of panel DSNTIPC.

If you decide to change this field, estimate the storage that is required for the RID pool with the following formula:

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

Choosing 0 disables the use of the RID pool. In this case, DB2 does not use access paths or join methods that depend on RID pool storage.

Twenty-five percent of this storage pool is located below the 2-GB bar and 75% is located above the 2-GB bar.

See Part 5 (Volume 2) of *DB2 Administration Guide* for how to choose a RID pool size for optimal performance.

8-11. **CLIST messages**

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

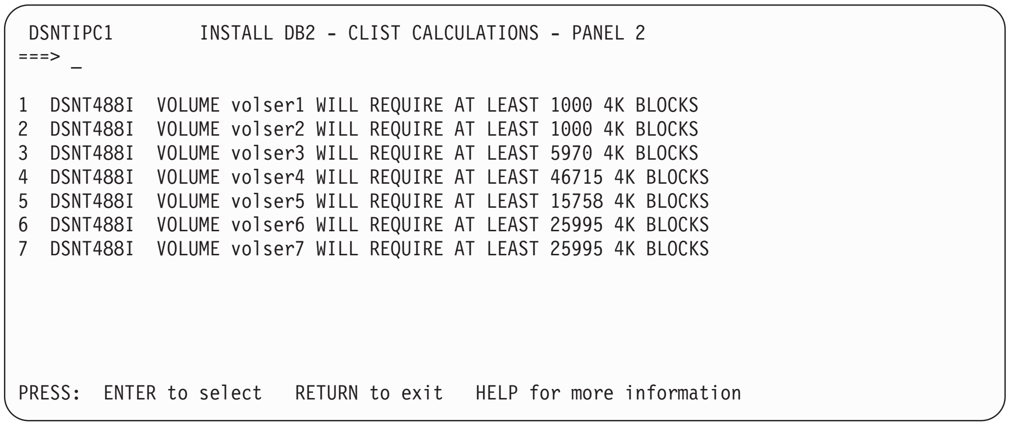
Specify sizes that are calculated by the CLIST. Fields 8 through 13 are protected and cannot be changed by the user.

12-13. **Storage messages**

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

Indicate the results of the calculations described in Figure 37 on page 239. These fields are protected and cannot be changed by the user.

CLIST calculations panel 2: DSNTIPC1



```
 DSNTIPC1        INSTALL DB2 - CLIST CALCULATIONS - PANEL 2
===> _

1  DSNT488I  VOLUME volser1 WILL REQUIRE AT LEAST 1000 4K BLOCKS
2  DSNT488I  VOLUME volser2 WILL REQUIRE AT LEAST 1000 4K BLOCKS
3  DSNT488I  VOLUME volser3 WILL REQUIRE AT LEAST 5970 4K BLOCKS
4  DSNT488I  VOLUME volser4 WILL REQUIRE AT LEAST 46715 4K BLOCKS
5  DSNT488I  VOLUME volser5 WILL REQUIRE AT LEAST 15758 4K BLOCKS
6  DSNT488I  VOLUME volser6 WILL REQUIRE AT LEAST 25995 4K BLOCKS
7  DSNT488I  VOLUME volser7 WILL REQUIRE AT LEAST 25995 4K BLOCKS





PRESS:  ENTER to select   RETURN to exit   HELP for more information
```

Figure 38. CLIST calculations panel 2: DSNTIPC1

1-6. **CLIST messages**

Acceptable values:	none
Default:	none
Update:	run CLIST again
DSNZP*xxx*:	none

These CLIST messages indicate a variety of calculations. These messages might be used depending on how many unique volume names you supplied on installation panel DSNTIPA2.

Completing the CLIST processing

After receiving the CLIST messages (on installation panel DSNTIPC1) that indicate the calculated sizes, press Enter to begin CLIST processing. You then receive a series of messages that give details about the CLIST processing. If you need more information about these messages, see Part 3 of *DB2 Messages*.

Responding to messages

You first receive the following message:

```
DSNT478I BEGINNING EDITED DATA SET OUTPUT
```

The CLIST is checking the parameter values that you entered. If it detects a problem, you receive an error or warning message indicating the name of the parameter and the type of problem. If you receive an error message, the CLIST cannot edit the installation or migration jobs properly. If you receive a warning message, check the conditions. A warning message can sometimes be issued even when the conditions are normal or acceptable. If you specify several large numbers in the panels, the CLIST might send a message indicating an overflow in CLIST arithmetic.

At this point the CLIST displays the Main Panel again. You can proceed through the panels, rechecking or changing parameter values.

If the CLIST does not find any errors, you receive messages that indicate the amount of required disk storage and virtual storage. For information about installation panel DSNTIPC1, which displays these messages, see "CLIST calculations panel 1: DSNTIPC" on page 238. (You might also receive some other information messages.)

Tailoring the installation jobs

The CLIST tailors each job according to the panel values that you specified. For each edited job, you receive the following message:

```
DSNT489I CLIST EDITING dsname(member), explanation
```

Important: If an error occurs while the installation CLIST edits your jobs, you will receive this message:

```
IKJ52555I NOTHING SAVED
ENTER SAVE OR END-
```

Enter END to prevent modification of the original copies of your installation jobs.

After the CLIST finishes tailoring the jobs, it displays the Main Panel again. If you need to continue your tailoring at another time, conclude this session. Then, when you start a new session, use the value that you specified for OUTPUT MEMBER NAME during this session as the value for INPUT MEMBER NAME during the new session. Enter these values on the Main Panel.

If you receive a message from the editor, such as `TEXT NOT FOUND`, enter END NOSAVE to exit. That message can indicate an error. You can rerun the CLIST with the trace control parameter set to CONTROL(SYMLIST) to learn what caused the problem. In some cases, specifying CONTROL(LIST) as the trace control parameter may provide enough information for you to find the source of the problem.

The installation CLIST uses the values that you specify on the installation panels to tailor and load the installation or migration jobs. Each job is composed of one or more JCL procedures or job steps. The CLIST loads each job as a separate member of the newly created *prefix*.NEW.SDSNSAMP. Before you run any of these jobs, however, you might want to perform some editing that the CLIST does not do. If disk allocation is completely controlled by SMS for your installation, verify that the input to IDCAMS from the install jobs does not conflict with the requirements for SMS.

This topic identifies several items you might want to add or change in the jobs. These changes are general; that is, they apply to all the jobs processed by the CLIST. Later chapters explain changes that you can make for specific jobs.

Which jobs you edit depends on the task that you are performing: installation, migration, or update. In data sharing environments, you edit different jobs depending on the data sharing function: group, member, or enable. The installation CLIST tailors a different set of jobs for each task.

If you are installing, the CLIST tailors these jobs:

DSNTIJMV	DSNTIJCA	DSNTIJIN	DSNTIJID	DSNTIJUZ
DSNTIJEX	DSNTIJVC	DSNTIJTC	DSNTIJTM	DSNTIJSG
DSNTIJIC	DSNTIJDE	DSNTEJ0	DSNTEJ1	DSNTEJ1L
DSNTEJ1P	DSNTEJ1S	DSNTEJ1T	DSNTEJ2A	DSNTEJ2C
DSNTEJ2D	DSNTEJ2E	DSNTEJ2F	DSNTEJ2P	DSNTEJ2U
DSNTEJ3C	DSNTEJ3P	DSNTEJ4C	DSNTEJ4P	DSNTEJ7
DSNTEJ71	DSNTEJ73	DSNTEJ75	DSNTESA	DSNTESC

DSNTESD	DSNTESE	DSNTEJ1U	DSNTEJ61	DSNTEJ62
DSNTEJ6V	DSNTEJ3M	DSNTEJ76	DSNTEJ77	DSNTEJ78

If you have activated DDF, the CLIST also tailors job DSNTEJ6.

If you have specified a default WLM environment name in field 6 of install panel DSNTIPX, the CLIST edits DSNTEJ6D, DSNTEJ6S, DSNTEJ6P, DSNTEJ6R, DSNTEJ6T, DSNTEJ6U, DSNTEJ6V, DSNTEJ6W, DSNTEJ6Z, DSNTEJ63, DSNTEJ64, and DSNTEJ65.

If CICS is selected, the CLIST edits DSNTEJ5A, DSNTEJ5C, and DSNTEJ5P.

If you are using data sharing, the CLIST edits DSNTIJGF and DSNTIJFT.

The installation CLIST tailors the DSNHC, DSNH, DSNU, and DSNEMC01 CLISTs for installation.

If you are migrating, the CLIST tailors the following jobs:

DSNTIJFV	DSNTIJIC	DSNTIJIN	DSNTIJMV	DSNTIJSG
DSNTIJTC	DSNTIJTM	DSNTIJUZ	DSNTIJVC	DSNTIJCX

The CLIST also tailors the DSNH, DSNHC, DSNU, and DSNEMC01 CLISTs for migration.

If you are updating, the CLIST tailors only one job: DSNTIJUZ.

If you are converting to new-function mode, the CLIST tailors the following jobs:
DSNTIJEN, DSNTIJMC, DSNTIJNE, DSNTIJNF, DSNTIJNG, DSNTIJNH,
DSNTIJNR.

All of these jobs are described in the following chapters. Recovery information is provided, along with a description of each job. Unless otherwise stated in the job description, a return code of 0 or 4 from any of the jobs indicates successful completion. Some of the jobs contain statements that could fail without causing the job to fail. For instance, delete commands for data sets, drop statements for SQL objects, and stop commands could fail when you first run a job because the data sets or objects do not exist. Unless otherwise stated, you can ignore these failures. The statements are needed to allow you to rerun the job (if necessary) without performing the deletes, drops, and stops manually; they are merely for cleanup or initialization processing.

When a job fails, follow the instructions that are provided in the recovery information for the job. If you need further recovery information, refer to *DB2 Messages*, and examine the descriptions of the messages that the job generated.

Before you begin editing, you might want to print or back up the jobs. You can print the JCL for these jobs by using IEBPTPCH or any other print facility that is available at your site.

Consider the following suggestions for possible changes or additions to the jobs:

- Tailor the jobs to suit the needs of your site. You should edit the jobs to conform to any unique requirements that you might have. Also, you might want to make any minor JCL changes for items that the ISPF panels did not handle.

- Examine the volume serial numbers that are used in the various jobs. The volume serial number fields of installation panel DSNTIPA2 allow you to specify up to seven volumes for the data sets that are defined during installation or migration. If you want to use more than seven volumes, specify them before continuing with installation or migration tasks in subsequent chapters.

  The DSNTINST CLIST spreads the data sets across the volumes that you specify. Adding more volumes to provide more separation of data sets can help improve system performance and recoverability. Many of the log data sets are large and easy to place on separate volumes. The CLIST produces a series of messages that estimate space distribution for the specified volumes. For more information, see "CLIST calculations panel 1: DSNTIPC" on page 238.
- Edit the DSNH CLIST if needed. The DSNH CLIST allows you to precompile, compile, prelink-edit, link-edit, bind, and run an application by issuing a single command. You might need to edit the DSNH CLIST to change values for some of the entries. Verify that DSNH keyword parameters for all DB2-supported compilers that your applications use. For a description of the parameters, see the information about the DSNH CLIST in *DB2 Command Reference*.
  – Check the default data set names for the licensed programs that you have installed. These defaults are in the parameter definitions at the beginning of each program. If the names and prefixes are not correct for your site, change them.
  – Check default library names. If the names and prefixes are not correct for your site, change them. Ensure that the data sets exist and are cataloged for BLIB, CLIB, LLIB, and PLIB. When the DSNH CLIST runs, it creates DBRMLIB and LOAD data sets if they do not already exist. The DBRMLIB data set is created only if the DBRMLIB(DEFAULT) is set. The following are the default library names:
    - BLIB(NONE)
    - DBRMLIB(DEFAULT) - The DBRM library must be allocated exclusively when the precompiler writes to it.
      Recommendation: Set up a temporary library, or one per user, rather than trying to share libraries. However, if your DB2 subsystem uses the DFSMSdfp partitioned data set extended (PDSE) for managing data sets, access is restricted at the member level rather than at the data set level; this provides another alternative for concurrent access to the DBRM library.
    - CLIB(NONE)
    - LLIB(NONE)
    - LOAD(RUNLIB.LOAD) - This library is allocated exclusively when it is being written.
      Recommendation: Set up a temporary library, or one per user, rather than trying to share libraries.
    - PLIB(NONE)
  – Check default processor options. If you prefer other default options, change them. The default processor options follow:

CICSOPT(NONE)	LOPTION(NONE)
COPTION(NONE)	PASS(DEFAULT)

  – Check print and work space defaults. If the default allocation sizes are not acceptable for your site, change them. The print and work space defaults follow:

PSECSPAC(20)	WSECSPAC(20)
PSPACE(20)	WSPACE(20)
WORKUNIT(DEFAULT)	

- Examine the data set names for other products. Many data set names for other products appear in the jobs. These names are shown in Table 35 on page 118. Change them if they are different at your site.

Editing the subsystem parameters, DSNHDECP values, and DSHNMCID values

Job DSNTIJUZ generates the subsystem parameter module each time you install, migrate, or update DB2. Seven macros expand to form this data-only subsystem parameter load module. It contains the DB2 execution-time parameters that you selected using the ISPF panels. These seven macros are DSN6ARVP, DSN6ENV, DSN6FAC, DSN6LOGP, DSN6SPRM, DSN6SYSP, and DSN6GRP. For more information, see "Installation step 4: Define DB2 initialization parameters: DSNTIJUZ" on page 259 or "Migration step 11: Define DB2 initialization parameters: DSNTIJUZ" on page 330.

Job DSNTIJUZ also generates the data-only load module DSNHDECP. It contains the application programming defaults. DB2 is shipped with a default DSNHDECP for compatibility with older applications. You cannot start DB2 or precompile applications with the default DSNHDECP. During DB2 start-up processing or for jobs that precompile a DB2 application, you must ensure that the DSNHDECP module that was created by job DSNTIJUZ resides in a library, usually *prefix*.SDSNEXIT, that is concatenated before the *prefix*.SDSNLOAD library where the DB2-supplied DSNHDECP resides.

Job DSNTIJUZ also creates the data-only load module DSNHMCID, which contains the EBCDIC CCSIDs for text conversion in offline messages. When you install DB2, job DSNTIJUZ creates DSNHMCID in *prefix*.SDSNEXIT and *prefix*SDSNLOAD.

In general, DSNHMCID should exist in both *prefix*.SDSNEXIT and *prefix*.SDSNLOAD. If it cannot reside in *prefix*.SDSNLOAD, take one of the following actions:

- Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in the system link list.
- Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statements for all DB2 applications, address space start-up procedures, TSO log-on procedures, CICS tasks, and IMS tasks that use DB2.

For a directory of subsystem parameters and DSNHDECP values, see Appendix C, "Directory of subsystem parameters," on page 533. DB2 allows online changes to many of the subsystem parameters. The SET SYSPARM command enables a function that allows reloading. For more information, see *DB2 Command Reference*.

The update process

This information describes how to modify some of the parameters that you specified during installation or migration of DB2. This process allows you to tailor DB2 more precisely to your needs.

The update process does not generate a complete set of installation or migration jobs, as the installation and migration process does. It generates only one job: DSNTIJUZ. This job assembles and link-edits the DB2 data-only subsystem parameter module, DSNZPARM (or the value that you specified for PARAMETER MODULE on installation panel DSNTIPO), and the application program's default module, DSNHDECP.

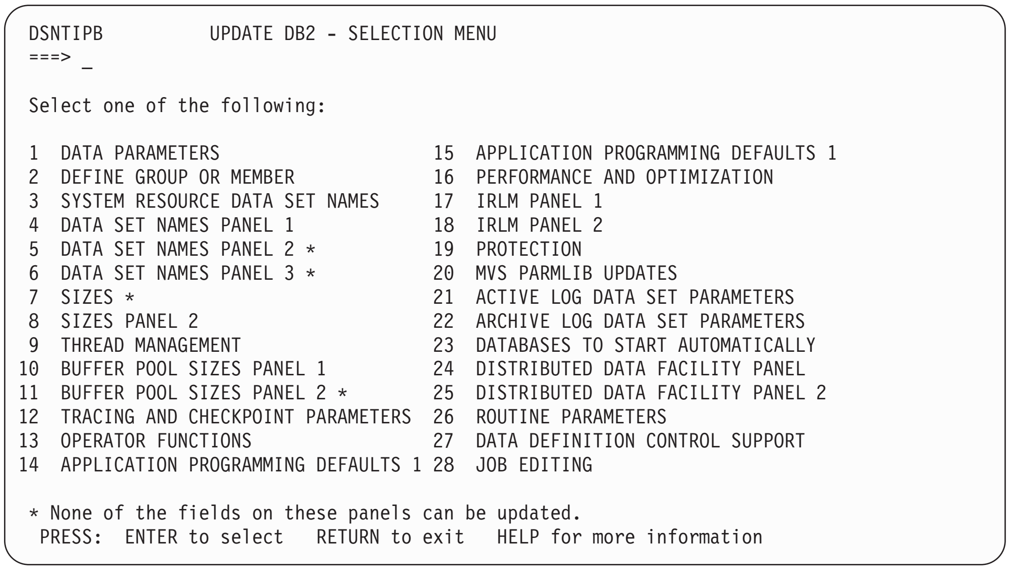
Updating parameters through the Update Selection Menu panel: DSNTIPB

To update most parameters, follow these steps:

1. Run the installation CLIST, and specify UPDATE on installation panel DSNTIPA1. See "Data set names panel 1: DSNTIPT" on page 113 for information about the output data sets.
2. Choose the output SDSNSAMP data set on installation panel DSNTIPT. See 113 for information about the output data sets.

   The CLIST then takes you to installation panel DSNTIPB.
3. From installation panel DSNTIPB, select the installation panel that you want to update. When you finish making changes to that panel, press ENTER to return to the Update Selection Menu Panel. You can select another panel to update, or press ENTER again to complete the update process. To cancel the update session, press END.
4. Run job DSNTIJUZ

During the update process, you can access all of the installation panels from this panel so that you can view the values that you specified during installation or migration. Parameters that you can update are highlighted. Panels whose fields cannot be updated are marked with an asterisk.



```
DSNTIPB          UPDATE DB2 - SELECTION MENU
===> _

Select one of the following:

 1  DATA PARAMETERS                    15  APPLICATION PROGRAMMING DEFAULTS 1
 2  DEFINE GROUP OR MEMBER             16  PERFORMANCE AND OPTIMIZATION
 3  SYSTEM RESOURCE DATA SET NAMES     17  IRLM PANEL 1
 4  DATA SET NAMES PANEL 1             18  IRLM PANEL 2
 5  DATA SET NAMES PANEL 2 *           19  PROTECTION
 6  DATA SET NAMES PANEL 3 *           20  MVS PARMLIB UPDATES
 7  SIZES *                            21  ACTIVE LOG DATA SET PARAMETERS
 8  SIZES PANEL 2                      22  ARCHIVE LOG DATA SET PARAMETERS
 9  THREAD MANAGEMENT                  23  DATABASES TO START AUTOMATICALLY
10  BUFFER POOL SIZES PANEL 1          24  DISTRIBUTED DATA FACILITY PANEL
11  BUFFER POOL SIZES PANEL 2 *        25  DISTRIBUTED DATA FACILITY PANEL 2
12  TRACING AND CHECKPOINT PARAMETERS  26  ROUTINE PARAMETERS
13  OPERATOR FUNCTIONS                 27  DATA DEFINITION CONTROL SUPPORT
14  APPLICATION PROGRAMMING DEFAULTS 1 28  JOB EDITING

* None of the fields on these panels can be updated.
 PRESS:  ENTER to select   RETURN to exit   HELP for more information
```

Figure 39. Individual update menu panel: DSNTIPB

On the command line, enter a number to select the family of parameters that you want to update. These numbers correspond to the installation panels in Table 46.

Table 46. Panel identifiers

Panel ID	Panel title	See
1. DSNTIPA2	Data Parameters	101
2. DSNTIPK	Define Group or Member	106
3. DSNTIPH	System Resource Data Set Names	109

Table 46. Panel identifiers (continued)

Panel ID	Panel title	See
4. DSNTIPT	Data Set Names Panel 1	113
5. DSNTIPU	Data Set Names Panel 2	118
6. DSNTIPW	Data Set Names Panel 3	127
7. DSNTIPD	Sizes	131
8. DSNTIP7	Sizes Panel 2	137
9. DSNTIPE	Thread Management	140
10. DSNTIP1	Buffer Pool Sizes Panel 1	146
11. DSNTIP2	Buffer Pool Sizes Panel 2	148
12. DSNTIPN	Tracing and Checkpoint Parameters	149
13. DSNTIPO	Operator Functions	155
14. DSNTIPF	Application Programming Defaults Panel 1	163
15. DSNTIP4	Application Programming Defaults Panel 2	171
16. DSNTIP8	Performance and Optimization	176
17. DSNTIPI	IRLM Panel 1	182
18. DSNTIPJ	IRLM Panel 2	188
19. DSNTIPP	Protection	194
20. DSNTIPM	MVS PARMLIB Updates	199
21. DSNTIPL	Active Log Data Set Parameters	204
22. DSNTIPA	Archive Log Data Set Parameters	210
23. DSNTIPS	Databases to Start Automatically	217
24. DSNTIPR	Distributed Data Facility	219
25. DSNTIP5	Distributed Data Facility Panel 2	225
26. DSNTIPX	Routine Parameters	229
27. DSNTIPZ	Data Definition Control Support	232
28. DSNTIPY	Job Editing	235

When the panel that you selected is displayed, enter the new parameters; press the Enter key to return to the Update Selection Menu Panel. Make another panel selection or press Enter again to process. Press End to leave the Update Selection Menu Panel and return to the Main Panel.

Updating other parameters

The following methods modify some of the parameters that you cannot update through the panels:

- To update the CATALOG ALIAS and DEFINE CATALOG fields on DSNTIPA2, see Part 2 (Volume 1) of *DB2 Administration Guide*. The CATALOG ALIAS parameter establishes an alias name for your ICF catalog. This name is also used as the high-level qualifier name for DB2 VSAM data sets. The DEFINE CATALOG parameter controls the creation of the ICF catalog.
- To update DB2 to use the distributed data facility (DDF), follow these steps:
  1. Go through the normal update process of running the CLIST to add DDF information to installation panel DSNTIPR.
  2. Run job DSNTIJUZ.

3. Populate the CDB. See “Step 4: Populate the communications database” on page 464.
4. Stop and start DB2.
5. Bind or rebind these plans:

```
BIND PLAN(DSNESPCS) PKLIST(*.DSNESPCS.DSNESM68)
     ISOLATION(CS) ACTION(REPLACE)

BIND PLAN(DSNESPRR) PKLIST(*.DSNESPRR.DSNESM68)
     ISOLATION(RR) ACTION(REPLACE)
```

6. Start DDF if you specified COMMAND instead of AUTO as the DDF STARTUP OPTION on installation panel DSNTIPR.

- To change the data set sizes for the DB2 catalog and directory:
  1. Copy the catalog and directory table spaces.
  2. Stop the table spaces or their databases.
  3. Delete the data sets and redefine them, using VSAM commands.
  4. Use the RECOVER utility to recover the catalog and directory to the new data sets.
  5. Start the table spaces or databases again.
- To change from single to dual logging for the active log:
  1. Define the second copy of the log with a VSAM IDCAMS DEFINE statement. Refer to job DSNTIJIN, which contains the DEFINE statement for the first copy of the log.
  2. Run the DSNJU003 (change log inventory) utility. This adds the second copy of the log to the BSDS.
  3. Update the NUMBER OF COPIES field on installation panel DSNTIPH from 1 to 2.
  4. Run job DSNTIJUZ to make the change effective.
- To move or expand the bootstrap data sets, use the IMPORT and EXPORT commands of access method services. The bootstrap data sets are accessed using JCL when DB2 starts.
- To access the log data sets, you can use stand-alone access macros or the IMPORT and EXPORT commands of access method service. See Part 4 (Volume 1) of *DB2 Administration Guide* for more information.
- To change the number of data sets for active logs, you can use the DSNJU003 utility. See Part 4 (Volume 1) of *DB2 Administration Guide* for details about the system programmer action in recovery scenarios for active log failures.

Chapter 7. Installing the DB2 subsystem

This chapter describes the jobs you run to install DB2. It also explains how to connect the facilities that allow TSO, batch, IMS, and CICS to access DB2 resources, and how to prepare DB2 for use.

Before you begin, you must perform SMP/E steps 1-12. They are described in Chapter 3, "Loading DB2 libraries," on page 47. You must also run the installation CLIST. It is described in Chapter 6, "Installing, migrating, and updating system parameters," on page 79.

If you plan to have data sharing enabled (Enable option on panel DSNTIPP1), you should save all your jobs from the original installation or migration in a different data set than the enable jobs. If you save them in *prefix*.NEW.SDSNSAMP, the jobs might be overwritten. For more information, see *DB2 Data Sharing: Planning and Administration*.

Before proceeding with the installation steps, refer to the *DB2 Program Directory*, which is shipped with the product, for keyword specifications used for Preventive Service Planning (PSP). Use Information/Access or the ServiceLink facility of IBMLink to check the most current information about DB2 and other products. Contact the IBM Support Center if you do not have access to IBMLink.

You must not use secondary authorization IDs to perform any of the following installation steps.

| After you have completed the following installation steps, your DB2 subsystem
| will be in new-function mode. In new-function mode, all DB2 Version 8 function is
| available for use.

The following topics provide additional information:

- "Installation step 1: Define DB2 to z/OS: DSNTIJMV"
- "Installation step 2: Define the ICF catalog and alias: DSNTIJCA" on page 257
- "Installation step 3: Define system data sets: DSNTIJIN" on page 257
- "Installation step 4: Define DB2 initialization parameters: DSNTIJUZ" on page 259
- "Installation step 5: Initialize system data sets: DSNTIJID" on page 261
- "Installation step 6: Define user authorization exit routines: DSNTIJEX (optional)" on page 262
- "Installation step 7: Record DB2 data to SMF (optional)" on page 263
- "Installation step 8: Establish subsystem security (optional)" on page 264
- "Installation step 9: Connect DB2 to TSO" on page 264
- "Installation step 10: Connect IMS to DB2 (optional)" on page 270
- "Installation step 11: Connect CICS to DB2 (optional)" on page 270
- "Installation step 12: IPL z/OS" on page 270
- "Installation step 13: Start the DB2 subsystem" on page 271
- "Installation step 14: Tailor the DB2 catalog" on page 272
- "Installation step 15: Define temporary work files: DSNTIJTM" on page 272
- "Installation step 16: Define and bind DB2 objects and user-maintained databases: DSNTIJSG" on page 274
- "Installation step 17: Populate the user-maintained databases (optional)" on page 278
- "Installation step 18: Bind the packages for DB2 REXX Language Support: DSNTIJRX (optional)" on page 278
- "Installation step 19: Back up the DB2 directory and catalog: DSNTIJIC" on page 278
- "Installation step 20: Verify the installation" on page 279
- "Installing support for a communications network" on page 279
- "Installing a second DB2 subsystem on the same operating system" on page 280
- "Enabling multiple DB2 subsystems to share disk storage" on page 284
- "Enabling stored procedures after installation" on page 285
- "Enabling MQSeries user-defined functions" on page 292
- "Enabling MQSeries XML user-defined functions and stored procedures" on page 297
- "Loading data with an SQL cursor" on page 300

Installation step 1: Define DB2 to z/OS: DSNTIJMV

This job performs some of the steps that are required to identify DB2 to z/OS. This includes updating members of SYS1.PARMLIB and SYS1.PROCLIB. These data sets are documented in *z/OS MVS Initialization and Tuning Guide*.

If job DSNTIJMV runs successfully, it produces return codes of 0.

z/OS requirements: Each DB2 and each IRLM that you define to z/OS in the IEFSSN*xx* PARMLIB member requires a z/OS system linkage index (LX). The default number of these indexes that z/OS reserves is 165. If you place all of your DB2 and IRLM subsystem definitions in a single IEFSSN*xx* member, you might need more than 165 LXs; otherwise your subsystems might not start. If you need more than 165 LXs, use the NSYSLX option on the z/OS IEASYS*xx* PARMLIB member to increase this number. See *z/OS MVS Initialization and Tuning Guide* for more information.

You must have the prerequisite level of z/OS installed. Do not overwrite the z/OS-supplied entries for DB2 and IRLM in the program properties tables (PPT).

The PPT must contain entries for modules DSNYASCP, DXRRLM00, and DSNUTILB. See Table 47 for a table of parameters for these modules.

The operating system supplies default values for the modules. You should not change these values. If you have modified or deleted the default values, you must enter the original values in the PPT by modifying the SYS1.PARMLIB member SCHED*xx*. Refer to the diagram of the PPT entry in *z/OS MVS Initialization and Tuning Reference*.

Table 47. Parameters for DSNYASCP, DXRRLM00, and DSNUTILB

Entries	Parameters							
DSNYASCP	CANCEL	KEY(7)	NOSWAP	NOPRIV	DSI	PASS	SYST	AFF(NONE)
DXRRLM00	CANCEL	KEY(7)	NOSWAP	NOPRIV	DSI	PASS	SYST	AFF(NONE)
DSNUTILB	CANCEL	KEY(7)	SWAP	NOPRIV	DSI	PASS	NOSYST	AFF(NONE)

IRLM requirements: For later diagnosis of IRLM problems, also ensure that:

- The IRLM dump formatting module name is in control table BLSCECT in SYS1.PARMLIB.
- Load modules DXRRL186 and DXRRLFTB, and the print dump formatting modules DXRRLM50, DXRRLM55, and DXRRLS55 are in SYS1.LINKLIB. If these modules are not in SYS1.LINKLIB, the job that prints the dump must contain a JOBLIB or STEPLIB statement that specifies the library that does contain the modules.

Additional changes to SYS1.PARMLIB and SYS1.PROCLIB: Because different sites have different requirements for identifying DB2 to z/OS, job DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes to SYS1.PARMLIB and SYS1.PROCLIB might be incomplete. You might have additional procedures of your own to rename. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB, or by editing DSNTIJMV.

Recommendation: Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV step. For SYS1.PROCLIB, submit the procedure-update section of job DSNTIJMV. Before you make the updates, read the following information and examine job DSNTIJMV to study the updates that it makes. Then use an editor such as ISPF/PDF to make the updates to SYS1.PARMLIB.

DSNTIJMV updates to SYS1.PARMLIB

Job DSNTIJMV updates the following SYS1.PARMLIB members:

- **IEFSSN***xx*

  This member contains an entry for every z/OS subsystem. DB2 adds to this list of entries, making one entry for DB2 and two entries for the IRLM. The second IRLM entry, whose subsystem name is JRLM, is there to make it easier to add a second IRLM to your system if the first is damaged. You must provide your own procedure to add JRLM. Unique names must be used for each entry.

  z/OS provides subsystem entries for DB2 and IRLM in IEFSSN00. Examine these entries to determine whether they are appropriate for your needs. Make sure that a subsystem name appears only once in the subsystem name list.

  You must ensure that the line that describes the JES subsystem is the first line in
# an IEFSSN*xx* member. The DB2 line should be the next entry.

The DB2 entry has the following format:

```
SUBSYS  SUBNAME(ssname)
        INITRTN(DSN3INI)
        INITPARM
('DSN3EPX,prefix<,scope<,group-attach>>'
```

where:

ssname is the DB2 subsystem name.

DSN3INI is the name of the DB2 load module z/OS invokes during master scheduler initialization. This module must be located in a link list data set (or in SYS1.LINKLIB).

DSN3EPX is the name of the DB2 load module that responds to DB2 requests that are received from the z/OS subsystem interface. (DB2 can be active or inactive when the requests are received.) This module must be located in a link list data set (or in SYS1.LINKLIB).

prefix is the one- to eight-character command prefix.

The first character of the command prefix must be one of the following characters: @ # . / ′) * + - = ¢ < | & ! ; % _ ? : ″. The remaining characters of the command prefix must be one of the above characters, A-Z, or 0-9. See Table 43 on page 200 more information. Do not use the JES2 backspace character or command prefix character. The default is the hyphen (-).

Do not assign a command prefix that is used by another subsystem or that can be interpreted as belonging to more than one subsystem or z/OS application. Specifically, do not specify a multiple-character command prefix that is a subset or a superset of another command prefix beginning from the first character. For example, you cannot assign '-' to one subsystem and '-DB2A' to another. Similarly, you also cannot assign '?DB2' to one subsystem and '?DB2A' to another. You can assign ′-DB2A′ and ′-DB2B′ to different DB2 subsystems.

scope is the one-character scope for the command prefix. DB2 registers its command prefix with the operating system. When this is done, the scope of the command prefix is controlled by the value you choose:

S Started; and the prefix is registered with Sysplex scope at DB2 startup instead of during z/OS IPL. This is the default.

Recommendation: Choose S, which allows you to have a
single IEFSSN*xx* PARMLIB member that all z/OS
systems in the Sysplex can use. Choosing S also
simplifies the task of moving a DB2 from one system to
another; you can stop DB2 on one z/OS and start it up
on another without having to re-IPL the system.

M z/OS system scope; the prefix is registered during z/OS IPL.

X Sysplex scope; the prefix is registered during z/OS IPL. As a result, this DB2 cannot be restarted on another z/OS without changing the definitions and re-IPLing both z/OSs.

For more information about the command prefix facility of z/OS, see *z/OS MVS Planning: Operations*.

group-attach	is the group attachment name, which is used for data sharing. You can specify this on installation panel DSNTIPK.

- **IEAAPF*xx* or PROG*xx***

  Job DSNTIJMV updates IEAAPF*xx* to include the DB2 program libraries (*prefix*.SDSNEXIT, *prefix*.SDSNLOAD, *prefix*.SDXRRESL, and *prefix*.SDSNLINK) as APF-authorized libraries.

  If the program library that contains DFSORT is not already APF-authorized, you can edit DSNTIJMV to authorize it. To do this, you can include the authorization either in this list or in LNKLST*xx*. All libraries that are concatenated with *prefix*.SDSNLOAD in STEPLIB and JOBLIB statements must be APF-authorized. Ensure that the volume serial number in this member is the volume on which the data set resides.

  If you are using the PROG*xx* member instead of the IEAAPF*xx* member, you need to update the member manually—job DSNTIJMV does not edit it.

- **LNKLST*xx***

  Whether you edit the updates directly or edit DSNTIJMV to make the updates, you might first want to review "Choosing link list options" on page 53.

  Job DSNTIJMV updates this member to include the DB2 load module library, *prefix*.SDSNLINK. If you moved the modules from *prefix*.SDSNLINK into another library, edit DSNTIJMV to include that library in the LNKLST*xx* member. If you have combined *prefix*.SDSNLINK and *prefix*.SDSNLOAD into one library, edit DSNTIJMV to include the combined library in the LNKLST*xx*member. See *z/OS MVS Initialization and Tuning Guide* for restrictions on data sets that are concatenated in LNKLST.

  Any data set that is added to the LNKLST*xx* member must be cataloged in the master catalog of the system. This is normally true of *prefix*.SDSNLINK;
  # however, if an alias points to a user catalog when you run DSNALLOC, *prefix*.SDSNLINK is cataloged in a user catalog. In this case, you must either ensure that *prefix*.SDSNLINK is also cataloged in the master catalog or give *prefix*.SDSNLINK a high-level qualifier other than *prefix*, the high-level qualifier for this release. You must give a high-level qualifier other than *prefix* to all release-sensitive data sets that are placed in the LNKLST*xx* member.

  If you do not include the DFSORT library in the LNKLST*xx* member, you must provide a JOBLIB or STEPLIB statement for all utility jobs that include the DFSORT program library. You can accomplish this by placing a STEPLIB statement in DSNUPROC, which appears later in this job. If you use customized modules and exits, *prefix*.SDSNEXIT must precede *prefix*.SDSNLOAD in JOBLIB and STEPLIB statements.

You must do additional editing for the SYS1.PARMLIB updates. If you are editing DSNTIJMV, rather than making the changes directly, you have a choice: either include your additional entries for the SYS1.PARMLIB members (IEAAPF*xx* and LNKLST*xx*) at the end of the existing list of entries, or place them earlier in the list.

If you include them at the end of the existing SYS1.PARMLIB entries, ensure that commas (the continuation character) delimit each entry except the last.

Another SYS1.PARMLIB change to consider at this time is the extended common storage area (ECSA) size, which is specified in the CSA parameter of the IEASYS00 parameter. Ensure that you have specified an adequate size for this subsystem (generally 2 MB plus the MAXIMUM ECSA on installation panel DSNTIPJ if the CROSS MEMORY value is NO).

The **IOP parameter** is another SYS1.PARMLIB change to consider at this time. DB2 can schedule I/O priority. To enable this, you must:

- Use the IOP parameter to set the I/O priority for the address space of a performance group. The IOP parameter is in the IEAIPS*xx* member of SYS1.PARMLIB.
- Enable z/OS I/O priority scheduling by specifying IOQ=PRTY in the IEAIPSxx member of SYS1.PARMLIB.

You must issue an IPL command for z/OS for the PARMLIB updates to take effect. To avoid issuing an IPL command for z/OS during DB2 installation, you can make these updates and issue the IPL in advance of your DB2 installation or migration session. See Part 5 of *DB2 Administration Guide* for more information about I/O priority scheduling.

DSNTIJMV updates to SYS1.PROCLIB

Job DSNTIJMV updates SYS1.PROCLIB to include the DB2 procedures. The procedure names must begin with *xxxx*, the subsystem name, and must end with MSTR, DBM1, or DIST. DSNTIJMV includes the following procedures:

- System services address space startup procedure (*xxxx*MSTR)
- Database services address space startup procedure (*xxxx*DBM1)
- Distributed data facility address space startup procedure (*xxxx*DIST)
- IRLM address space startup procedure (IRLMPROC or user-defined address space name)
- Precompiler procedures
- Utilities procedure (DSNUPROC)
- z/OS Workload Manager (WLM) procedure
- | WLM environment for DSNACICS stored procedure (*xxxx*CICS)

Examine the SYS1.PROCLIB updates carefully. You might want to use a procedure library other than SYS1.PROCLIB for the procedures. Four of the procedures are used for start-up tasks; the other procedures are used to prepare application programs for execution and to invoke DB2 utilities. The program preparation procedures are required for the sample applications and can be helpful in generating other JCL procedures.

Change any data set names that differ at your site. If you specified a suffix on panel DSNTIPA1, that suffix is appended to data sets &USER..DBRMLIB.DATA, &USER..RUNLIB.LOAD, and &USER..SRCLIB.DATA. To override these data set names, you must edit the updates to SYS1.PROCLIB.

The language preparation procedures in job DSNTIJMV use the DISP=OLD
parameter to enforce data integrity. However, when the installation CLIST runs, the
DISP=OLD parameter for the DBRM library data set is modified to DISP=SHR.
This might cause data integrity problems when you run multiple precompiler jobs.
To avoid these data integrity problems, if you are not using DFSMSdfp partitioned
data set extended (PDSE), you must change the language preparation procedures
(DSNHICOB, DSNHFOR, DSNHC, DSNHCPP, DSNHCPP2, DSNHPLI,
DSNHASM, and DSNHSQL) to specify the DISP=OLD parameter instead of the
DISP=SHR parameter.

If compiler STEPLIB statements are needed, add them.

The STEPLIB concatenation of the *xxxx*DBM1 address space procedure includes a commented-out DD for the IBM Language Environment runtime library (SCEERUN). If your system does not include the SCEERUN library in the system linklist, you must uncomment this DD.

Examine the size of the private area on the DB2 start procedures. If necessary, modify the procedures to satisfy the requirements for environmental descriptor manager (EDM) pool size, buffers, number of data sets open, and amount of available private address space. For more information about private address spaces, refer to "Working storage calculation" on page 42.

Installation step 2: Define the ICF catalog and alias: DSNTIJCA

The ICF catalog is the VSAM object in which DB2 records the data sets that you create during the process of installing. Job DSNTIJCA creates the ICF catalog and its alias. DB2 uses the catalog alias as the prefix for your DB2 VSAM data sets.

Running DSNTIJCA is optional. If you specified YES for the DEFINE CATALOG option on installation panel DSNTIPA2, you must run this job to create the catalog. Before running this job, examine the DEFINE UCAT statement carefully to ensure that the parameters are appropriate for your needs.

Do not run this job if you want to use an existing ICF catalog and alias (that is, you specified NO for the DEFINE CATALOG parameter on installation panel DSNTIPA2). However, ensure that the ICF catalog you are going to use is created and that you defined an ICF catalog alias.

If job DSNTIJCA runs successfully, it produces return codes of 0. If DSNTIJCA fails or abends, delete the ICF catalog (if it was created) and rerun the job. To delete the ICF catalog, run job DSNTIJDE as explained on page 258.

Installation step 3: Define system data sets: DSNTIJIN

Job DSNTIJIN defines VSAM and non-VSAM data sets for DB2. It:

- Defines three non-VSAM data sets for the DB2 sample objects:
  prefix.DBRMLIB.DATA
  prefix.RUNLIB.LOAD
  prefix.SRCLIB.DATA
- Defines the VSAM clusters for the bootstrap data sets.

  Each bootstrap data set (BSDS) consists of a VSAM key-sequenced data set. You defined the BSDS names during the ISPF tailoring session.
- Defines the VSAM clusters for the active log data sets.

  You specified up to 31 primary active log data sets during the ISPF tailoring session (NUMBER OF LOGS on installation panel DSNTIPL). You might also have requested dual logging to generate two copies of each active log data set.
  | Consequently, job DSNTIJIN can define up to 62 active log data sets. After
  | installation, you can run the DSNJU003 utility to add additional active log data
  | sets to the BSDS.
- Defines the DB2 directory database.

  Job DSNTIJIN creates and catalogs the DB2 directory database (DSNDB01). The DB2 directory database contains information that is required to start DB2 and is also used by DB2 during its normal operation. It contains table spaces and index spaces that the installation job allocates.
- Defines the DB2 catalog database.

Job DSNTIJIN creates and catalogs the DB2 catalog database (DSNDB06). The DB2 catalog contains information about every object that DB2 maintains.

- Invokes the LISTCAT command of access method service so that you can verify that the VSAM definitions were successful.
- Creates control intervals based on the value you specified in VARY DS CONTROL INTERVAL on panel DSNTIP7. If you specified YES, DB2 creates 4-KB, 8-KB, 16-KB, and 32-KB control intervals as appropriate in the AMS DEFINE CLUSTER commands for the DB2 catalog and directory data sets. If you specified NO, these data sets are created using a fixed control interval of 4-KB.

Check the DEFINE CLUSTER statements in job DSNTIJIN to ensure that they allocate adequate disk space for your system. See Part 5 (Volume 2) of *DB2 Administration Guide* for guidance on allocating and extending data sets.

Recommendation: For recovery purposes, place system data sets such as the DB2 recovery log and the VSAM catalog on different disk volumes. Because these data sets are used frequently, do not migrate them by using DFSMShsm.

If DSNTIJIN runs successfully, it produces return codes of 0 for all DEFINE statements and steps. Check any VSAM messages carefully.

If job DSNTIJIN fails or abends, remove the z/OS catalog delete statements from job DSNTIJDE, run DSNTIJDE (to delete the data sets that are created by DSNTIJIN), and rerun DSNTIJIN.

Deleting DB2 data sets (DSNTIJDE): Job DSNTIJDE is not part of the normal installation process; use this job only for rerunning part of the process. Do not run this job during migration or fallback.

This job deletes the previously created data sets for the DB2 directory and DB2 catalog. If a job fails or abends, you might need to run this job before restarting the DB2 installation process.

In most cases, you must remove or comment out the delete statement in this job for the ICF catalog (if the statement is present). The ICF catalog probably does not need to be deleted and redefined.

Deletes might fail for data sets that do not exist. This does not necessarily indicate that the job failed. If you receive other messages, check them carefully.

Job DSNTIJDE does not work properly if job DSNTIJSG has been executed. This job does not delete the resource limit specification table or the data sets that are used by the distributed data facility.

If job DSNTIJDE fails or abends, correct the error conditions and rerun the job. If you want to delete the ICF catalog, first list its contents and delete the data sets that are cataloged there. This could include sample data sets, user-defined data sets, or subsystem data sets that were not deleted properly. You can use a FORCE command to delete the user catalog.

If you delete the catalog using FORCE before deleting all the data sets, you can use the RECATALOG option of DEFINE CLUSTER and delete the data sets.

Installation step 4: Define DB2 initialization parameters: DSNTIJUZ

Job DSNTIJUZ generates the DB2 subsystem parameter module DSNZPARM (or the name that you specified for PARAMETER MODULE on installation panel DSNTIPO) and the data-only load modules DSNHDECP and DSNHMCID.

Seven macros expand to form the data-only subsystem parameter load module. It contains the DB2 execution-time parameters that you selected using the ISPF panels. These seven macros are DSN6ARVP, DSN6ENV, DSN6FAC, DSN6LOGP, DSN6SPRM, DSN6SYSP, and DSN6GRP.

The DSNTINST CLIST performs calculations on some of the parameter values that you enter during an installation session. The results of these calculations appear in the macro descriptions.

In addition to defining subsystem parameters, job DSNTIJUZ:

- Link-edits the assembled modules into the *prefix*.SDSNEXIT library. Also link-edits the DSNHMCID load module into the *prefix*.SDSNEXIT library.
- Uses the assembler and the DSNHDECM macro to move your subsystem name and the application programming values you specified on installation panels DSNTIPF and DSNTIP4 into another data-only load module called DSNHDECP. DSNHDECP also contains the default SSID from the SUBSYSTEM NAME field on installation panel DSNTIPM. Job DSNTIJUZ includes DSNHDECP in various DB2 load module libraries.
- Link-edits the DSNHDECP module into the *prefix*.SDSNEXIT data set.
- Updates the BSDS with DDF information by using the change log inventory utility. In a data sharing environment, data sharing information is updated, too.
- Uses SMP/E in step DSNTIMQ to read in the edited version of DSNTIJUZ. This action is required to pick up the appropriate includes and library names. After the initial run of step DSNTIMQ, re-running this step is required only when changes have been made to DSNHDECP.
- Uses JCLIN to ensure that DSNHDECP service is placed in all the required load modules.
- Assembles and link-edits the DSNHMCID data-only module that is needed for message conversion by DB2 applications and utilities.

If you added a STEPLIB statement to the DB2 start procedures, modify the SYSLMOD steps to point to the library in the STEPLIB statement instead of to the library in *prefix*.SDSNEXIT.

The subsystem parameter PTASKROL in macro DSN6SYSP, indicates whether to roll up accounting trace records from a parallel query task into the originating task's accounting trace. A value of YES means the originating task is to generate an additional accounting trace record with all the roll-up values from parallel tasks.

Recommendation: Use the default value of YES. A value of NO means that each parallel task produces its own accounting trace record.

The subsystem parameter OJPERFEH, in macro DSN6SPRM, lets you specify whether to disable performance enhancements for outer join operations. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about how to set OJPERFEH.

The subsystem parameters SMSDCFL and SMSDCIX, in macro DSN6SPRM, specify whether SMS data class names are to be used for table spaces and indexes. SMSDCFL and SMSDCIX are one to eight characters long with a default value of a blank. When you use DFSMS and DB2 storage groups, you can use the subsystem parameters SMSDCFL and SMSDCIX to assign table spaces and indexes to different DFSMS data classes.

- SMSDCFL specifies a DFSMS data class for table spaces. If you assign a value to SMSDCFL, DB2 specifies that value when it uses Access Method Services to define a data set for a table space. If the value of SMSDCFL is one or more blanks, DB2 does not specify a data class when it creates data sets for table spaces.
- SMSDCIX specifies a DFSMS data class for indexes. If you assign a value to SMSDCIX, DB2 specifies that value when it uses Access Method Services to define a data set for an index. If the value for SMSDCIX is one or more blanks, DB2 does not specify a data class when it creates data sets for indexes.

The default value for both subsystem parameters is a blank.

Before you set the data class system parameters, you need to define the data classes for your table space data sets and index data sets. You also need to code SMS automatic class selection (ACS) routines to assign indexes to one SMS storage class and table spaces to a different SMS storage class. For more information on creating data classes, see *z/OS DFSMS: Implementing System-Managed Storage*.

The subsystem parameter NPGTHRSH, in macro DSN6SPRM, lets you specify that DB2 is to use special access path selection for tables under a given size. The default value is 0, which means that no special access path selection is used. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information on how to set NPGTHRSH.

The subsystem parameter STARJOIN and SJTABLES, in macro DSN6SPRM, lets you specify when DB2 is to enable star join processing. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information on how to set STARJOIN and SJTABLES.

The subsystem parameter DISABSCL in macro DSN6SPRM allows you to choose whether SQLWARN1 and SQLWARN5 are set for non-scrollable cursors on OPEN and ALLOCATE CURSOR. The default value is NO, which means that SQLWARN1 and SQLWARN5 are set.

Subsystem parameter INLISTP, in macro DSN6SPRM, allows you to specify the maximum number of elements in an IN-list for certain IN-list predicate optimizations to occur. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information on how to set INLISTP.

The subsystem parameter SMF89, in macro DSN6SYSP, lets you specify whether DB2 is to do detailed tracking for measured usage pricing. The default value is NO, which means that DB2 does not do detailed measured usage tracking. If the SMF type 89 record is activated, only high-level tracking is recorded in the SMF type 89 record. Selecting NO reduces CPU usage, but also increases the amount of time spent in DB2 as measured by SMF 89. If you select YES, DB2 does detailed measured usage tracking if SMF type 89 records are activated. When SMF89 is set to YES, DB2 invokes a z/OS service on every entry or exit into or out of DB2 to ensure accurate tracking.

Recommendation: Select SMF89 YES only if you use measured usage pricing.

Subsystem parameter COMCRIT, in macro DSN6SPRM, sets the Common Criteria
environment. The default value is NO, which means the behavior of DB2 is
unchanged. When the value of COMCRIT is YES, all tables that you create (other
than created global temporary tables, declared global temporary tables, and
auxiliary tables) must have multilevel security. If the AS SECURITY LABEL clause
is missing from a table, an error occurs and the table is not created. Setting
COMCRIT to YES will cause some of the current installation and migration
processes to fail. You can change the value of COMCRIT online by using SET
SYSPARM and you can audit COMCRIT with IFCID 0106.

Subsystem parameter SUPPRESS_TS_CONV_WARNING, in macro DSN6SPRM,
indicates whether DB2 should suppress the SQLCODE +20272 warning whenever
DB2 implicitly converts a table space from index-controlled to table-controlled
partitioning. The default value is NO, which means that DB2 issues the SQLCODE
+20272 warning. If you specify YES, DB2 suppresses the SQLCODE +20272
warning.

If the DB2 distribution library prefix is different from the target library prefix, edit DSNTIJUZ to correct the data set name for *prefix*.ADSNLOAD.

If you have not run the SMP/E ACCEPT job (DSNTIJAC) of FMID HDB8810, you must edit DSNTIJUZ so that the SMP/E temporary data set (SMPTLIB) is included in the concatenation for the ADSNLOAD DD statement in steps DSNTIZL and DSNTIZQ.

You might receive message GIM65001 when you run steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when you run step DSNTIMQ. You can ignore these messages.

If job DSNTIJUZ fails or abends, correct the problem and rerun the job.

Installation step 5: Initialize system data sets: DSNTIJID

Job DSNTIJID initializes VSAM data sets for DB2. It performs these functions:

- Initializes the BSDS by invoking the change log inventory utility.
- Initializes the DB2 directory database using data in the untailored version of *prefix*.SDSNSAMP.
- Initializes the DB2 catalog database using data from the untailored *prefix*.SDSNSAMP.
- Invokes the DSNJLOGF utility to preformat active log data sets that are created during installation. See *DB2 Utility Guide and Reference* for more information on DSNJLOGF.

If job DSNTIJID runs successfully, it produces return codes of 0. If you receive any VSAM messages, check them carefully. If DSNTIJID fails or abends, remove the catalog delete statements from job DSNTIJDE, run job DSNTIJDE, and then rerun DSNTIJIN and DSNTIJID.

Installation step 6: Define user authorization exit routines: DSNTIJEX (optional)

Job DSNTIJEX builds the sample authorization exit routines DSN3@SGN and DSN3@ATH, and the user version of the access control authorization exit routine, DSNX@XAC, from the source code in *prefix*.SDSNSAMP. Job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. Then DSNTIJEX places the exit routines in the *prefix*.SDSNEXIT library. The DB2 CLIST tailors the JCL in DSNTIJEX to match your site's environment.

The sample authorization exit routines are not the same as the default authorization exit routines that are supplied by DB2. By implementing the sample authorization exit routines, you can provide group names as secondary authorization IDs. By modifying the sample authorization exit routines, you can tailor authorization processing for your subsystem.

DSNXSXAC is a copy of the default access control authorization exit routine that users can modify. This exit routine allows you to bypass some or most of DB2 authorization checking to specify your own authorization checking. If you do not modify it, this step is not needed and you should delete it.

DSNACICS is a stored procedure that invokes user exit routine DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. If you do not need to modify the caller's parameter values, you can use the default DSNACICX exit routine. However, if you need to modify the caller's parameter values, you need to perform the following tasks:

1. Write a user exit routine in assembler, COBOL, C, or PL/I
2. Assemble or compile the source code
3. Link-edit the object code into the DB2 exit routine library

Installation job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX. You can use this step as a model for your program preparation job.

For information about writing exit routines, see Appendix B (Volume 2) of *DB2 Administration Guide*. For more information on controlling data access, see Part 3 (Volume 1) of *DB2 Administration Guide*.

You have the following options regarding exit routines:

- To use the sample authorization exit routines, run job DSNTIJEX.
- To use the default authorization exit routines, skip job DSNTIJEX.
- To use the modified sample authorization exit routines, modify DSNTIJEX to reference the correct library before you run it.

If you will use the RACF/DB2 external security module (DSNXRXAC) as your DB2 access control authorization exit, modify DSNTIJEX to refer to DSNXRXAC instead of DSNXSXAC. See *DB2 RACF Access Control Module Guide* for more information.

If job DSNTIJEX runs successfully, it produces return codes of 4.

If job DSNTIJEX fails or abends, correct the problem and rerun the job.

Installation step 7: Record DB2 data to SMF (optional)

When you install DB2, you can specify if DB2 statistical, accounting, and audit trace data are to be collected. To have DB2 collect:

- Statistical information, accept the default (YES class 1) for the SMF STATISTICS option on installation panel DSNTIPN. To collect statistical information for deadlock or timeout, specify class 3. To collect information about DDF error conditions, specify class 4.
- Accounting information, accept the default (1) or specify * (all classes) for the SMF ACCOUNTING option on installation panel DSNTIPN.
- Auditing information, specify * (all classes) for the AUDIT TRACE option on installation panel DSNTIPN.

In all cases, DB2 invokes a trace, passing the data it collects to the System Management Facility (SMF) of z/OS. For more information about the START TRACE command, see *DB2 Command Reference*.

DB2 also passes performance data to SMF whenever an accounting, statistics, or audit trace is successfully started or stopped. DB2 can also record other performance data. After you complete the installation process, you can use commands to have DB2 record performance data for over 230 different subsystem events.

See *DB2 Command Reference* to see what kind of data DB2 can collect and pass to SMF.

You must make some additional updates if, during installation, you requested that DB2 pass accounting and statistics data to SMF. Specifically, you must update the SMFPRM*xx* member of SYS1.PARMLIB as follows:

- Specify the ACTIVE parameter.
- Specify the proper TYPE subparameter of SYS and SUBSYS.

During DB2 execution, you can use the SMF SET or SS command to alter the SMF parameters. For example, you can record the statistics trace class 1 IFCIDs 0001, 0002, and 0202 (SMF record type 100); accounting trace class 1 IFCIDs 0003 and 0239 (SMF record type 101); and all other DB2 trace records (SMF record type 102) to SMF. To record this information, issue the following command:

```
SYS(TYPE(100:102))
```

For DB2 to pass data to SMF, you must allocate an adequate supply of SMF buffers. The default buffer settings are probably insufficient.

You can specify SMF buffering on the VSAM BUFSP parameter of the Access Method Services DEFINE CLUSTER statement. Do not use the default settings if DB2 data is sent to SMF. Specify CISZ(4096) and BUFSP(81920) on the DEFINE CLUSTER statement for each SMF VSAM data set. These values for CISZ and BUFSP are the minimum requirement for DB2. You might need higher values for CISZ and BUFSP, depending on the requirements of all your z/OS subsystems.

You can also code an IEFU84 SMF exit routine to process the records that are produced.

Detailed information about starting and stopping DB2 traces for performance, accounting, audit, and statistics data is provided in Part 5 (Volume 2) of *DB2 Administration Guide*.

For more information about SMF, refer to *z/OS MVS Initialization and Tuning Reference* and *z/OS MVS Initialization and Tuning Guide*.

Installation step 8: Establish subsystem security (optional)

DB2 can control access to data within DB2. It also works together with outside security systems, such as RACF, that control access to the DB2 subsystem. See Part 3 (Volume 1) of *DB2 Administration Guide* for suggestions and instructions for including DB2 in your security system.

Installation step 9: Connect DB2 to TSO

Although you can eventually connect DB2 to IMS, CICS, or both, connecting initially only to TSO is recommended for a first-time installation. At this point, you can run the sample applications that do not require CICS or IMS, allowing your database and system administrators to gain familiarity with the administrative facilities of this version of DB2.

If you have previously installed DB2 and are performing that task again, your database and system administrators are probably already familiar with DB2. In this case, you can connect IMS, CICS, or both, at the same time that you connect TSO. You can then run the sample applications that require CICS and IMS at the same time that you run the sample applications for TSO and batch.

To attach DB2 to TSO:
1. Make DB2 load modules available to TSO and batch users.
2. Make DB2 CLISTs available to TSO and batch users.
3. Make PL/I options available (if applicable).
4. Make panels, messages, and load modules available to ISPF and TSO.
5. Connect the DB2I panels to the ISPF Main Panel.
6. Establish TSO and RACF user IDs for DB2 users.

These tasks are explained in the following topics.

Make DB2 load modules available to TSO and batch users

If you included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, you can skip this step.

If you have not included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, you must add STEPLIB statements to your logon procedures and to the JCL for jobs to ensure that you access the DB2 Version 8 load modules.

If *prefix*.SDSNEXIT is not in your LINK*xx*, add it to your STEPLIB and JOBLIB concatenations before *prefix*.SDSNLOAD. Refer to “Choosing link list options” on page 53 for information about link lists.

Make DB2 CLISTs available to TSO and batch users:
DSNTIJVC

From *prefix*.SDSNCLST, the CLIST reads and edits these four CLISTs: DSNEMC01, DSNH, DSNU, and DSNHC. It then places those CLISTs in *prefix*.NEW.SDSNTEMP. You might want to modify the default values. See “Completing the CLIST processing” on page 243 for information on the items can modify. The DSNEMC01 CLIST provides installation default values for the DB2I Default panels (option D on panel DSNEPRI). The first time that a TSO user executes DB2I on a specific ISPF application such as DSNE (NEWAPPL), the DSNEMC01 CLIST sets the defaults based on the values that are specified on installation panel DSNTIPF. DSNEMC01 stores the values in the ISPF profile member DSNEPROF.

Job DSNTIJVC merges the tailored CLISTs from *prefix*.NEW.SDSNTEMP with unchanged CLISTs and REXX execs from *prefix*.SDSNCLST. Then job DSNTIJVC places all CLISTs and REXX execs in *prefix*.NEW.SDSNCLST. It also converts the record format of the DB2 CLISTs from fixed-block to variable-block format with a record length of 84 and a block size of 3120.

If you use fixed-block format for your CLIST libraries, modify job DSNTIJVC as follows:

- Change the SYSIN DD statement to DUMMY.
- Change the allocation of *prefix*.NEW.SDSNCLST to match the data control block (DCB) attributes of your other CLIST libraries.

A CLIST that has been converted from fixed-block to variable-block format cannot be used as input to the DSNTINST CLIST; use the unedited version of the SDSNCLST data set, as created by SMP/E.

To make the CLISTs available to TSO and batch users, you must either concatenate *prefix*.NEW.SDSNCLST with your existing CLIST libraries or copy *prefix*.NEW.SDSNCLST into an existing CLIST library.

If you need to rerun this job, first delete data set *prefix*.NEW.SDSNCLST, which is created by this job.

When corrective service is applied to a CLIST, SMP/E changes only the *prefix*.SDSNCLST data set. You need to redo any record format changes and reapply any special tailoring that is required. You also need to move the CLIST to *prefix*.NEW.SDSNCLST. Corrective service (program temporary fixes) for these CLISTs is sent with ++HOLD statements, calling to your attention the possibility of additional work.

Ensure that PL/I options are available

If you are using PL/I, ensure that the options that your DB2 programmers use are included in the compiler. Restrictions that are imposed by your site on PL/I compiler options affect how you can use DB2 program preparation. The program preparation function uses the following options:

```
FLAG    OBJECT    SOURCE    TERMINAL    XREF
```

If the macro pass is used, the following options are also needed:

```
MACRO    MDECK    SYNTAX
```

Make panels, messages, and load modules available to ISPF
and TSO

You must concatenate the DB2 ISPF libraries with the ISPPLIB, ISPSLIB, and ISPMLIB DD statements in your logon procedures and in any of your CLISTs where they might be allocated. These libraries are *prefix*.SDSNSPFP, *prefix*.SDSNSPFM, *prefix*.SDSNSPFS, and either *prefix*.SDSNPFPE or *prefix*.SDSNPFPK depending on whether you are using English or Kanji DB2I panels. If you are using online help, include *prefix*.SDSNSPFT.

DB2I uses the ISPF PROFILE and SHARED variable pools for most panel variable fields. As a result, you can easily re-enter a panel when panel variables have previously been specified. For the DB2 subcommands that permit lists of plan names, package names, DBRMs, and ENABLE and DISABLE statements, DB2I provides ISPF to contain all the user-specified variables for these subcommand keywords.

DB2I creates and maintains a set of ISPF tables in a user-defined TSO data set that is allocated to a data set with a ddname of DSNETBLS. The DB2I-generated tables in this library are DSNCONNS, DSNDBRMS, and DSNPLPKN. Table 48 shows the library table member names and their contents.

Table 48. The DB2 ISPF table library

DSNCONNS	ENABLE or DISABLE connection type and connection name variables that are referenced by plan or package name
DSNDBRMS	Subcommand DBRM member and LIBRARY name variables that are referenced by plan name
DSNPLPKN	Package list variables that are referenced by package name
DSNPATHS	Schema name variables that are referenced by plan or package name and that are to be included in the PATH keyword

When allocating this data set, you should assign the following DCB attributes, where `n` is any integer:

```
DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(n*LRECL)
```

The following example shows how to set up an ALLOCATE statement to create the data set:

```
ALLOC DA(DSNSPFT) NEW SP(1 1) TR DIR(10) +
DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(800)
F(DSNETBLS) REUSE
```

The following example shows how to allocate an existing data set to the data set with the DSNETBLS ddname:

```
ALLOC DA(DSNSPFT)      F(DSNETBLS) REUSE
```

Edit SYS1.HELP(COMMANDS) to add the DSN command to the HELP list of available TSO commands. Also, add a line indicating that the DSN command allows you to perform DB2 functions from TSO. SYS1.HELP(COMMANDS) is help information only; it describes DB2 function, but it does not provide that function.

DB2I uses ISPF table services to maintain individual ISPF tables within the DSNETBLS data set. For performance reasons, ISPF keeps this table library in an **open** state once an individual table has been updated. Attempts to **close** this data set by using the TSO FREE command results in error message IKJ56861I.

For additional information on this TSO error message and how to **close** this data set, refer to *z/OS ISPF Messages and Codes*.

If you want to run the ISPF/CAF sample application that is provided with DB2, ensure that the data set *prefix*.RUNLIB.LOAD is included in the logon procedures or in the ISPLLIB concatenation. If you chose IBMCOB for your LANGUAGE DEFAULT on panel DSNTIPF, you should also include the data set *prefix*.CEE.SCEERUN. For more information about the ISPF/CAF sample application, see “Running dynamic SQL and the ISPF/CAF application” on page 384.

Connect DB2I panels to the ISPF main panel

This topic explains how to connect the DB2 panels to the standard ISPF panels that are already installed on your system.

Recommendation: Use the following panels for establishing the connection.

- ISP@MSTR, ISR@PRIM, or ISRFPA for the connection to DB2 Interactive services
- ISR00003 for the tutorial menu update
- See your TSO administrator for other possibilities.

Two example panels are provided here. Their names are DSNTIPRM (the DB2 version of an ISPF primary options panel) and DSNTIPTU (the DB2 version of a tutorial table of contents). Using the TSO RENAME command, give DSNTIPRM an alias of ISR@PRIM, and give DSNTIPTU an alias of ISR00003. For example:

```
RENAME 'prefix.SDSNSPFP(DSNTIPRM)' (ISR@PRIM) ALIAS
RENAME 'prefix.SDSNSPFP(DSNTIPTU)' (ISR00003) ALIAS
```

If the DB2 panel library is concatenated before the standard ISPF library, the connection is made.

If your site has made changes to either of these panels, change your existing panels rather than use the following examples. The panels in Figure 40 on page 268 and Figure 41 on page 269 display the needed modifications.

```
)ATTR
/*********************************************************************/
/* COPYRIGHT = 5740-XYR (C) COPYRIGHT IBM CORP 1982, 1985, 1990, 2003 */
/* REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083             */
/* STATUS = VERSION 8, LEVEL 0                                       */
/*********************************************************************/
)BODY
%-----------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
%OPTION  ===>_ZCMD                                        +USERID   - &ZUSER .
%   0 +ISPF PARMS  - Specify terminal and user parameters   +TIME     - &ZTIME .
%   1 +BROWSE      - Display source data or output listings +TERMINAL - &ZTERM .
%   2 +EDIT        - Create or change source data           +PF KEYS  - &ZKEYS .
%   3 +UTILITIES   - Perform utility functions
%   4 +FOREGROUND  - Invoke language processors in foreground
%   5 +BATCH       - Submit job for language processing
%   6 +COMMAND     - Enter TSO command or CLIST
%   7 +DIALOG TEST - Perform dialog testing
%   8 +DB2I        - Perform DATABASE 2 Interactive functions
%   C +CHANGES     - Display summary of changes for this release
%   T +TUTORIAL    - Display information about ISPF/PDF
%   X +EXIT        - Terminate ISPF using log and list defaults
%
+Enter%END+command to terminate ISPF.
%
 )INIT
   .HELP = ISR00003
   &ZPRIM = YES        /* ALWAYS A PRIMARY OPTION MENU      */
   &ZHTOP = ISR00003   /* TUTORIAL TABLE OF CONTENTS        */
   &ZHINDEX = ISR91000 /* TUTORIAL INDEX - 1ST PAGE         */
 )PROC
   &ZSEL = TRANS( TRUNC (&ZCMD'.')
                 0,'PANEL(ISPOPTA)'
                 1,'PGM(ISRBRO)'
                 2,'PGM(ISREDIT)'
                 3,'PANEL(ISRUTIL)'
                 4,'PANEL(ISRFPA)'
                 5,'PGM(ISRJB1) PARM(ISRJPA) NOCHECK'
                 6,'PGM(ISRPTC)'
                 7,'PGM(ISRYXDR) NOCHECK'
                 8,'CMD(DSNECPRI) NEWAPPL(DSNE)'
                 C,'PGM(ISPTUTOR) PARM(ISR00005)'
                 T,'PGM(ISPTUTOR) PARM(ISR00000)'
               ' ',' '
                 X,'EXIT'
                 *,'?' )
   &ZTRAIL = .TRAIL
 )END
```

Figure 40. ISPF primary option panel (DSNTIPRM), edited to include DB2I

Figure 40 shows panel DSNTIPRM. Notice the added lines in boldface type. Adding these lines allows you to invoke the DB2 Interactive (DB2I) functions. The added lines include one displayed line:

```
%   8 +DB2I        - Perform DATABASE 2 Interactive functions
```

You can also choose to add one of these undisplayed lines:

```
8,'CMD(DSNECPRI) NEWAPPL(DSNE)'
8,'CMD(DSNECPRI SSID(xxxx)) NEWAPPL(DSNE)'
```

The displayed line lets the user choose DB2I. Both of the undisplayed lines invoke the DB2I main panel (DSNEPRI). If you use the first undisplayed line, you accept the default for the subsystem identifier (SSID) parameter. If you use the second undisplayed lines, you can specify a different SSID parameter.

DSNECPRI is a CLIST and can be invoked directly from another user CLIST. It is an alternative way to invoke DSNEPRI without updating the primary ISPF panel.

By specifying NEWAPPL(DSNE), you define DSNE as the ISPF application that DB2I uses. ISPF uses the name DSNE to create the ISPF profile pool member name (DSNEPROF) in the TSO_*userid*.ISPPROF data set, which contains all ISPF panel variables defined during DB2I execution. Any customized DSNEPROF members can be migrated from Version 8.

Recommendation: Examine any new or changed default panel values to ensure that your custom values are still valid, specifically the option values for the subcommands BIND PLAN, REBIND PLAN, BIND PACKAGE, and REBIND PACKAGE.

Using a NEWAPPL name other than DSNE: You can define any valid ISPF application name. If you define an application name to be something other than DSNE and you plan to use DB2 online help, you must create a new command table member that corresponds to your application name. You can create a new command table name by using the TSO Command Table Utility and specifying the Verb as Help and the Action as `Select Panel(&Helppan)` after defining your new ISPF application name.

```
)ATTR
/*******************************************************************/
/* COPYRIGHT = 5740-XYR (C) COPYRIGHT IBM CORP 1982, 1985, 1990, 2003 */
/* REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083             */
/* STATUS = VERSION 8, LEVEL 0                                       */
/*******************************************************************/
)BODY
 %TUTORIAL -------------------- TABLE OF CONTENTS ------------------ TUTORIAL
 %OPTION  ===>_ZCMD

                 ---------------------------------------------
                 | ISPF PROGRAM DEVELOPMENT FACILITY TUTORIAL |
                 |             TABLE OF CONTENTS              |
                 ---------------------------------------------
    The following topics are presented in sequence, or can be selected by
    entering a one-character selection code in the option field on line 2:
       %G+ GENERAL     - General information about ISPF
       %0+ ISPF PARMS  - Specify terminal and user parameters
       %1+ BROWSE      - Display source data or output listings
       %2+ EDIT        - Create or change source data
       %3+ UTILITIES   - Perform utility functions
       %4+ FOREGROUND  - Invoke language processors in foreground
       %5+ BATCH       - Submit job for language processing
       %6+ COMMAND     - Enter TSO command or CLIST
       %7+ DIALOG TEST - Perform dialog testing
       %8+ DB2         - Information about DB2
       %X+ EXIT        - Terminate ISPF using log and list defaults
    The following topics are presented only if explicitly selected:
       %A+ APPENDIX A  - Dynamic allocation interface routine (DAIR) errors
       %B+ APPENDIX B  - ISPF listing formats
       %I+ INDEX       - Alphabetic index of tutorial topics
 )PROC
    &ZSEL = TRANS(&ZCMD
                 G,ISR01000
                 0,ISP05000
                 1,ISR10000
                 2,ISR20000
                 3,ISR30000
                 4,ISR40000
                 5,ISR50000
                 6,ISR60010
                 7,ISR70000
                 8,DSN4V2DB
                 X,ISP90100
                 A,*ISP93030
                 B,*ISR95000
                 I,*ISR91000
                 )
 )END
```

Figure 41. ISPF program development facility tutorial panel (DSNTIPTU), edited to include DB2 tutorial

Figure 41 shows panel DSNTIPTU. Notice the two added lines in boldface type. Adding these lines allows you to invoke the DB2 tutorial panels. One of the two added lines is displayed:

```
%8+ DB2         - Information about DB2
```

The other added line is undisplayed:

```
8,DSN4V2DB
```

The displayed line presents the user with a choice for the DB2 tutorial. The undisplayed line actually invokes the DB2 tutorial menu (DSN4V2DB).

For information about ISPF, see *ISPF V4 Dialog Developer's Guide and Reference* and *ISPF V4 User's Guide*.

Establish DB2 authorization IDs in TSO and RACF

You specified the following IDs on installation panel DSNTIPP:

- Two system administrator (installation SYSADM) authorization IDs
- Two system operator (installation SYSOPR) authorization IDs
- One authorization ID (installation IBMUSER) if RACF is not available for batch access and USER= is not specified in the job statement.

Before attempting to access DB2, be sure that the installation SYSADM IDs that you specified are defined in your TSO and RACF systems. You also can define installation SYSOPR IDs there, as well as the installation IBMUSER ID.

Installation step 10: Connect IMS to DB2 (optional)

Connecting DB2 to IMS requires coordination with your IMS support group. To connect the IMS attachment facility:

- Make DB2 load modules available to IMS.
- Define DB2 to IMS.
- Define new application programs and transactions to IMS.
- Prepare IMS applications for DB2.

Depending on your site, you might also need to:

- Define DB2 plans for IMS applications.
- Generate a user language interface.

These tasks are explained in Chapter 11, "Connecting the IMS attachment facility," on page 439. This chapter also explains the requirements from a DB2 perspective and refers you to IMS books for specific IMS information.

Installation step 11: Connect CICS to DB2 (optional)

| To connect DB2 to CICS, you must regenerate several CICS tables with additional
| entries. Coordinate this connection with your CICS support group. See *CICS
| Transaction Server for z/OS DB2 Guide* for more information about connecting CICS
| to DB2.

Installation step 12: IPL z/OS

The first time that you start DB2, you must IPL z/OS, and then start the DB2 subsystem.

The load module library SDSNLINK contains the early code. SDSNLINK contains modules that must be placed in the link list look-aside address space (LLA) because they are loaded at subsystem initialization during the IPL.

The z/OS IPL is necessary because installation job DSNTIJMV makes changes to SYS1.PARMLIB that are not recognized by z/OS until the next IPL. DSNTIJMV makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member

- Creates new load module libraries in the LNKLST*xx* member

Complete these changes before performing the IPL. For more information, refer to "Installation step 1: Define DB2 to z/OS: DSNTIJMV" on page 252 and "Choosing link list options" on page 53.

During the z/OS IPL, message DSN3100I appears on the z/OS console, stating that DB2 is ready for the START command.

Installation step 13: Start the DB2 subsystem

Perform the following steps to start DB2:

1. Start the IRLM, if you have not requested that DB2 automatically start the IRLM. You must start IRLM before you start DB2.

   Use the following command:

   ```
   START irlmproc
   ```

   where *irlmproc* is the name that you specified for the PROC NAME option on IRLM Panel 1 (DSNTIPI).

   If you specified YES for the AUTO START option on IRLM Panel 1 (DSNTIPI), DB2 starts the IRLM automatically.

2. Start DB2 from the z/OS console. Use the following command:

   ```
   -DSN1 START DB2
   ```

   where (-DSN1) is the subsystem command prefix that you defined for DB2.

   DB2 uses the subsystem parameter module that is specified in the start-up JCL procedure in SYS1.PROCLIB:

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

   where *DSNZPxxx* is the value that you specified for PARAMETER MODULE on panel DSNTIPO.

   If you need to change the *DSNZPxxx*, you can edit SYS1.PROCLIB. Alternatively, you can override the *DSNZPxxx* by using the PARM option as follows:

   ```
   -DSN1 START DB2, PARM(DSNZPxxx)
   ```

   If DB2 starts successfully, two to four address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, and possibly *ssnm*DIST, and *irlmproc*, where *ssnm* is the DB2 subsystem name and *irlmproc* is the IRLM procedure name.

   If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

   ```
   DSNR002I   RESTART COMPLETED
   DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
   ```

   # You must run job DSNTIJTC to complete the tailoring of the DB2 catalog. In a
   # data sharing environment, do not run DSNTIJTC after installing
   # non-originating members. See "Installation step 14: Tailor the DB2 catalog" on
   # page 272 for more information about tailoring the DB2 catalog.

   # When you start DB2 Version 8 for the first time, DB2 issues message DSNT501I
   # with reason code 00C900A6. This message is expected. When you run job
   # DSNTIJTC in "Installation step 14: Tailor the DB2 catalog" on page 272, the
   # cause of this message is corrected.

   After you start DB2, identify unusual conditions for databases with the command:

```
-DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
```

If DB2 does not start successfully, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Ensure that the DSNTIJUZ, DSNTIJIN, and DSNTIJID jobs ran correctly. Also, verify that the subsystem parameter member that you specified (or allowed by default) when you started DB2 is the one that is built by job DSNTIJUZ. Check the JCL for the DB2 startup procedure.

Note to distributed data facility users: You must define VTAM before DDF can start. However, you do not need to have TCP/IP configured to start DDF. In addition, transactions such as those from DDF or CICS may fail because work files have not yet been defined.

3. Optionally, start TSO.

   If you want to use the TSO SUBMIT command to do housekeeping and installation verification, you must start TSO (if it is not already started).

4. Take one of the following actions to enable primary and secondary user IDs to issue DB2 commands from the z/OS console or TSO SDSF:
   - Grant SYSOPR authority to all primary and secondary user IDs that issue DB2 commands from the z/OS console or TSO SDSF. Issue the following command:

     ```
     GRANT SYSOPR TO userid
     ```
   - Define RACF classes to authorize DB2 commands. Use the following statements:

     ```
     SETR CLASSACT(DSNADM)
     RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
     SETR RACLIST(DSNADM) REFRESH
     PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
     SETR RACLIST(DSNADM) REFRESH
     ```

Installation step 14: Tailor the DB2 catalog

DSNTIJTC invokes the CATMAINT utility to tailor your Version 8 catalog. DSNTIJTC contains one step that performs site-specific tailoring of the catalog. You must tailor the catalog to fully install DB2.

Important: In a data sharing environment, do not run DSNTIJTC after installing non-originating members.

A status message, DSNU777I, is issued at several points to indicate progress. Diagnostic error messages are issued when CATMAINT processing fails. If a problem is found during the SQL processing phase of installation, message DSNU778I is issued. If non-supported functions are encountered, message DSNU776I is issued. All of these messages are written to the SYSPRINT data set. See the message descriptions in *DB2 Messages* for details.

To execute DSNTIJTC, you must have installation SYSADM authority.

Installation step 15: Define temporary work files: DSNTIJTM

The DSNTIJTM job defines the database for temporary work files and provides some cleanup. For non-data-sharing installations, the work file database is DSNDB07. This job assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically, and DSNTWR, which is the load module for the WLM_REFRESH stored procedure. It also defines the initial buffer pool sizes as specified on installation panels DSNTIP1 and DSNTIP2.

For data sharing installations, the work file database is the name that you specified for WORK FILE DB on installation panel DSNTIPK. When you use the ENABLE or MEMBER functions, the steps to bind the DSNTIAD program are unnecessary and are removed during editing.

You must ensure that the installation jobs run on the same z/OS system on which the appropriate DB2 subsystem is running. See 235 for more information.

The DSNTIJTM job creates data sets in the work file database. These data sets are to be used as working storage for table spaces that require 4-KB and 32-KB buffering.

To create these data sets, the DSNTIJTM job uses as input the values that you specified for the TEMP 4K SPACE option and the TEMP 32K SPACE option on the Sizes Panel (DSNTIPD).

To specify the number of 4-KB and 32-KB table spaces, the DSNTIJTM job uses as input the values that you specified for TEMP 4K DATA SETS and TEMP 32K DATA SETS on the same panel. The naming convention for these data sets is provided by the installation CLIST:

```
vcatalog.DSNDBC.DSNDB07.DSNkknn.I0001.A001
```

where *vcatalog* is the catalog alias name that you specified for the CATALOG ALIAS field on installation panel DSNTIPA2, *DSNDB07* is the name of the work file database, *kk* is either 4K or 32K, *nn* is the number of the data set, and *y* is I or J. For example, if you specify 2 as the number of temporary 4-KB data sets and 1 as the number of temporary 32-KB data sets in a non-data-sharing environment, DSNTIJTM defines the following table spaces:

```
vcatalog.DSNDBC.DSNDB07.DSN4K01.y0001.A001
vcatalog.DSNDBC.DSNDB07.DSN4K02.I0001.A001
vcatalog.DSNDBC.DSNDB07.DSN32K01.I0001.A001
```

You can increase the number of additional temporary work file table spaces, increasing the values that you specify for the TEMP 4K DATA SETS or TEMP 32K DATA SETS options, particularly if you expect a great deal of sorting at your site. Additional temporary work file table spaces can improve DB2 performance by reducing device contention among applications. These additional work files also can be used for sorting indexes on large tables during index creation. For more information on adding temporary work files, see "Work file database storage requirements" on page 22. You can choose to have job DSNTIJTM create these additional table spaces, or you can create them after you run the job.

| During installation, if you specify YES for VARY DS CONTROL on panel
| DSNTIP7, DB2 tailors job DSNTIJTM to include 4-KB or 32-KB control intervals as
| appropriate in the AMS DEFINE CLUSTER commands for the DB2 catalog and
| directory data sets. If you specify NO for this parameter, these data sets are created
| with a fixed control interval of 4-KB.

To create the temporary work file table spaces after you run job DSNTIJTM, comment out all steps except DSNTTMP and DSNTIST from the job. Remove the COND statements, and change DSN4K01, DSN32K01, or both, to the table space names that you want to use. Then, run the edited steps.

If job DSNTIJTM runs successfully, it produces the return codes shown in Table 49:

Table 49. DSNTIJTM return codes

Step	PROCSTEP	Return code
DSNTIAD	PC ASM LKED	0000 0000 0000
DSNTWR	PC ASM LKED	0000 0000 0000
DSNTIAB	(none)	0000
DSNTIAS	(none)	0000 or 0012
DSNTICR	(none)	0000
DSNTTMP	DSNTIC	0000
DSNTIST	(none)	0000

The first time that you run the DSNTIJTM job, you will probably receive a return code of 12 for step DSNTIAS. This is because step DSNTIAS issues a command to stop DSNDB07 (the work file database), which has not yet been defined. A return code of 0 is issued if the job failed the first time and needs to be rerun. When migrating, you also might receive a return code of 12 if the work file database DSNDB07 was dropped before job DSNTIJTM runs.

You might receive a return code of 12 for step DSNTTMP if you have already run it.

Because this is the first use of DB2, errors from earlier steps might be detected here.

If you receive an abend reason code from the data manager (X'00C9'*xxxx*) or buffer manager (X'00C2'*xxxx*), carefully recheck jobs DSNTIJIN and DSNTIJID.

Installation step 16: Define and bind DB2 objects and user-maintained databases: DSNTIJSG

The default storage group is for user-defined DB2 tables that are not specifically assigned to a storage group. To create the default storage group and to bind DB2 plans for DCLGEN and SPUFI, run job DSNTIJSG. DSNTIJSG also defines the DB2-supplied stored procedures and binds packages for them.

| Before you run this job, you must specify the name of the WLM environment for
| each DB2-supplied stored procedure. You also can change the following items:
| • The volume for the default storage group.
| • The authorization level. You can make it more restrictive.
| • Miscellaneous authorizations. You can add additional authorization to other
| buffer pools.

If you use a product that uses a semicolon as a delimiter: The CLIST adds SQL statements to job DSNTIJSG. Products that use a semicolon as a delimiting character cause semicolons to be removed from the installation CLIST before it is executed. To correct the problem, replace the semicolons at the end of each SQL statement in job DSNTIJSG before you run the job.

Important: In a data sharing environment, you must ensure that the resource limit facility (RLF) is inactive on all members in the data sharing group before running DSNTIJSG. To do this, issue the STOP RLIMIT command to each member.

If you want to use the Data Facility Storage Management Subsystem to control the placement of data across volumes, edit job DSNTIJSG to replace the volume serial with `'*'`, as in the following statement:

```
CREATE STOGROUP SYSDEFLT VOLUMES ('*') ...
```

When you run the DSNTIJSG job, it:

- Binds DB2 plans, including the plans for SPUFI and DCLGEN. Also binds a package for a message routine that is used by SPUFI and DCLGEN, and binds that package into the plans for SPUFI and DCLGEN. If you use SPUFI to access remote sites or if some users need to run SPUFI under different terminal CCSIDs, you might need to bind different packages and plans for SPUFI. For more information, see "Special packages and plans for SPUFI" on page 300.
- Creates the default storage group, which is used for your database, table space, and table definitions that are not related to a specific storage group.
- Grants use of the default buffer pool and storage group to PUBLIC.
- Grants use of the SYSDEFLT table space. This table space does not really exist, but this GRANT statement is necessary to give users the ability to implicitly create table spaces in the default database.
- Grants authority to create tables and table spaces in the default database to PUBLIC.
- Grants execute privilege on the plans for the SPUFI, DCLGEN, and the DSNTIAD sample program to PUBLIC.
- Grants SELECT authority to the dummy table, SYSIBM.SYSDUMMY1 in the DSNDB06.SYSSTR table space.
- Defines the following DB2 tables to support JDBC and ODBC client functions:
  – SYSIBM.SYSDUMMYA
  – SYSIBM.SYSDUMMYE
  – SYSIBM.SYSDUMMYU
- Defines the following DB2-supplied stored procedures and binds packages for them:

DSNACICS	CICS transaction invocation stored procedure that invokes CICS server programs
DSNACCOR	DB2 real-time statistics stored procedure
DSNTJSPP	Stored procedure that DB2 Development Center calls to do Java stored procedure program preparation
DSNTBIND	Stored procedure that DSNTJSPP calls to bind Java stored procedures
DSNTPSMP	The SQL procedures processor stored procedure, which is used by DB2 Development Center. See IVP job DSNTEJ65 for an example of using DSNTPSMP.
WLM_REFRESH	Stored procedure for refreshing a specified WLM environment. See IVP job DSNTEJ6W for an example of using WLM_REFRESH.

DSNUTILS	Utility invocation stored procedure. See IVP jobs DSNTEJ6U and DSNTEJV for examples of using DSNUTILS.
DSNUTILU	Utility invocation stored procedure with utility control statement input in Unicode. See IVP job DSNTEJ6R for an example of using DSNUTILU.
DSNWZP	Subsystem parameter settings stored procedure. For an example of how to invoke DSNWZP, see Chapter 10, "Verifying installation with the sample applications," on page 363.
INSTALL_JAR	Stored procedure that installs a set of Java classes into the current SQL catalog and schema
REMOVE_JAR	Stored procedure that removes a Java JAR file and its classes from a specified catalog
REPLACE_JAR	Stored procedure that replaces a previously installed JAR file
# **DSNLEUSR**	Stored procedure that encrypts the authorization ID and password. DSNLEUSR runs only in new-function mode

Important: You must have TRACE and MONITOR1 privileges to run DSNWZP.

Important: You must set NUMTCB=1 in your WLM environment for DSNWZP.

The prolog of DSNTIJSG contains information about additional ways that you need to customize the CREATE PROCEDURE statements for DSNTJSPP and DSNTBIND before you run DSNTIJSG. DSNTJSPP and DSNTBIND require their own WLM environments. In addition, DSNTJSPP requires its own properties file. See *Application Programming Guide and Reference for Java* for more information about setting up DB2 UDB for z/OS Java support to work with DB2 Development Center.

See Appendix B of *DB2 Utility Guide and Reference* for information on stored procedures that perform utility functions.

- Defines the following DB2-supplied stored procedures to support JDBC and ODBC client functions:
  - SQLCOLPRIVILEGES
  - SQLCOLUMNS
  - SQLFOREIGNKEYS
  - SQLPRIMARYKEYS
  - SQLPROCEDURECOLS
  - SQLPROCEDURES
  - SQLSPECIALCOLUMNS
  - SQLSTATISTICS
  - SQLTABLEPRIVILEGES
  - SQLTABLES
  - SQLGETTYPEINFO
  - SQLUDTS
  - SQLCAMESSAGE

Stored procedures can be enabled after installation or migration. See "Enabling stored procedures after installation" on page 285 for the specific required steps.

Stored procedures run in a WLM-managed stored procedure address space. A WLM environment must be set up for these stored procedures. The name of this environment must match that of the WLM ENVIRONMENT parameter value in the CREATE PROCEDURE statement for each stored procedure. A step in DSNTIJSG grants EXECUTE authority to PUBLIC on these stored procedures and their packages. If this is undesired, the step should be edited to grant the authority only to specific users or groups. This job should be executed by a user with all the specific privileges needed.

The DSNTIJSG job also does the following actions for user maintained database activity:

- Creates the resource limit facility database. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information.
- Creates the data definition control support database. For more information, see Part 3 (Volume 1) of *DB2 Administration Guide*.

The DSNTIJSG job inserts a blank row into the communication database (CDB) table SYSIBM.LUNAME. The CDB holds tables that contain information about your connection with remote DB2 subsystems. A blank row allows all SNA clients
| to access DDF. TCP/IP remote clients cannot be controlled by using the CDB. If
| you run DSNTIJSG again, you will receive an SQLCODE -803 from the INSERT
| request because the blank row already exists.

For more information about starting DDF, see "Distributed data facility panel 2: DSNTIP5" on page 225. Field 3 of panel DSNTIP5, TCP/IP ALREADY VERIFIED, defines the minimum security requirements for all TCP/IP clients because inbound security requirements cannot be established on individual clients. For instructions on how to populate these tables, see "Step 4: Populate the communications database" on page 464 for VTAM information and "Step 4: Populate the communications database" on page 501 for TCP/IP information.

If the DSNTIJSG job runs successfully, it produces return codes of 0. It can also produce a return code of 4 because a step within this job attempts to delete a row from a table that might not exist at the time this job runs. Expect the following messages from the BIND statement for each object DB2 provides:

```
DSNE932I  WARNING, ONLY IBM-SUPPLIED PLAN NAMES SHOULD BEGIN WITH DSN

DSNE932I  WARNING, ONLY IBM-SUPPLIED PACKAGE-IDS SHOULD BEGIN WITH DSN

DSNE932I  WARNING, ONLY IBM-SUPPLIED COLLECTION-IDS SHOULD BEGIN WITH DSN
```

If the DSNTIJSG job fails or abends, be sure that the user that is specified on the JOB statement is authorized. Use the same name that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP.

Correct any other problems with the DSNTIJSG job and rerun it. If you do not have enough resources to run the job, review the values you specified for the DB2 installation parameters (see job DSNTIJUZ). Use the standard update procedure to make any necessary modifications. Refer to "The update process" on page 247 for information on the standard update procedure. Then stop DB2, rerun the DSNTIJUZ job, start DB2, and rerun the DSNTIJSG job.

Installation step 17: Populate the user-maintained databases (optional)

DSNTIJSG creates user-maintained databases that need to be populated. These include the resource limit specification table, and the data definition control support tables. Although DB2 automatically creates the user-maintained databases, you must complete this step if you plan to use data definition control or the resource limit facility (RLF). See the following sections for more information:

- Part 5 (Volume 2) of *DB2 Administration Guide* for information about the resource limit specification table
- Part 3 (Volume 1) of *DB2 Administration Guide* for information about the data definition control support tables.

Installation step 18: Bind the packages for DB2 REXX Language Support: DSNTIJRX (optional)

Before you can use DB2 REXX Language Support, you must bind the DB2 packages that DB2 REXX Language Support uses. Run job DSNTIJRX to do this. Before you run DSNTIJRX, make the following changes:

- Add a job statement.
- Change DSN SYSTEM(DSN) to DSN SYSTEM(*ssid*), where *ssid* is the name of the DB2 subsystem on which you plan to use DB2 REXX Language Support.
- Change all instances of DSN!!0 to your DB2 data set name prefix.
- Change all instances of DSNTIA!! to the plan name for the DSNTIAD program.

This job is not edited by the CLIST and is not in the SAMPLE LIBRARY from panel DSNTIPT (*prefix*.NEW.SDSNSAMP). DSNTIJRX is included in the target library if you have ordered DB2 REXX Language Support.

Installation step 19: Back up the DB2 directory and catalog: DSNTIJIC

Attention: You need to create a backup copy of the DB2 directory and catalog. DB2 starts if you do not have backup copies. However, if errors occur in the directory or catalog that require you to reinstall DB2 and you do not have backup copies, you lose all your tables and data.

Recommendation: Copy the catalog and directory at least daily if you make any changes in them. Recovery time for these databases is longer if the copies are not current, and the entire subsystem is affected.

For data sharing considerations, see *DB2 Data Sharing: Planning and Administration*.

To create a copy of the DB2 directory and catalog, run the DSNTIJIC job. Before you run the DSNTIJIC job, examine the job for the following information:

- The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also the unit name for the default archive log data sets.
- Expiration date or retention period. You can add a retention period or an expiration date to the job.
- The user on the JOB statement. Ensure that the user is authorized. This must be the same user that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP.

The DSNTIJIC job contains a list of all the DB2 directory and catalog table spaces. When you run job DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. Having copies of table spaces enables you to recover the DB2 catalog and DB2 directory in case of a failure.

If the DSNTIJIC job fails or abends, verify that no problems exist with the tape setup for image copy. If you find no problems with the tape setup, examine the utility job output (JOBLOG) or the console log for problems. For example, look for I/O errors or incorrect sizes.

Run the DSNTIJIC job periodically, perhaps daily or weekly, to reduce the amount of time required for recovering the directory or catalog. The copied data and log data sets are needed for recovery.

After you complete all the preceding steps, DB2 is available to TSO.

During the ISPF tailoring session, you named one or two IDs to have installation SYSADM authority. One of these users can now grant various levels of authority to other users. You can use SPUFI or a job similar to DSNTIJSG (described on page 274) to perform the authorization. For suggestions about using DB2 authorities and privileges, see Part 3 (Volume 1) of *DB2 Administration Guide*.

Installation step 20: Verify the installation

Run the sample applications to verify that your installation process has been successful. Select the phases that you need to run based on the attachment facilities you installed, the languages you use, and whether the sample objects exist. For more information, see Chapter 10, “Verifying installation with the sample applications,” on page 363.

Installing support for a communications network

Recommendation: If you plan to use the distributed data facility (DDF), become familiar with the DDF function and install Virtual Telecommunications Access Method (VTAM) (which DDF requires) and optionally z/OS Unix System Services TCP/IP support before loading the data sets into DB2 libraries. For information about installing VTAM, see Chapter 13, “Connecting systems with VTAM,” on page 453. For information about installing z/OS UNIX System Services TCP/IP support, see Chapter 14, “Connecting systems with TCP/IP,” on page 493.

The first step in accessing distributed data is to install VTAM if it is not installed already. The minimum VTAM release that DDF supports is Version 3 Release 3.

For information about planning your network configuration, see *Planning for NetView, NCP, and VTAM*. For information about installing your VTAM system, see *VTAM for MVS/ESA Network Implementation Guide*. For factors that you should consider before installing or customizing VTAM, see Part 4 (Volume 1) of of *DB2 Administration Guide*. For information about customizing VTAM to work optimally with DB2, see “Step 1: Customize VTAM for DB2” on page 455.

Installing NetView®: If you use DDF, you might want to install NetView so that DB2 can send alerts to NetView if DB2 detects security exposures or protocol errors. For information about installing NetView, see *Tivoli NetView for z/OS Installation: Getting Started*.

Installing a second DB2 subsystem on the same operating system

This topic explains how to install a second DB2 subsystem on an operating system. This topic explains:
- Planning considerations
- Installation procedure

Planning considerations

The primary consideration in planning for a second DB2 subsystem is its purpose. Using a second subsystem has a substantial impact on your environment. Second DB2 subsystems are not uncommon; organizations use a second DB2 subsystem to:
- Run separate service levels of the code. This can provide more extensive testing of preventive service before use with a production system.
- Run separate releases of the code. However, two different releases of DB2 on the same system must have separate libraries.
- Separate test and production activities. This setup can improve DB2's performance and availability for production.

  For example, suppose the processor that runs your production subsystem fails. If your test subsystem is on another processor, you can stop the test subsystem and start the production subsystem on that processor. This only works if the requisite DB2 data sets are on shared storage devices and you have used global resource serialization (GRS) (or an equivalent) to protect the production DB2 data sets.
- Prevent access by one class of users to certain data. If this is your primary purpose, reconsider the DB2 authorization scheme.

Table 50 lists other considerations in planning for a second DB2 subsystem.

Table 50. Considerations for installing a second DB2 subsystem

Considerations	Decisions that you need to make	Where to find information
Libraries	A single, shared library or separate libraries	Page 281
RACF protection	Resource and ID for the two subsystems	Note [1]
DB2 logging	BSDS, active, and archive log space requirements	Page 19
Database space requirements	DB2 directory, DB2 catalog, and user data	Note [3]
Performance	Processor and main storage use	Note [2]
Distributed data facility requirements	Coordination of location names, logical unit names, and network passwords with remote DB2 subsystems	Page 449
Common service area (CSA) requirement	DB2 and the IRLM	Page 29
Shared storage devices	Giving different names to active and archive log data sets, or including those data sets in the GRS inclusion list	Page 281

Note:
[1] See Part 3 (Volume 1) of *DB2 Administration Guide*.
[2] See Part 5 (Volume 2) of *DB2 Administration Guide*.
[3] See *DB2 Program Directory*.

Each subsystem must have a separate *prefix*.SDSNEXIT library. Sharing the *prefix*.SDSNSAMP library requires coordination to avoid overlaying parameter members.

In an environment in which DB2 uses shared disk storage, use different data set names for the active and archive log data sets if possible. If using different names is not possible, include those data sets in the GRS inclusion list. See "Enabling multiple DB2 subsystems to share disk storage" on page 284 for more details.

Installation procedure

This topic describes the principal steps to take when installing a second DB2 subsystem.

Loading DB2 libraries

This step is required if you want separate libraries for the two systems. If the systems have different code releases or different service levels, they must have separate libraries. You must plan the space for each library separately and load the libraries, using different prefixes for each library and for the SMP/E data sets or separate SMP/E zones.

Tailoring installation jobs

You can use the procedure described in Chapter 6, "Installing, migrating, and updating system parameters," on page 79 to specify appropriate parameter values for the second subsystem. Table 51 shows the parameter values that you can change and the parameter values you must change during the installation process. The figure includes the panel on which the parameter appears, the panel parameter name, and any comments that pertain to the parameter.

Table 51. Parameters to change when installing the second subsystem

Installation panel	Action
DSNTIPA0	Check all values and make appropriate changes.
DSNTIPA1	For separate libraries, change LIBRARY DATA SET NAME PREFIX and DATA SET NAME PREFIX.
DSNTIPA2	You must change CATALOG ALIAS. You can change VOLUME SERIALS 1 through 6.
DSNTIPK	Check all values and make appropriate changes.
DSNTIPH	Check all values and make appropriate changes.
DSNTIPT	Check all values and make appropriate changes.
DSNTIPU	Check all values and make appropriate changes.
DSNTIPQ	Check all values and make appropriate changes.
DSNTIPG	Check all values and make appropriate changes.
DSNTIPW	Check all values and make appropriate changes.
DSNTIPD	Check all values and make appropriate changes.
DSNTIP7	Check all values and make appropriate changes.
DSNTIPE	Check all values and make appropriate changes.
DSNTIP1	Check all values and make appropriate changes.
DSNTIP2	Check all values and make appropriate changes.
DSNTIPN	Check all values and make appropriate changes.
DSNTIPO	Check all values and make appropriate changes.

Table 51. Parameters to change when installing the second subsystem (continued)

Installation panel	Action
DSNTIPF	Check all values and make appropriate changes.
DSNTIP4	Check all values and make appropriate changes.
DSNTIP8	Check all values and make appropriate changes.
DSNTIPI	Change SUBSYSTEM NAME.
DSNTIPJ	Check all values and make appropriate changes.
DSNTIPP	Change ICF catalog to match the new ICF catalog. Change all passwords.
DSNTIPM	Change SUBSYSTEM NAME and SUBSYSTEM PREFIX.
DSNTIPL	Check all values and make appropriate changes. You must change data set names and prefixes.
DSNTIPA	Check all values and make appropriate changes. You must change data set names and prefixes.
DSNTIPS	Check all values and make appropriate changes. You must change data set names and prefixes.
DSNTIPR	Check all values and make appropriate changes. You must change names and password.
DSNTIP5	Check all values and make appropriate changes.
DSNTIPX	Check all values and make appropriate changes.
DSNTIPZ	Check all values and make appropriate changes.
DSNTIPY	Check all values and make appropriate changes.
DSNTIPC	Check values of DSMAX, EDMPOOL, EDMSTMTC, EDMDBC, SRTPOOL, and MAXRBLK; no other changes can be made.

Installing a second DB2 subsystem

The following list shows the jobs that you must run, or those that you need to run, when installing the second DB2 subsystem.

Job	Comments
DSNTIJCA	Run this job if you are defining a new ICF catalog.
DSNTIJIN	Run this job. It allocates the following data sets, which contain the sample plans and programs: *prefix*.DBRMLIB.DATA *prefix*.SRCLIB.DATA *prefix*.RUNLIB.LOAD Two subsystems cannot share these data sets. If undetected sharing occurs, data could be lost. If you use the same library prefix for both subsystems, change the name of the data sets, unless you do not need them on the first subsystem. Subsequent jobs overwrite them, and then the plans that are bound in the first subsystem do not work with the new load modules.
DSNTIJID	Run this job to initialize the DB2 data sets.
DSNTIJMV	Add another subsystem name and subsystem recognition character to IEFSSN*xx*. LNKLST*xx* modifications are needed only for separate libraries.

	For separate libraries, add STEPLIB statements to the precompile and bind steps for program preparation. Add STEPLIB statements for the DB2 offline utilities. Choose a naming convention for any new procedures, and change those as needed. For a single library, you must add the exit module data set (*prefix*.SDSNEXIT) to the STEPLIB statement to contain your changed subsystem parameter. Put this data set first in the STEPLIB concatenation.
DSNTIJUZ	Run this job, without changes if you use separate libraries. For a single library, provide a separate *prefix*.SDSNEXIT data set for each subsystem. The default SSID displayed by certain panels and procedures is the same for every subsystem. Ensure that the correct subsystem is specified in these cases.

If you are using RACF, you can define new user profiles and IDs to provide a separate level of security for each DB2 subsystem. See Part 3 (Volume 1) of *DB2 Administration Guide* for information.

Connecting attachment facilities

This topic contains information about connecting the attachment facilities for the second DB2 subsystem.

Connecting the TSO attachment facility: The following list shows the procedures to follow when connecting the TSO attachment facility.

Procedure	**Comments**
Logon procedure	Add STEPLIB statements if you are using a separate library for the second subsystem or are managing two different SDSNEXIT data sets. The STEPLIB statement must be authorized using the authorized program facility (APF). A DB2 logon procedure can use only one set of DB2 libraries.
Concatenate CLISTs	Update the SYSPROC library if you decide to have a separate library for the second subsystem.
Concatenate panels	Update the ISPPLIB library if you decide to have a separate library for the second subsystem.
Concatenate messages	Update the ISPMLIB library if you decide to have a separate library for a second subsystem.
Customize panel	You must update the ISPF primary option panel (ISR@PRIM) if you decide to have a separate library for the second subsystem.
Authorize users	Grant authorization on both subsystems as necessary.
DB2I defaults panel	Specify the new subsystem name as required.

Connecting the IMS attachment facility: The following list shows the procedures to follow when connecting a second IMS attachment facility.

Procedure	**Comments**

STEPLIB	For separate libraries, change this DD statement to refer to the new libraries.
DFSESL	For separate libraries, change this DD statement to refer to the new libraries.
Subsystem member	Define a new subsystem name, language interface token (LIT), and command recognition character (CRC).
Language interface	Define a new LIT and reassemble.
Linkage editor JCL	Specify the library that contains the new language interface.

Preparing DB2 for use

The following list explains how to prepare the subsystem for use when two DB2 subsystems are installed.

Procedure	**Comments**
IPL z/OS	This step is required to make any SYS1.PARMLIB changes take place.
Start DB2	Use the new command prefix (formerly called the subsystem recognition character) that you named during the installation process.
DSNTIJIC	Run this job.
DSNTIJTM	If you use the same library prefix for both subsystems, change the name of the data sets listed below, unless you do not need them on the first subsystem. Subsequent jobs write information in them and prevent use of the new load module DSNTIAD on the second subsystem with the previously bound plan DSNTEP81 on the first subsystem. *prefix*.DBRMLIB.DATA *prefix*.RUNLIB.LOAD
DSNTIJSG	Run this job.
DSNTEJ*xx*	If you use the same library prefix for both subsystems, change the name of the data sets listed below, unless you do not need them on the first subsystem. Subsequent jobs overwrite information in them and prevent use of the new load modules of sample programs on the second subsystem with the previously bound sample plans on the first subsystem. *prefix*.DBRMLIB.DATA *prefix*.SRCLIB.DATA *prefix*.RUNLIB.LOAD

Verifying your installation process

Run the sample applications to verify a successful installation process. Select the phases that you need to run based on the attachment facilities you installed, the languages you use, and whether the sample objects exist. For more information, see Chapter 10, “Verifying installation with the sample applications,” on page 363.

Enabling multiple DB2 subsystems to share disk storage

This topic applies to non-data sharing environments only. If you use a data sharing environment, the logs must be on shared disk storage.

If you plan to share disk storage among DB2 subsystems, avoid problems with your active and archive log data sets by taking the following precautions:

- Ensure that each subsystem has unique log data set names to avoid situations like the one described below:

  Subsystem A on operating system 1 and subsystem B on operating system 2 share the same z/OS catalog name, and their log data set names are the same. You start subsystem B while subsystem A is still running on operating system 1. This causes log data sets to be allocated for subsystem A, even though they already exist.
- Use GRS, or an equivalent, and include your active and archive log data sets in the GRS inclusion list. This prevents situations like the following one:

  Subsystem A on operating system 1 and subsystem B on operating system 2 share disk storage, and the active log is in a shared disk volume. Subsystem B fails. You attempt to start subsystem B, but you accidentally start subsystem A on operating system 2, even though it is still running on operating system 1. This causes log data sets to be allocated for subsystem A, even though they already exist.

Enabling stored procedures after installation

To enable stored procedures **after** you have completed the installation process, you can either run the installation CLIST in INSTALL or MIGRATE mode, or edit the job DSNTIJUZ.

DB2 stored procedures address space has been deprecated. Only stored procedures that were defined Version 7 can be run in the DB2-established stored address procedure space. All new stored procedures must be run in a WLM-managed stored procedure address space.

| **Recommendation:** Use Table 52 to determine the value that you should use for
| NUMTCB in your WLM environment for DB2-supplied stored procedures. If your
| system resources are constrained, you may need to choose a lower value.

Table 52. Recommended NUMTCB values for DB2-supplied stored procedures

#	DB2-supplied stored procedure	Recommended value of NUMTCB
#	DSN8EXP	1
#	DSNACCOR	15
#	DSNACICS	40
#	DSNTBIND	1
#	DSNTJSPP	60
#	DSNTPSMP	1
#	DSNUTILS	1
#	DSNUTILU	1
#	DSNWZP	1
#	SQLJ.INSTALL_JAR	15
#	SQLJ.REMOVE_JAR	15
#	SQLJ.REPLACE_JAR	15
#	SQLJ.DB2_INSTALL_JAR	15
#	SQLJ.DB2_REPLACE_JAR	15
#	SQLJ.DB2_REMOVE_JAR	15

Table 52. Recommended NUMTCB values for DB2-supplied stored procedures (continued)

DB2-supplied stored procedure	Recommended value of NUMTCB
SQLJ.DB2_UPDATEJARINFO	15
WLM_REFRESH	60

Run the installation CLIST in INSTALL or MIGRATE mode: On the installation panels, leave the existing values. Change only the following fields on their respective panels:

1. On panel DSNTIPA1, specify the input member that contains field values for your current installation.
2. On panel DSNTIPT, choose a different name for TEMP CLIST LIBRARY and SAMPLE LIBRARY to avoid overwriting your original libraries.
3. On panel DSNTIPX, configure your environment for running stored procedures.
4. On panel DSNTIPY, specify a remote location name. (This name is used for the stored procedure sample applications.)

Edit DSNTIJUZ:

1. Edit job DSNTIJUZ to add or change values of the following stored procedure parameters: MAX_NUM_CUR, MAX_ST_PROC, STORMXAB, and STORTIME.
2. Run DSNTIJUZ.
3. Restart DB2 with the new parameters.
4. Define JCL procedures for the WLM-managed stored procedures address spaces.

   Member DSNTIJMV of data set DSN810.SDSNSAMP contains sample JCL procedures for starting WLM-established address spaces.
5. Define WLM application environments for groups of stored procedures, and associate a JCL startup procedure with each application environment.

   See Part 5 (Volume 2) of *DB2 Administration Guide* for information about how to do this.
6. If the name of the WLM environment contains an underscore, the name must be contained within apostrophes (single quotes). For example:

```
//V71AWLM1 PROC   DB2SSN-V71A, NUMTCB=1, APPLENV='WLM_ENV1'
//TCBNUM1  EXEC   PGM=DSNX9WLM, TIME=1440
//             PARM='&DB2SSN,&NUMTCB,&APPLENV'
//             REGION=OM
```

Editing DSNTIJUZ does not give you a migration path because your DSNTID*xx* member and DSNTIJUZ parameters are not saved for future input. In addition, this method does not generate the sample jobs DSNTEJ6S, DSNTEJ6P, DSNTEJ6D, DSNTEJ6T, DSNTEJ6U, DSNTEJ6V, DSNTEJ6W, DSNTEJ6Z, DSNTEJ61, DSNTEJ62, DSNTEJ63, DSNTEJ64, and DSNTEJ65 because you are not running the DSNTINST CLIST.

For more information about stored procedures, see Part 6 of *DB2 Application Programming and SQL Guide*.

Enabling DB2-supplied stored procedures

You can enable the DB2-supplied stored procedures after you complete the installation process. The DB2-supplied stored procedures are as follows:

DSN8EXP	Stored procedure to explain SQL statements without needing the authority to run the SQL statements. For more information about DSN8EXP, see *DB2 Application Programming and SQL Guide*.

DSNAIMS	IMS transaction invocation stored procedure that invokes IMS transactions and commands. For more information about running DSNAIMS, see *DB2 Application Programming and SQL Guide*.
DSNACICS	CICS transaction invocation stored procedure that invokes CICS server programs
DSNACCOR	DB2 real-time statistics stored procedure
DSNLEUSR	Stored procedure that encrypts the authorization ID and password. DSNLEUSR runs only in new-function mode
DSNTBIND	Stored procedure that DSNTJSPP calls to bind Java stored procedures
DSNTJSPP	Stored procedure that DB2 Development Center calls to do Java stored procedure program preparation
DSNTPSMP	SQL procedures processor stored procedure
DSNUTILS	Utility invocation stored procedure
DSNUTILU	Utility invocation stored procedure with utility control statement input in Unicode
DSNWZP	Subsystem parameter stored procedure, which is used by DB2 Visual Explain and the DB2-supplied stored procedure WLM_REFRESH
SQLJ.INSTALL_JAR	Stored procedure that installs a set of Java classes into the current SQL catalog and schema
SQLJ.REMOVE_JAR	Stored procedure that removes a Java JAR file and its classes from a specified, local catalog
SQLJ.REPLACE_JAR	Stored procedure that replaces a previously installed JAR file in a local catalog
SQLJ.DB2_INSTALL_JAR	Stored procedure that installs a set of Java classes into a local or remote catalog
SQLJ.DB2_REMOVE_JAR	Stored procedure that removes a Java JAR file and its classes from a local or remote catalog
SQLJ.DB2_REPLACE_JAR	Stored procedure that replaces a previously installed JAR file in a local or remote catalog
SQLJ.DB2_UPDATEJARINFO	Stored procedure that inserts class, class source, and associated options for a previously installed JAR file in a local or remote catalog
WLM_REFRESH	Stored procedure for refreshing a specified WLM environment. See IVP job DSNTEJ6W for an example of using WLM_REFRESH.

For more information about stored procedures, see Part 6 of *DB2 Application Programming and SQL Guide*. See Appendix B of *DB2 Utility Guide and Reference* for information about stored procedures that perform utility functions.

Enabling DB2 Development Center procedures for Java

Java stored procedure support for the DB2 Development Center tool requires the following DB2-supplied stored procedures:

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

These stored procedures are created by job DSNTIJSG when you install or migrate to DB2 Version 8. DSNTIJSG also binds the packages for these stored procedures.

See *Application Programming Guide and Reference for Java* for more information about setting up DB2 Java support to work with DB2 Development Center.

For more information about stored procedures, see Part 6 of *DB2 Application Programming and SQL Guide*.

Enabling DB2 Control Center procedures

If you already installed the DB2 Universal Database Control Center (FMID JDB881D), use job DSNTIJCC to define the DB2-supplied stored procedures for DB2 Universal Database Control Center. Installation job DSNTIJCC creates a database that the Control Center uses to store information about templates, object lists, and utility procedures. The database also contains information that the DB2 Universal Database Control Center uses to restart stopped utilities. By default, installation job DSNTIJCC names this database CC390. Because DSNTIJCC is not tailored by the installation process, you need to make the following changes before you run DSNTIJCC:

- Add a JOB statement.
- Change all instances of DSN!!0.SDSNLOAD to the name of your DB2 load library.
- Change all instances of DSN!!0.RUNLIB.LOAD to the name of your application load library.
- Change all instances of DSN!!0.SDSNDBRM to the name of your DB2 DBRM library.
- Change all instances of DSNTIA!! to the plan name for program DSNTIAD.
- In all instances of SYSTEM(DSN), change DSN to your DB2 subsystem name.

The Control Center stored procedures are as follows:

DSNACCAV
DB2 Universal Database Control Center partition information stored procedure

DSNACCDD
DB2 Universal Database Control Center stored procedure to delete data sets

DSNACCDE
DB2 Universal Database Control Center stored procedure to indicate if a data set exists

DSNACCDL
DB2 Universal Database Control Center stored procedure to list data sets

DSNACCDR
DB2 Universal Database Control Center stored procedure to rename data sets

DSNACCDS
DB2 Universal Database Control Center stored procedure to create, append to, or replace LCRECL=80, RECFM=FB PDSE data set members or PS data sets

DSNACCQC
DB2 Universal Database Control Center catalog query stored procedure

For more information about stored procedures, see Part 6 of *DB2 Application Programming and SQL Guide*.

Enabling SQL procedures support

You can enable SQL procedures support after you complete the installation process. The objects that are required for the DB2 SQL procedures support are:

- DSNHSQL, the JCL procedure for preparing SQL procedures in batch mode, provided by job DSNTIJMV.
- DSNTPSMP, a stored procedure that can be used to prepare SQL procedures for execution, created by job DSNTIJSG.
- The DSNTPSMP REXX EXEC, which is in the *prefix*.SDSNCLST data set.

If you use the SQL procedures processor, you need a WLM startup procedure and the DB2 REXX Language Support feature. See *prefix*.SDSNSAMP(DSN8WLMP) for an example of such a procedure.

Enabling stored procedures and tables for JDBC and ODBC support

As part of the DB2 installation process, installation job DSNTIJSG registers and binds stored procedures for JDBC and ODBC support. These stored procedures are:

- SYSIBM.SQLCOLPRIVILEGES
- SYSIBM.SQLCOLUMNS
- SYSIBM.SQLFOREIGNKEYS
- SYSIBM.SQLPRIMARYKEYS
- SYSIBM.SQLPROCEDURECOLS
- SYSIBM.SQLPROCEDURES
- SYSIBM.SQLSPECIALCOLUMNS
- SYSIBM.SQLSTATISTICS
- SYSIBM.SQLTABLEPRIVILEGES
- SYSIBM.SQLTABLES
- SYSIBM.SQLGETTYPEINFO
- SYSIBM.SQLUDTS
- SYSIBM.SQLCAMESSAGE

DSNTIJSG also creates the following tables:
- SYSIBM.SYSDUMMYA
- SYSIBM.SYSDUMMYE
- SYSIBM.SYSDUMMYU

These tables are used by JDBC and ODBC to avoid unnecessary character conversion during execution of functions such as `LEN()` and `SUBSTR()` when the arguments of these functions are CLOB or DBCLOB locators.

If you did not create these procedures during installation, you need to customize and run job DSNTIJMS to define stored procedures that provide support for JDBC and ODBC client functions. These stored procedures run in a WLM-managed stored procedure address space. You must set up a WLM environment for these stored procedures. The name of this environment must match the WLM ENVIRONMENT parameter value in the CREATE PROCEDURE statement for each stored procedure. If you create stored procedures after migration to Version 8 new-function mode, run DSNTIJMS again.

DSNTIJSG and DSNTIJMS contain a step which grants EXECUTE authority to PUBLIC on these stored procedures and their packages. If you do not desire this, edit the jobs to grant the authority only to specific users or groups. These jobs should be executed by a user with all the specific privileges needed.

For more information about stored procedures, see Part 6 of *DB2 Application Programming and SQL Guide*. For more information about enabling stored procedures for JDBC support, see *DB2 Application Programming Guide and Reference for Java*.

Enabling the IMS transaction invocation procedure

You must have IMS Version 7 or later with OTMA Callable Interface enabled before you run DSNAIMS.

DSNAIMS needs to run in a WLM-established stored procedure address space. Although DSNAIMS can run in an address space with other stored procedures, DSNAIMS performs better if it runs in its own address space. Therefore, setting up a WLM application environment and startup procedure specifically for running DSNAIMS is recommended.

After you set up the WLM application environment, create a JCL startup procedure for the stored procedure address space.

Installation job DSNTIJMV contains sample startup procedure DSNWLM, which you can use as a model for your startup procedures.

Change the following items in each startup procedure:
- Change the procedure name from DSNWLM to the procedure name that you specified when you set up the WLM environment.
- Change the value of APPLENV to the name of the WLM environment that you set up for DSNAIMS stored procedure. This name must match the name in the WLM ENVIRONMENT parameter in the CREATE PROCEDURE statement in DSNTIJIM.
- Change the value of DB2SSN to your DB2 subsystem name.

Customize and run job DSNTIJIM to define DSNAIMS to DB2. See the prolog of DSNTIJIM for instructions.

For information about how to set up a WLM environment and associate an address space startup procedure with that environment, see *z/OS MVS Planning: Workload Management* and *DB2 Administration Guide*. For information about running the IMS transaction invocation stored procedure, see *DB2 Application Programming and SQL Guide*.

Enabling the CICS transaction invocation procedure

DSNACICS needs to run in a WLM-established stored procedure address space. Although DSNACICS can run in an address space with other stored procedures, DSNACICS performs better if it runs in its own address space. Therefore, setting up a WLM application environment and startup procedure specifically for running DSNACICS is recommended.

After you set up the WLM application environment, create a JCL startup procedure for the stored procedure address space. Installation job DSNTIJMV contains a sample startup procedure for a stored procedure address space for running DSNACICS called DSNCICS. When you run the installation or migration CLIST, DB2 customizes DSNTIJMV with data set names that you specify. Running DSNTJMV installs the startup procedure in your SYS1.PROCLIB data set. Use panel DSNTIPW to specify CICS data set names. You can specify the library that contains the load modules for the CICS EXCI interface in the CICS EXCI LIBRARY field of panel DSNTIPW. DSNACICS uses this library to connect to CICS and call CICS programs.

For information about how to set up a WLM environment and associate an address space startup procedure with that environment, see *z/OS Planning: Workload Management* and *DB2 Administration Guide*.

Enabling the EXPLAIN stored procedure

1. Customize and run job DSNTEJXP to define DSN8EXP to DB2, to optionally precompile, compile, and link-edit the source code version of DSNTEJXP, and to bind the DSNTEJXP package.

   Two versions of DSN8EXP are available: a source code version and a load module version. If you plan on customizing the DSN8EXP program, use the source code version. If you do not plan on customizing the DSN8EXP program, delete the step to precompile, compile, and link the program, and use the load module version.
2. Set up the WLM environment.

   DSN8EXP needs to run in a WLM-established stored procedure address space. Although DSN8EXP can run in an address space with other stored procedures, DSN8EXP performs better if it runs in its own address space. Therefore, it is recommended that you set up a WLM application environment and startup procedure specifically for running DSN8EXP.

   The following is an example of an address space startup procedure for the WLM environment used to run DSN8EXP:

   ```
   //WLMENVXP PROC DB2SSN=DSN,NUMTCB=1,APPLENV=WLMENVXP
   //*
   //DSN8EXP EXEC PGM=DSNX9WLM,TIME=1440,
   //             PARM='&DB2SSN,&NUMTCB,&APPLENV',
   //             REGION=0M
   //STEPLIB  DD  DISP=SHR,DSN=DB2.SDSNLOAD
   //         DD  DISP=SHR,DSN=CEE.SCEERUN
   ```

```
//SYSTSPRT DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSABEND DD  DUMMY
//DSNTRACE DD  SYSOUT=*
```

This example assumes that you use the DB2-supplied load module for DSN8EXP, which is in the *prefix*.SDSNLOAD library. If you create your own DSN8EXP module from the source code in *prefix*.SDSNSAMP, you need to add the library that contains your load module to the STEPLIB concatenation, ahead of *prefix*.SDSNLOAD.

3. Define a PLAN_TABLE for each user. See member DSNTESC in *prefix*.SDSNSAMP for sample SQL statements.
4. Grant authorization for users to execute the package for DSN8EXP. For example:

   ```
   GRANT EXECUTE ON PROCEDURE DSN8.DSN8EXP TO user;
   ```

   Important: The privileges to run DSN8EXP should be granted with consideration that DSN8EXP can EXPLAIN on any explainable SQL statement that is valid on the system, and that EXPLAIN output can reveal potentially sensitive information. For example, you should not grant access to PUBLIC to use DSN8EXP.

For information on how to set up a WLM environment and associate an address space startup procedure with that environment, see *DB2 Administration Guide*. For more information about setting up and running the EXPLAIN stored procedure, see *DB2 Application Programming and SQL Guide*.

Enabling MQSeries user-defined functions

This topic describes the tasks that you need to perform on your z/OS system before applications can call the MQSeries user-defined functions. If you want to use MQSeries publish/subscribe user-defined functions, you must install WebSphere MQ Integrator Broker for z/OS Version 2 Release 1 and WebSphere MQ Integrator Broker for Windows 2000/NT Version 2 Release 1.

For more information on installing or configuring these products, see the following books:

Product	Reference
RRS	*z/OS MVS Programming: Resource Recovery*
WLM	*z/OS MVS Planning: Workload Management*
MQSeries for OS/390 and AMI	*MQSeries for OS/390 System Setup Guide*
	MQSeries Application Messaging Interface
WebSphere MQ Integrator for z/OS	*Customization and Administration Guide* (SC34-6175)
WebSphere MQ Integrator	*Windows NT and Windows 2000 Installation Guide* (GC34-5600)
	Introduction and Planning (GC34-5599)
	Administration Guide (SC34-6171)
	Using the Control Center (SC34-6168)

After you install the prerequisite products, you need to perform the following steps before you can use MQSeries user-defined functions:

1. Edit the MQSeries Application Messaging Interface configuration files. See "Editing the MQSeries configuration files" on page 293.

2. Implement cache files for the AMI repository file and the AMI local host file. This is an optional but recommended step. See "Using caches for AMI files (recommended)" on page 294.
3. To use publish/subscribe user-defined functions, create and configure a broker domain. See "Creating and configuring a broker domain" on page 294.
4. Start the queue manager. See "Starting the queue manager" on page 295.
5. To use publish/subscribe user-defined functions, start the broker. See "Starting the broker" on page 295.
6. Customize WLM for running MQSeries user-defined functions. See "Customizing WLM for running MQSeries user-defined function support" on page 295.
7. Define MQSeries user-defined functions to DB2. See "Defining the MQSeries user-defined functions to DB2" on page 296.
8. Start the address space in which the MQSeries user-defined functions run. See "Starting the WLM address spaces for MQSeries user-defined functions" on page 297.
9. Verify the installation. See "Verifying the DB2 and MQSeries setup" on page 297.

Editing the MQSeries configuration files

This topic describes changes that you need to make to two configuration files to make the MQSeries user-defined functions work with your MQSeries installation. The changes that are described let you use the MQSeries user-defined functions with the default queue (DB2MQ_DEFAULT_Q), default service (DB2.DEFAULT.SERVICE), and default policy (DB2.DEFAULT.POLICY). If you want to add new services, policies, or queues, you need to make additional changes to these files. The files that you need to change are:

DSNAMT	The AMI repository file that describes services, policies, and policy handlers for the MQSeries user-defined functions.
DSNAMTHT	The AMI local host file, which describes the mapping from a connection name to the name of the MQSeries queue manager that you want to connect to.

Recommendation: To edit these files, use the MQSeries AMI Administration Tool. You can download the MQSeries AMI Administration Tool at www.ibm.com/software/integration/support/supportpacs/individual/ma0f.html You can also edit these files using a plain text editor.

The unedited copies of these files are in *prefix*.SDSNSAMP library. They are XML files that are in the UTF-8 encoding scheme.

To edit these files, use the MQSeries AMI Administration Tool. If you do not use the MQSeries AMI Administration Tool, follow these steps to edit the files using a plain text editor:

1. Download the files via FTP in binary mode from the *prefix*.SDSNSAMP data set to a workstation.
2. Edit the files. Change the following items:
   - In the DSNAMT file:
     – Change all instances of `MQND` to the name of your queue manager.

    - If you plan to use the two-phase commit functions, in the `GeneralAttributes_syncpoint` entry, change `value='No'` to `value='Yes'`.
  - In the DSNAMTHT file, change the defaultConnection value from `MQND` to the name of your queue manager.
3. Upload the files via FTP in binary mode to a different data set, such as *prefix*.SRCLIB.DATA, which has a logical record length of 80 and a fixed-block format.

Using caches for AMI files (recommended)

To improve the performance of your MQSeries user-defined functions, you can use cache files instead of the AMI repository file and AMI host file. To implement the caches, follow these steps:

1. Create JCL to run the cache generator.

   The cache generator, AMTASM10, is shipped with MQSeries for OS/390 in the SCSQLOAD data set. This program generates source code for the AMI cache files. The following JCL example shows how to execute the cache generator:

   ```
   //GO EXEC PGM=AMTASM10
   //STEPLIB DD DSN=scsqload-target-library,DISP=SHR
   // DD DSN=scsqanle-target-library,DISP=SHR
   //AMTHOST DD DSN=prefix.SRCLIB.DATA(DSNAMTHT),DISP=SHR
   //AMT DD DSN=prefix.SRCLIB.DATA(DSNAMT),DISP=SHR
   //SYSPRINT DD SYSOUT=*
   //ASMHOST DD DSN=source-code-library(AMTHOST),DISP=SHR
   //ASMREPOS DD DSN=source-code-library(AMT),DISP=SHR
   ```

   To generate this JCL:

   - Replace *scsqload-target-library* with the name of your SCSQLOAD target library.
   - Replace *scsqanle-target-library* with the name of your SCSQANLE target library.
   - Replace *prefix*.SRCLIB.DATA with the name of the library that contains your customized DSNAMTHT and DSNAMT files.
   - Replace *source-code-library* with the name of any partitioned data set that has a logical record length of 80 and a fixed-block format.

   If AMTASM10 runs successfully, you should receive the following messages in the SYSPRINT data set:

   ```
   AMT0004I AMI repository cache created
   AMT0005I AMI host file cache created
   ```

   AMTASM10 puts the source code for the AMI cache files in the AMTHOST member of the ASMHOST data set and the AMT member of the ASMREPOS data set.
2. Assemble and link-edit the source code for the AMI cache files. You need to link-edit the object modules for these files into a data set that is in the STEPLIB concatenation for the WLM startup procedure for the stored procedure address space in which the MQSeries user-defined functions run. See "Customizing WLM for running MQSeries user-defined function support" on page 295 for more information about this address space startup procedure.

Creating and configuring a broker domain

To use publish/subscribe user-defined functions, you must create and configure a broker domain using the WebSphere MQ Integrator Control Center. To create the broker domain, complete the following steps:

1. Create a new message flow by connecting the MQInput node SYSTEM.BROKER.DEFAULT.STREAM to the Publication node.
2. Assign the new message flow to an Execution group.
3. Deploy the message flow.

For more information about creating and configuring a broker domain, see *WebSphere MQ Integrator for z/OS: Customization and Administration Guide*.

Starting the queue manager

To start the queue manager, issue the following command from the z/OS console:

```
<command-prefix-string> START QMGR
```

<command-prefix-string> is the command prefix string for the MQSeries subsystem.

For example, if the command prefix string for your MQSeries subsystem is –MQND, issue this command:

```
–MQND START QMGR
```

To check whether the queue manager is available, issue the following command from the TSO Command Processor panel, which is option 6 of the ISPF/PDF primary options menu:

```
CSQOREXX
```

Starting the broker

To use publish/subscribe user-defined functions, you must start the broker by issuing the following command:

```
/s <broker-name>
```

<broker-name> is the name of the broker that you that you created to use the publish/subscribe user-defined functions.

For more information about starting the broker, see *WebSphere MQ Integrator for z/OS: Customization and Administration Guide*.

Customizing WLM for running MQSeries user-defined function support

You need to set up two different WLM environments and corresponding WLM startup procedures for MQSeries functions:
- An environment and startup procedure for running the single-phase commit functions
- An environment and startup procedure for running the two-phase commit functions

After you set up the WLM application environment, you need to create JCL startup procedures for the stored procedure address spaces.

Installation job DSNTIJMV contains sample startup procedure DSNWLM, which you can use as a model for your startup procedures.

Change the following items in each startup procedure:
- Change the procedure name from DSNWLM to the procedure name that you specified when you set up the WLM environment.

- Change the value of APPLENV to the name of the WLM environment that you set up for each set of the MQSeries user-defined functions. This name must match the name in the WLM ENVIRONMENT parameter in the CREATE FUNCTION statement, which is discussed in the topic "Defining the MQSeries user-defined functions to DB2."
- Change the value of DB2SSN to your DB2 subsystem name.
- Add the following DD statements after the STEPLIB DD statement:

  ```
  //AMT DD DSN=prefix.SRCLIB.DATA(DSNAMT),DISP=SHR
  //AMTHOST DD DSN=prefix.SRCLIB.DATA(DSNAMTHT),DISP=SHR
  ```

  Change the data set names to match the data sets that contain your edited DSNAMT and DSNAMTHT files.
- If you use the AMI repository file and the AMI host file, add the following DD statements to your STEPLIB concatenation:

  ```
  //  DD  DSN=MQSERIES.SCSQLOAD, DISP=SHR
  //  DD  DSN=MQSERIES.SCSQAUTH, DISP=SHR
  //  DD  DSN=MQSERIES.SCSQANLE, DISP=SHR
  ```

  Change the data set name to match the name of your MQSeries load library.
- If you use caches instead of the AMI repository file, add the following DD statements to your STEPLIB concatenation:

  ```
  //     DD  DSN=MQSERIES.ASM.LOAD,DISP=SHR
  ```

  Change the data set name to match the name of the data set that contains the caches.

Defining the MQSeries user-defined functions to DB2

You define MQSeries user-defined functions to DB2 by executing CREATE FUNCTION statements. Installation jobs DSNTIJM1 and DSNTIJM2 contain the steps to define MQSeries user-defined functions to DB2 and grant the authority to execute them. Job DSNTIJM1 defines the single-phase commit functions. Job DSNTIJM2 defines the two-phase commit functions.

Additional installation jobs contain the steps to define publish/subscribe user-defined functions to DB2 and grant the authority to execute them. Job DSNTIJS1 defines the single-phase commit user-defined functions. Job DSNTIJS2 defines the two-phase commit user-defined functions. Instructions for customizing the jobs are in the job prologs.

Before you execute the CREATE FUNCTION statements for MQSeries user-defined functions, you might need to make the following changes to the statement:

- Change the WLM ENVIRONMENT parameter value to match the name of the WLM application environment that you created for running the MQSeries functions. Ensure that you specify different WLM environments for functions in DSNTIJM1 and DSNTIJM2.
- If you want the functions to run under the authorization ID of the stored procedure address space, change the SECURITY parameter to SECURITY DB2. If you want the functions to run under the authorization ID of the process that invokes the functions, change the security parameter to SECURITY USER.
- Change the CCSID clauses for input and output parameters to the encoding schemes that you use for your data. Possible values are CCSID EBCDIC, CCSID ASCII, or CCSID UNICODE.

Starting the WLM address spaces for MQSeries user-defined functions

The easiest way to control WLM address spaces is to run WLM under automatic control. To use automatic control, you specify the name of the WLM address space startup procedure when you define the application environment. After you start the application environment, WLM controls the stored procedure address spaces.

WLM-environment-name is the name of a WLM environment that you set up for MQSeries user-defined functions.

If WLM address spaces are already started, you might need to restart them manually. For MQSeries user-defined functions, you need to restart the address spaces if you add AMI file caches. To restart WLM address spaces for an application environment, issue this z/OS command:

```
VARY WLM,APPLENV=WLM-environment-name,REFRESH
```

Verifying the DB2 and MQSeries setup

To verify that your MQSeries environment is set up correctly for invoking DB2 MQSeries user-defined functions, customize and run job DSNTEJMQ. Instructions for customizing this job are in the job prolog. Job DSNTEJMQ defines a local queue and invokes each of the MQSeries functions through DSNTEP2.

To verify that your MQSeries environment is set up correctly for invoking DB2 MQSeries publish/subscribe user-defined functions, customize and run jobs DSNTEJSQ and DSNTEJSV. Instructions for customizing these jobs are in the job prolog.

Enabling MQSeries XML user-defined functions and stored procedures

This topic describes the tasks that you need to perform on your OS/390 or z/OS system before applications can call the MQSeries XML user-defined functions and stored procedures.

To install MQSeries XML user-defined functions and stored procedures, complete the following steps:

- Run the verification jobs to verify that the core MQSeries functions, extended MQSeries functions, and DB2 XML Extender are properly installed.
- Set up the WLM environments for MQSeries XML user-defined functions and stored procedures.
- Customize and run installation job DSNTIJMX to define the MQSeries user-defined functions and stored procedures to DB2.
- Customize and submit jobs DSNTEJX1, DSNTEJX2, and DSNTEJX3 to verify that the MQSeries user-defined functions and stored procedures are correctly installed.

Installation job DSNTIJMX calls the DSNMXADM program to enable or disable stored procedures. DSNMXADM invokes the following stored procedures:

DMQXML2C.DXXENBMQXML
Enables all MQSeries XML user-defined functions and stored procedures

DMQXML2C.DXXDISMQXML
Disables all MQSeries XML user-defined functions and stored procedures

DSNMXADM can be executed through sample job DSNTIJMX or from UNIX System Services.

DSNMXADM has the following syntax for installation:

```
dsnmxadm enable_MQXML -i ssid -w wlmName -c ccsid -s security -p phase
-f ENBLPUB|EXCLPUB
```

where:

- *ssid* is the DB2 subsystem ID
- *wlmName* is the WLM environment name
- *ccsid* is the CCSID of the parameters of the MQSeries XML stored procedure (ASCII, EBCDIC, or UNICODE)
- *security* is the security that is used for running the routines (DB2 or USER)
- *phase* specifies whether single-phase commit or two-phase commit routines are to be installed (ONE or TWO)
- ENBLPUB specifies whether publish and subscribe functions are enabled
- EXCLPUB specifies to disable publish and subscribe functions

All parameters except `-f ENBLPUB|EXCLPUB` are required. If you do not include this parameter, ENBLPUB is assumed.

DSNMXADM has the following syntax for uninstallation:

```
dsnmxadm disable_MQXML -i ssid -p phase
```

If you want to run this program from UNIX System services, you must complete the following steps

- Create a dummy module for DSNMXADM and turn on the sticky bit for this module. Issue the following command in UNIX System Services:
  ```
  chmod 1755 dsnmxadm
  ```
- Include the *prefix*.SDSNLOAD library in the STEPLIB concatenation in your profile file.

Enabling DB2 Web services

Web services are sets of business functions that applications or other Web servers invoke over the Internet using standard HTTP requests.

Enabling DB2 as a Web service provider allows you to create Web services on z/OS with your DB2 data and applications. Enabling DB2 as a Web service consumer allows you to receive Web service data in your DB2 applications.

Enabling DB2 as a Web service provider

DB2 UDB for z/OS as a Web service provider has the following prerequisites:

- Enable JDBC (legacy or universal) in DB2
- Install WebSphere Application Server Version 5 or later

  Ensure that the WebSphere Application Server library contains the following two files:
  - mail.jar
  - activation.jar

  If mail.jar is not present, download JavaMail Version 1.2 or later. If activation.jar is not present, download JavaBeans Activation Framework Version 1.0.1 or later. You can download these products from the following Web site:

```
http://java.sun.com
```

- Install IBM DB2 XML Extender for OS/390 Version 7 or later (optional)

To use DB2 Web Services Object Runtime Framework (WORF), you need to make the runtime services available to WebSphere Application Server (WAS). By default, WORF is installed in the HFS directory:

```
/usr/lpp/db2810_worf/
```

The base install directory contains the `lib/` subdirectory that contains the runtime JAR file `worf.jar`. To begin using WORF, copy `worf.jar`, `mail.jar`, and `activation.jar` to a WAS shared library directory that you have already set up and restart WAS.

WORF provides a sample web application in the following directory:

```
lib/services.war
```

The application contains sample Document Access Definition Extension (DADX) files that define sample DB2 Web services. To set up this application with sample DADX files:

1. Follow the instructions in the job prolog to customize and run job DSNTEJWS, which is located in the *prefix*.SDSNSAMP directory. DSNTEJWS creates the DB2 tables that are used by the sample application.
2. By default, the sample application uses the legacy JDBC driver connectivity to connect to a DB2 server. If you configure WAS with JDBC providers that use the universal JDBC driver, you need to perform the following actions to configure the sample application to use the universal JDBC driver:
   a. Copy `services.war` to a temporary directory.
   b. Unjar `services.war` by issuing the following command:
      ```
      jar -xvf services.war
      ```
   c. Open the `group.properties` files, which are located in the following directories:
      ```
      WEB-INF/classes/groups/dxx_sample
      WEB-INF/classes/groups/dxx_travel
      ```
      Modify the dbDriver, dbURL userID, and password fields to have the following values:
      ```
      dbDriver=com.ibm.db2.jcc.DB2Driver
      dbURL=jdbc:db2://server:port/database
      userID=DB2 userid
      password=DB2 userid password
      ```
   d. Jar the files by issuing the following command:
      ```
      jar -cvf services.war *
      ```
3. Use a Web browser to connect to your WAS Administrative Console.
4. Under "Applications", select "Install New Application".
5. Select the "Server Path" option, and type the location of the `services.war` file in the text box. If you installed WORF in the default location, the server path is:
   ```
   /usr/lpp/db2810_worf/lib/services.war
   ```
6. Fill in a context root for the application (for example, services). Click "Next".
7. On the following screens, respond as necessary for your local setup. You can accept the default settings.
8. On the last screen, click "Finish". Click "Save to Master Configuration" to apply your changes.

#		To load the application:

1. On the WAS Administrative Console's main page, under "Applications", select "Enterprise Applications". Select the application and click "Start" to load the application.
2. After the application loads, point a browser to your server with the context root that you chose (for example, http://*server*:*port*/services/). The welcome page lists the sample DADX files that are provided in `services.war`. To test the services, click on the links.

Enabling DB2 as a Web service consumer

DB2 UDB for z/OS as a Web service consumer has the following prerequisites:

- Install z/OS V1R4 or later

  Attention: Be sure to apply Language Environment APARs PQ63045 and PQ84190
- Install IBM XML Toolkit for z/OS, C++ Edition 1.8
- Configure TCP/IP

Installation job DSNTIJMV contains sample startup procedure DSNWLM, which you can use as a model for your startup procedures.

Change the following items in each startup procedure:

- Change the procedure name from DSNWLM to the procedure name that you specified when you set up the WLM environment.
- Change the value of APPLENV to the name of the WLM environment that you set up for the Web services consumer user-defined functions. This name must match the name in the WLM ENVIRONMENT parameter in the CREATE PROCEDURE statement in DSNTIJIM.
- Change the value of DB2SSN to your DB2 subsystem name.
- Add the data set name of the XML Toolkit load library (XPLINKed version) to the STEPLIB concatenation. If you used the default data set names when you installed the XML Toolkit, the load library data set name is *userid*.SIXMLOD1.

After you set up the WLM application environment, create a JCL startup procedure for the stored procedure address space.

To create load modules that are used by the Web service consumer, customize job DSNTIJWL as indicated in its prolog and run the job.

To define the Web service consumer user-defined functions to DB2, customize job DSNTIJWS as indicated in its prolog and run the job.

Loading data with an SQL cursor

You can use an SQL cursor with the LOAD utility to load data from a remote location. Before loading data, you must bind the DSNUT810 package at each location from which you want to load data. A local package for DSNUT810 is bound by installation job DSNTIJSG when you install or migrate to a new version of DB2 UDB for z/OS.

Special packages and plans for SPUFI

This topic discusses situations where special packages and plans for SPUFI are required.

Running SPUFI at remote systems

You can use SPUFI to execute an interactive CONNECT statement and then execute SQL statements at a remote location. To do that on non-DB2 systems, you must bind a package for SPUFI on each of those systems. Use the following commands:

```
BIND PACKAGE (location_name.DSNESPCS) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(CS) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.DSNESPRR) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(RR) LIB('prefix.SDSNDBRM')
```

If the BIND PACKAGE command fails, the package already exists. See if the time and date formats returned by the existing packages are satisfactory. If they are, the existing packages can be used without any change to the package list in the SPUFI plans. If you need to change the time and date formats returned by the existing packages, you must bind new packages with different collection identifiers that have been agreed to by the database server.

For example, if the collection identifiers are PRIVATCS and PRIVATRR, the commands for doing a remote bind are as follows:

```
BIND PACKAGE (location_name.PRIVATCS) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(CS) LIB('prefix.SDSNDBRM')

BIND PACKAGE (location_name.PRIVATRR) MEMBER(DSNESM68)
            ACTION(ADD) ISOLATION(RR) LIB('prefix.SDSNDBRM')
```

The SPUFI plans at the DB2 system must be rebound because the location name parameter (which is usually optional) must be explicitly specified for the remote access functions to construct the correct package name. (SPUFI does not use the SQL statement SET CURRENT PACKAGESET.) The location name entry in the package list must precede any pattern-matching character entry. For example, the package list for the DSNESPCS plan is as follows:

```
location_name.PRIVATCS.DSNESM68
*.DSNESPCS.DSNESM68
```

The package list for the DSNESPRR plan is as follows:

```
location_name.PRIVATRR.DSNESM68
*.DSNESPRR.DSNESM68
```

Making SPUFI work with different terminal CCSIDs

In cases where the terminal CCSID cannot be changed to the SPUFI CCSID, consider creating additional SPUFI packages and plans that specify the terminal CCSID encoding. You can then use the SPUFI default panels to use the plans with the special CCSID encoding instead of the DB2-supplied plans (DSNESPCS, DSNESPRR, and DSNESPUR).

Example: Suppose that when you install DB2, you create the SPUFI plans and packages using ENCODING(EBCDIC), and the default subsystem CCSID for EBCDIC data, as specified on installation panel DSNTIPF in the EBCDIC CCSID field. Most of the people in your organization use this CCSID as their terminal CCSID, so they do not encounter errors. However, a large group of DB2 application programmers code in the C language and prefer CCSID 1047 as their terminal CCSID. You can prevent this group from corrupting DB2 data and receiving errors by completing the following tasks:

1. Bind SPUFI packages and plans specifically for users who require a terminal CCSID setting of 1047. For example, your commands might look like the following commands:

```
BIND PACKAGE(TIAP1047) MEMBER(DSNTIAP) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPCS1047) -
     PKLIST(SPCS1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
     ISOLATION(CS) ENCODING(1047) ACTION(REPLACE)
BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPRR1047) -
     PKLIST(SPRR1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
     ISOLATION(RR) ENCODING(1047) ACTION(REPLACE)
```

2. Grant access to the new packages and plans to the target user group.
3. Instruct users who use terminal CCSID 1047 to specify the following settings in the SPUFI default panels:
   - In panel DSNESP02, specify YES for CHANGE PLAN NAMES.
   - In panel DSNESP07, specify SPCS1047 in the CS ISOLATION PLAN field, SPRR1047 in the RR ISOLATION PLAN field, and SPUR1047 in the UR ISOLATION PLAN field.

The programmers who require CCSID 1047 can now use SPUFI without receiving an error message.

Chapter 8. Migrating the DB2 subsystem to compatibility mode

This chapter describes the steps that are necessary to migrate DB2 Version 7 to Version 8 compatibility mode.

After migration, the conversion process is comprised of three progressive catalog levels: compatibility mode, enabling-new-function mode, and new-function mode. *Compatibility mode* is the state of the catalog after the Version 8 migration process is complete. *Enabling-new-function mode* is marked by the beginning and ending of catalog conversion jobs DSNTIJNE and DSNTIJNF. DSNTIJNE enables new-function-mode processing. DSNTIJNF places the DB2 subsystem in new-function mode. *New-function mode* begins after a complete and successful conversion of the DB2 catalog to Unicode.

Before you begin migration to compatibility mode, you need to load the DB2 libraries. This process is described beginning on page 47. You also need to run the installation CLIST. The process is described beginning on page 80. Also see "Migration considerations" on page 304 for information about changes that might affect your migration process.

When you migrate to Version 8, you cannot use new DB2 facilities until you are in new-function mode. When you are in compatibility mode or enabling-new-function mode, you cannot use the new Version 8 facilities.

Attention: If you do not follow the documented procedures, unpredictable results can occur after migration to compatibility mode.

Ensure that your Version 7 subsystem is at the proper service level. Before you migrate to Version 8 compatibility mode, you must have a maintenance level on Version 7 that contains the fallback SPE. If you do not have the fallback SPE applied, your Version 8 compatibility mode migration process will fail.

When you start DB2, the code level of the starting DB2 is compared to the code level required by the current DB2 catalog. If the starting DB2 has a code level mismatch with the catalog, DB2 does not start and a message is issued.

Refer to *DB2 Program Directory*, which is shipped with the product, for keyword specifications for preventive service planning (PSP). Check Information/Access or the ServiceLink facility of IBMLink for PSP information before you migrate. Also check monthly for the most current information about DB2.

Before migrating the first member to Version 8, check the service levels of any coupling facilities running with coupling facility control code (CFCC) at CFLEVEL=7 or CFLEVEL=8. Coupling facilities at CFLEVEL=7 should have service level 1.06 or higher, and coupling facilities at CFLEVEL=8 should have service level 1.03 or higher. If these service levels are not installed, data corruption can occur. No service level requirements exist for the other CFLEVELs.

You can use the z/OS DISPLAY CF command to display the service level of coupling facilities.

The following topics provide additional information:

Migration considerations

Be aware of the following changes that might affect your migration to Version 8 compatibility mode. For information about how migration might affect your DB2 operations, see "Make adjustments for release incompatibilities" on page 317.

DB2 Version 8 publications assume new-function mode

All publications in the DB2 Version 8 library assume that your DB2 subsystem is running in Version 8 new-function mode. Changes to functions, statements, and limits are available in new-function mode unless stated otherwise.

Changed behavior for ORDER BY clause in SELECT statement

If you order a query by a qualified column where the column name is the same as the AS NAME of the column in the select list, DB2 issues an error.

Use triggers instead of field, edit, and validation procedures

It is recommended that you use triggers instead of field, edit, and validation procedures. Triggers can have long names, but field, edit, and validation procedures are limited to names of 18 bytes or smaller.

DB2 treats certain large fixed-length strings as varying-length strings

Fixed-length character host variables cannot be over 255 bytes long. Fixed length graphic host variables cannot be over 127 characters long. When DB2 manipulates a fixed-length character string that is declared as more than 255 bytes long, or a fixed-length graphic string that is declared as more than 127 characters long, DB2 treats that string as a varying-length string.

MEMLIMIT cannot be customized through the installation process

In previous releases of DB2, you could modify the MEMLIMIT setting for the DBM1 address space by setting a value for STG AVAILABLE ABOVE 2GB in panel DSNTIPC. This field has been removed and DB2 now determines the appropriate setting.

DBDs cannot be accessed if DB2 starts in deferred mode

If you start DB2 in a deferred mode, database descriptors (DBDs) cannot be accessed until the restart has completed. If you attempt to load a DBD during DB2 start-up in deferred mode, the DBD is not loaded and DB2 start-up continues.

DB2 LOCATION NAME value

In previous releases of DB2, DB2 could start if a value was not specified in the DB2 LOCATION NAME field in panel DSNTIPR. However, in DB2 Version 8, DB2 will not start if no value has been specified in that field.

Type 1 indexes are not supported

Before you migrate to Version 8, you must convert all type 1 indexes to type 2 indexes. DB2 migration fails if your subsystem contains type 1 indexes. See "Migration step 1: Premigration activities" on page 317 for more information.

Declared global temporary tables need an 8-KB buffer pool

Global temporary tables require an 8-KB buffer pool, which are required to install DB2. Existing jobs that create a table space in the temporary database might also need to be modified.

Declared global temporary tables need an 8-KB table space in the temporary database

If you use declared temporary tables, you must define at least one of the table spaces in the temporary database to have a page size of 8 KB or greater. Member DSNTESQ of the *prefix*.SDSNSAMP library contains a sample query to check your temporary databases.

System-level point-in-time recovery

If you plan to use the BACKUP SYSTEM online utility to take volume copies of the data and logs of your non-data-sharing DB2 subsystem or of your DB2 data sharing group, all of your DB2 data sets must reside on volumes that are managed by DFSMS. The BACKUP SYSTEM utility and its counterpart, the RESTORE SYSTEM utility, require:

- z/OS Version 1 Release 5 or above.
- Disk control units that support ESS FlashCopy®.
- HSM copy pools whose definitions follow the DB2 naming convention.
- SMS copy target storage pools that are defined. (The BACKUP SYSTEM utility enables volume-level backups of a DB2 subsystem that uses these target storage groups.)

Exception: If you use RESTORE SYSTEM with the LOGONLY option, you do not need the preceding requirements. You can perform the restoration manually by using your preferred method, and then run RESTORE to complete the recovery.

Enhanced support for scrollable cursors

Support for scrollable cursors enables dynamic access to data in tables. In Version 7, scrollable cursors required storage space in the temporary database and in segmented table spaces. In Version 8, with dynamic scrollable cursors, this restriction no longer exists, which might result in a decrease in the needed storage.

Changes to space allocations for DB2-managed data sets

The default values for primary space allocations have increased. For non-LOB table spaces and indexes, the default primary space allocation is one cylinder. For LOB table spaces, the default primary space allocation is ten cylinders.

The default values for secondary space allocations can now use a sliding scale. If you specify a value of YES for the field OPTIMIZE EXTENT SIZING on panel DSNTIP7, DB2 uses a sliding scale to determine the secondary allocations for DB2-managed data sets if you explicitly specify SECQTY (with a valid value that is not -1) in the CREATE TABLESPACE or CREATE INDEX statement or in any of the subsequent ALTER TABLESPACE or ALTER INDEX statements.

Using a sliding scale for secondary space allocations might result in increased disk space usage. However, overall, this method generally results in better space utilization and fewer situations in which the maximum number of extents are reached.

Changed default value for DESCRIBE FOR STATIC

During installation of DB2 Version 8, the default for subsystem parameter DESCSTAT on installation panel DSNTIP4 is now YES. If your DB2 UDB for z/OS subsystem or DB2 UDB for Linux, UNIX, and Windows systems uses the new JDBC driver, or if your DB2 UDB Linux, UNIX, and Windows systems uses the new CLI driver, you must set DESCSTAT to YES.

Recommendation: Change the DESCSTAT setting to YES if your DB2 UDB for z/OS subsystem is used as a server for applications that use either of these drivers.

Changed data types and lengths for some catalog columns

Some catalog columns have new data types and lengths. In Version 8, they are now VARCHAR(*n*), where *n* is 8 or greater.

If your application program uses the values of these columns in comparison statements such as a statement that uses a LIKE predicate, you might need to adjust your application program to get the desired results.

Example: Assume that you created a table in Version 7 with the following SQL statement:

```
CREATE TABLE T1 (C1 CHAR(5));
```

Assume that you then issued the following statement in Version 7:

```
SELECT LENGTH(NAME) FROM SYSIBM.SYSTABLES WHERE NAME = 'T1';
```

DB2 Version 7 returns a value of 8.

Now, after converting to Version 8 new-function mode, assume that you create a new table with the following SQL statement:

```
CREATE TABLE T2 (C1 CHAR(5));
```

Assume that you then issue the following statements in Version 8 new-function mode:

```
SELECT LENGTH(NAME) FROM SYSIBM.SYSTABLES WHERE NAME = 'T1';
SELECT LENGTH(NAME) FROM SYSIBM.SYSTABLES WHERE NAME = 'T2';
```

In this example, DB2 returns a value of 8 for the first statement, because the padding characters from Version 7 are not stripped during migration to Version 8. DB2 returns value of 2 for the second statement because T2 was created in Version 8.

Changed data types and lengths for some special registers

Some special registers have new data types and lengths. The changed registers and their new data types and lengths are:
- CURRENT OPTIMIZATION HINT is now VARCHAR(128).
- CURRENT PACKAGESET is now VARCHAR(128).
- CURRENT SQLID is now VARCHAR(8).
- USER is now VARCHAR(8).
- CURRENT PATH is now VARCHAR(2048).

If your application program uses the values of these registers in comparison statements such as a LIKE predicate, you might need to adjust your application program to get the desired results.

SQL reserved words may be used in delimited identifiers for procedure names

In Version 8, you can use SQL reserved words in delimited identifiers for procedure names. See DB2 SQL Reference for more information.

Encoding schemes of string parameters for routines

The new PARAMETER CCSID clause allows you to define the encoding scheme of all string parameters for user-defined functions and stored procedures at the same time. In previous versions, you needed to define a CCSID for each string parameter if you wanted an encoding scheme other than the default. Also, EBCDIC is no longer the default encoding scheme for system-defined parameters. DB2 now uses the same encoding scheme for both user-specified and system-generated string parameters.

Modify RUNSTATS jobs

After you migrate to Version 8, some existing RUNSTATS jobs might fail if data-partitioned secondary indexes are defined on the tables on which they run. RUNSTATS jobs on data-partitioned secondary indexes require sort operations; if the sort package that you use does not dynamically allocate sort work data sets, you need to modify these RUNSTATS jobs to allocate the sort work data sets. You can modify the RUNSTATS jobs with the SORTDEVT and SORTNUM keywords, or you can add STATWK*nn* DD statements to the JCL.

More history statistics are collected

If you specify SPACE or ACCESSPATH for the STATISTICS HISTORY parameter on panel DSNTIPO, DB2 might insert more statistics into the catalog statistics history tables. For example, DB2 inserts statistics when you run a utility with the UPDATE(ACCESSPATH) or UPDATE(SPACE) parameter but without the HISTORY parameter.

Creating tables with DBCS and mixed columns

You can no longer create extended binary-coded decimal interchange code (EBCDIC) tables with GRAPHIC, VARGRAPHIC, or DBCLOB columns when the setting for installation option MIXED DATA is NO. You also cannot alter EBCDIC tables to add GRAPHIC, VARGRAPHIC, or DBCLOB columns when MIXED DATA is NO.

Consider increasing IDBACK and CTHREAD

Because utilities might use additional threads, you should consider increasing the values of the IDBACK and CTHREAD subsystem parameters. Increasing these parameter values can help you avoid failure of some utilities due to increased thread usage. An increase also supports the additional parallelism that is associated with the utilities.

Support for DB2-established data space for cached dynamic statements is deprecated

In Version 8, support for a DB2-established data space for cached dynamic statements is deprecated. You can no longer specify the parameters EDMDSPAC or EDMDSMAX during installation or migration. A new EDM statement cache is provided for cached dynamic statements. See “CLIST calculations panel 1: DSNTIPC” on page 238 for a description of the parameters for the new EDM statement cache.

Consider changing EDM pool size

Cached dynamic statements and database descriptors are in a separate pool in Version 8, which could result in decreased storage requirements. You can change the EDM pool size by modifying the EDMPOOL STORAGE SIZE field on installation panel DSNTIPC, and then stopping and restarting DB2. You can also modify the EDM pool size without stopping and restarting DB2 by using the SET SYSPARM command. However, using the SET SYSPARM command might result in a pool that is not contiguous, which is less efficient.

Customized DB2I defaults can be migrated

You can migrate a DB2I TSO IPSF profile member from a prior release to the current release. The DSNEMC01 CLIST uses the values that are specified on installation panel DSNTIPF and stores the results in the ISPF profile member DSNEPROF. You can migrate any customized DSNEPROF members from Version 7 to Version 8. However, you need to examine any new or changed default panel values to ensure that your customized values are still valid.

LANGUAGE COMPJAVA no longer supported for stored procedures

After migrating to Version 8, you can no longer define or run COMPJAVA stored procedures. Convert LANGUAGE COMPJAVA stored procedures to LANGUAGE JAVA by following these steps:

1. Use ALTER PROCEDURE to change the LANGUAGE and the WLM ENVIRONMENT. The EXTERNAL NAME clause must also be specified. Use the following example as a model:

   ```
   ALTER PROCEDURE SYSPROC.JAVADVR
   LANGUAGE JAVA EXTERNAL
   NAME 'display.display.main'
   WLM ENVIRONMENT WLMENVJ;
   ```

   You must specify a valid language option when issuing any ALTER PROCEDURE statement for a procedure that was defined with LANGUAGE COMPJAVA. If you do not, DB2 issues an error.
2. Ensure that the WLM environment is configured and that the required JVM is installed.
3. Ensure that the .class file that is identified in the EXTERNAL NAME clause of the ALTER PROCEDURE is present in one of the following places:
   - In a JAR that was installed to DB2 by an invocation of the INSTALL_JAR stored procedure
   - In a directory in the CLASSPATH ENVAR of the data set that is named on the JAVAENV DD statement of the WLM stored procedures address space JCL

DSNWZP runs in WLM-established stored procedure address space

In DB2 Version 8, the DB2-supplied stored procedure DSNWZP is defined to run in a WLM-established stored procedure address space that uses external module DSNWZP. If you ran DSNWZP in a WLM-established stored procedure address space in DB2 Version 7, you redefined DSNWZP to use external module DSNWZPR. If you do not use job DSNTIJSG to define DB2-supplied stored procedures in DB2 Version 8, you must alter stored procedure DSNWZP to use external module DSNWZP. The SQL ALTER statement is:

```
ALTER PROCEDURE SYSPROC.DSNWZP EXTERNAL NAME DSNWZP;
```

Important: You must set NUMTCB=1 in your WLM environment for DSNWZP.

Support for DB2-established stored procedure address spaces is deprecated

In Version 8, support for DB2-established address spaces is deprecated. You can no longer specify the NO WLM ENVIRONMENT option when you create or alter stored procedure definitions. Although existing stored procedures can still run in a

DB2-established stored procedure address space, you should move your stored procedures to WLM environments as soon as possible. For more information about moving stored procedures, see Part 5 (Volume 2) of *DB2 Administration Guide*.

New precompiler option for string host variables

In previous releases of DB2, if you selected a value from a character column into a C or C++ host variable of the nul-terminated character form, and the length of the host variable was longer than the length of the value, DB2 padded the string with blanks and inserted the nul-terminator after the blanks. In Version 8, the DB2 default behavior is to not pad the string with blanks. If you want to produce blank-padded strings, as in previous releases, specify YES in field PAD NUL-TERMINATED in installation panel DSNTIP4, or precompile your program with the PADNTSTR option.

Example: Suppose that a CHAR(5) column contains the value ’ABCDE’. You select the value into a C host variable that is defined as char hv. In previous releases of DB2 and DB2 UDB in non-z/OS environments, after you performed the SELECT, the value in hv was X'C1C2C3C4C500'. In Version 8, if you do not change the installation default, the value in hv is X'C1C2C3C4C5404040404000'.

New SYSIBM.SYSROUTINES column for encoding scheme

After you successfully migrate to Version 8, the encoding scheme that is used for system-generated parameters for procedures and functions is stored in a new column in SYSIBM.SYSROUTINES. This information was previously stored in a special row in the SYSIBM.SYSPARMS table.

LANGUAGE REXX sets PROGRAM_TYPE column in SYSIBM.SYSROUTINES

If you specify LANGUAGE REXX, DB2 sets the PROGRAM_TYPE column in SYSIBM.SYSROUTINES to ’M’. You cannot override this value by specifying PROGRAM TYPE MAIN or PROGRAM TYPE SUB. The procedure will continue to run as in Version 7, where all REXX procedures were treated as a main procedure.

DB2 start-up and precompilation require a user-supplied DSNHDECP module

Installation job DSNTIJUZ generates the data-only load module DSNHDECP. It contains the application programming defaults. DB2 is shipped with a default DSNHDECP for compatibility with older applications. You cannot start DB2 or precompile applications with the default DSNHDECP. During DB2 start-up processing or for jobs that precompile a DB2 application, a DSNHDECP module that is customized by the installation CLIST must exist in a library that is before the library that contains the default DSNHDECP module in the STEPLIB concatenation, the JOBLIB concatenation, or the system link list.

To load the user-specified DSNHDECP module, take the following actions:

1. Ensure that the user-specified DSNHDECP resides in a library, usually *prefix*.SDSNEXIT, that is concatenated before the *prefix*.SDSNLOAD library where the DB2-supplied DSNHDECP resides.
2. The DSNH call must include the PCLOAD parameter `PCLOAD('*(DSNHPC)')`.

CCSIDs in DSNHDECP must be valid

All CCSIDs in the DSNHDECP module must be valid. During start-up processing, if DB2 detects invalid CCSID values, DB2 issues a message and terminates.

New data-only load module DSNHMCID

The new data-only load module DSNHMCID contains EBCDIC CCSIDs for offline message conversion. Version 8 utilities and applications must have access to this module. You can provide access to DSNHDECM in one of the following ways:

- Permit the DSNHDECM module to reside in SDSNLOAD.
- Include the library SDSNEXIT before SDSNLOAD in the system link list.
- Verify that all jobs and tasks that use DB2 utilities or call DB2 application programs are updated to STEPLIB or JOBLIB to SDSNEXIT.

Plans and packages bound prior to DB2 Version 2 Release 3

If you have plans and packages that were bound prior to DB2 Version 2 Release 3, DB2 will autobind these packages. Thus, you may experience an execution delay the first time that such a plan is loaded. Also, DB2 may change the access path due to the autobind, potentially resulting in a more efficient access path.

Multiple calls to the same stored procedure

In previous versions of DB2, if a stored procedure was called twice from the same program and at the same nesting level, DB2 closed the result set cursors and released storage for the first instance of the stored procedure before making the second call. In DB2 Version 8, if the requester and the server are both DB2 Version 8 subsystems in new-function mode, when the second call is made, both instances of the stored procedure can run at the same time. DB2 does not close the result sets from the first call or release storage for the first instance of the stored procedure. If you make multiple calls to the same stored procedure in the same application, be aware of the following considerations:

- A DESCRIBE PROCEDURE statement describes the last instance of the stored procedure.
- The ASSOCIATE LOCATOR statement works on the last instance of the stored procedure. You should issue an ASSOCIATE LOCATOR statement after each call to the stored procedure to provide a unique locator value for each result set.
- The ALLOCATE CURSOR statement must specify a unique cursor name for the result set of each instance of the stored procedure. Otherwise, you will lose the data from the result sets that are returned from prior instances or calls to the stored procedure.

External stored procedures and user-defined functions can return any valid SQLSTATE value

In previous versions of DB2, an external stored procedure or user-defined function could return only SQLSTATE values of the form ’01Hxx’, ’38xxx’, ’00000’, or ’02000’. In DB2 Version 8, an external stored procedure or user-defined function can return any valid SQLSTATE value.

Programs called by a stored procedure require packages

In previous versions of DB2, if a stored procedure called a subprogram using a host language call, and that subprogram contained SQL statements, DB2 did not require a package for that subprogram at the location where the stored procedure was defined. In DB2 Version 8, if a stored procedure calls a subprogram that contains SQL statements, and a package does not exist for that subprogram at the server where the stored procedure is defined, DB2 issues an error message.

Port of entry name changed

If you are using z/OS Version 1 Release 5, TCP/IP Network Access Control, and the RACF SERVAUTH class is active, the port of entry name that is passed to RACF for verification is the point of entry security zone name. The port of entry security zone name is defined in the TCP/IP Network Access Control profile. In previous releases of DB2, the port of entry name that was passed to RACF was the string ’TCPIP’.

New name for type 1 inactive threads and type 2 inactive threads

Type 1 inactive threads are now referred to as inactive DBATs. Type 2 inactive threads are now referred to as inactive connections.

Column names and labels in SQLDA SQLNAME field for statements involving UNION

Prior to Version 8, the result column name in a SQLNAME field of the SQLDA for a statement involving a UNION reflected the column name or label of the first sub-query in the statement. In Version 8, if labels are used, DB2 returns the label of the column in the first sub-query. If labels are not used, the result column name will only be returned if the column name is the same across all sub-queries in the statement. You can temporarily override this behavior by setting subsystem parameter UNION_COLNAME_7 to YES.

IFCID 197 is no longer supported

In Version 8, IFCID 197 is no longer supported. If you make a READS call for IFCID 197, DB2 issues return code 8 and reason code 00E60821.

Change data capture cannot be enabled on catalog tables during enabling-new-function mode

During enabling-new-function mode processing, change data capture is disabled on most catalog tables. You cannot re-enable change data capture until your DB2 subsystem is in Version 8 new-function mode.

DB2 Version 8 requires IRLM 2.2

IRLM 2.2 is delivered with DB2 Version 8. You must use the DB2–supplied IRLM procedure. To modify an existing IRLM procedure for use with IRLM, you must do the following:

- Add the MEMLIMIT parameter with the appropriate setting.
- Specify YES for the PC parameter.
- Specify 0 for the MAXCSA parameter.

See “IRLM panel 2: DSNTIPJ” on page 188 for more information about these parameters.

Detailed tracking of DB2 measured usage is disabled

In previous releases of DB2, DB2 automatically used detailed tracking of measured usage. In Version 8, subsystem parameter SMF89 controls whether DB2 uses detailed tracking of measured usage. The default value is NO, which means that

DB2 does not do detailed measured usage tracking. If the SMF type 89 record is activated, only high-level tracking is recorded in the SMF type 89 record.

Programming language support has changed

Programming language support in DB2 Version 8 has changed. For a list of all supported languages, see *DB2 Program Directory*. If your DB2 Version 7 subsystem uses languages other than those specified in *DB2 Program Directory*, you must migrate to a supported release of that language before migrating your DB2 subsystem to Version 8.

New return code for -START DATABASE, -STOP DATABASE, and -DISPLAY DATABASE commands

In previous releases of DB2, if the object of a -START DATABASE, -STOP DATABASE, or -DISPLAY DATABASE command was not found, the command completed with a return code of 12. In Version 8, if the object of a -START DATABASE, -STOP DATABASE, or -DISPLAY DATABASE command is not found, the command completes with a return code of 0. The behavior of these three commands is now similar to the behavior of other commands.

Views might be marked with view regeneration errors

DB2 automatically regenerates views that reference the DB2 catalog. However, as a result of changes to the catalog, some views may be marked with view regeneration errors. Views that are marked with view regeneration errors may be usable, but will not be automatically regenerated. You must manually regenerate these views. See "Migration step 25: Verify views" on page 342 for more information about regenerating views.

Resource limit facility tables might need to be updated

In Version 5 and earlier, the resource limit facility (RLF) table has eight columns. In subsequent releases, the RLF table has eleven columns. If your RLF table is not updated, DB2 Version 8 might terminate when you issue a dynamic SQL statement that involves the RLF table.

To determine whether your RLF tables are updated, issue the following SQL statement:

```
SELECT TBCREATOR, TBNAME, COUNT(*) AS NUM_COLS
    FROM SYSIBM.SYSCOLUMNS
    WHERE TBNAME LIKE 'DSNRLS%'
    GROUP BY TBCREATOR, TBNAME
    HAVING COUNT(*) <> 11
```

If this SQL statement returns results, this means that the RLF tables are not yet updated. Use the following SQL statement to update the returned RLF table:

```
ALTER TABLE table
  ADD RLFASUERR INTEGER;
ALTER TABLE table
  ADD RLFASUWARN INTEGER;
ALTER TABLE table
  ADD RLF_CATEGORY_B CHAR(1) NOT NULL WITH DEFAULT;
```

where *table* is the name of the returned RLF table.

Changed default values for subsystem parameters

The default values for several parameters have changed. The new values are listed in Table 53.

Table 53. Subsystem parameters with new default values

Panel	Field	Parameter	Version 7 default value	Version 8 default value
DSNTIP7	USER LOB VALUE STORAGE	LOBVALA	2048	10240
DSNTIPE	MAX USERS	CTHREAD	70	200
	MAX REMOTE ACTIVE	MAXDBAT	64	200
	MAX REMOTE CONNECTED	CONDBAT	64	10000
	MAX TSO CONNECT	IDFORE	40	50
	MAX BATCH CONNECT	IDBACK	20	50
DSNTIPN	DDF/RRSAF ACCUM	ACCUMACC	NO	10
DSNTIP8	CACHE DYNAMIC SQL	CACHEDYN	NO	YES
DSNTIPP	PLAN AUTH CACHE	AUTHCACH	1024	3072
DSNTIPL	LOG APPLY STORAGE	LOGAPSTG	0	100
	CHECKPOINT FREQ	CHKFREQ	50000	500000
DSNTIPA	BLOCK SIZE	BLKSIZE	28672	24576
DSNTIPR	DDF THREADS	CMTSTAT	ACTIVE	INACTIVE
	IDLE THREAD TIMEOUT	IDTHTOIN	0	120
	EXTENDED SECURITY	EXTSEC	NO	YES
DSNTIP5	TCP/IP KEEPALIVE	TCPKPALV	ENABLE	120
DSNTIPC	MAXIMUM OPEN DATA SETS	DSMAX	3000	10000
	EDMPOOL STORAGE SIZE	EDMPOOL	7312	32768[1]

Note:

[1] The installation CLIST calculates the default value for the EDMPOOL parameter.

If you specify SMFSTAT=YES, DB2 starts traces for SMF classes 1, 3, 4, 5, and 6. In DB2 Version 7, specifying SMFSTAT=YES only started traces for SMF classes 1, 3, 4, and 5.

If the values that you specified for these parameters are lower than the new default values, you might want to increase your values.

Subsystem parameter CLAIMDTA has been removed

Subsystem parameter CLAIMDTA has been removed. In Version 8, DB2 always operates as if CLAIMDTA=YES.

Migrating a data sharing group

Before you migrate to compatibility mode, ensure that maintenance through the Version 8 fallback SPE is applied to all started DB2 members. If the fallback SPE is not on all active group members, Version 8 does not start but issues a message. If you have quiesced members in your data sharing group, you do not need to apply the fallback SPE to the quiesced member.

Start only one DB2 member for migration processing. During the migration, other group members can be active. However, other active group members may experience delays or timeouts if they attempt to access catalog objects that are locked by migration or enabling-new-function mode processing. After migration completes on the first member, you can migrate the other data sharing group members.

Migrating a data sharing group requires careful planning:

1. Read the information about migration considerations in this book and also in Chapter 3 of *DB2 Data Sharing: Planning and Administration*.
2. Make a plan to minimize the amount of time that some members operate at the Version 7 level and others operate at the Version 8 compatibility mode level.
3. Apply the fallback SPE to the Version 7 load library for each non-quiesced member in the data sharing group. For best availability, you can apply the SPE to one member at a time. While your data sharing group is in Version 7, you can have Version 7 subsystems with the SPE running at the same time as subsystems that are without the SPE. Stop and restart each member to activate the change. If members of the data sharing group are quiesced and no longer in active use, you do not need to apply the fallback SPE to those members
4. Follow the procedure about migrating the data sharing group in *DB2 Data Sharing: Planning and Administration*. You must complete the migration of the first member of the data sharing group to Version 8 compatibility mode before starting any other members at the Version 8 level.
5. To prepare for fallback from Version 8 compatibility mode, keep the subsystem parameter load module that is used by Version 7.
6. After all members have migrated to Version 8 compatibility mode, remain in compatibility mode until your data sharing group has processed a full range of typical work. The period of time that a data sharing group needs to remain in Version 8 compatibility mode varies depending on the size of the data sharing group and the complexity of its typical work.

The CLIST edits different jobs for enabling data sharing and migrating a data sharing member. See Chapter 3of *DB2 Data Sharing: Planning and Administration* for the list of jobs that are edited for each case.

Work file database size calculations

The migration job DSNTIJTC creates and updates indexes on catalog tables. These indexes are created and updated sequentially during migration. The work file database is used for the sort of each index; DB2 needs enough work file storage to sort the largest of the indexes in Table 54. The migration fails if you do not have enough storage. Therefore, ensure that you have enough space before you begin. See “Work file database storage requirements” on page 22 for information about space requirements.

Table 54 shows the indexes that are new and changed for existing catalog tables.

Table 54. Indexes that are added or updated sequentially using the work file database

Catalog table name	Index name	Column names
SYSIBM.SYSCOLAUTH	SYSIBM.DSNACX01	CREATOR, TNAME, COLNAME
SYSIBM.SYSFOREIGNKEYS	SYSIBM.DSNDRH01	CREATOR, TBNAME, RELNAME
SYSIBM.SYSINDEXES	SYSIBM.DSNDXX04	INDEXTYPE
SYSIBM.SYSRELS	SYSIBM.DSNDLX02	CREATOR, TBNAME
SYSIBM.SYSSEQUENCESDEP	SYSIBM.DSNSRX02	BSCHEMA, BNAME, DTYPE
SYSIBM.SYSTABAUTH	SYSIBM.DSNATX04	TCREATOR, TTNAME
SYSIBM.SYSTABLEPART	SYSIBM.DSNDPX03	DBNAME, TSNAME, LOGICAL_PART
SYSIBM.SYSTABLES	SYSIBM.DSNDTX03	TBCREATOR, TBNAME
SYSIBM.SYSVIEWDEP	SYSIBM.DSNGGX04	BCREATOR, BNAME, BTYPE, DTYPE

Release coexistence

This section highlights considerations for coexistence between Version 7 and Version 8 in a data sharing environment and in a distributed environment. In a data sharing environment, coexistence is limited to Version 8 compatibility mode with Version 7.

IRLM

As you apply IRLM service to members of a data sharing group, some members run with the newer service level, and some run with the older service level. A mix of service levels can raise issues that you must consider. For more information about IRLM coexistence, see *DB2 Data Sharing: Planning and Administration*.

DISPLAY GROUPBUFFERPOOL output

Because the DISPLAY GROUPBUFFERPOOL command output in Version 8 returns both operational and actual coupling facility levels, the command output in a coexistence environment depends on the member on which the command was issued. If the command is issued from Version 7, only the operational coupling facility level is displayed.

The coupling facility batching commands RFCOM and WARM are not used in a coexistence environment.

Distributed environment

DB2 UDB for z/OS communicates in a distributed data environment with Version 6 and Version 7 of DB2, using either DB2 private protocol access or DRDA access. However, the distributed functions that are introduced in Version 8 of DB2 UDB for z/OS can be used only when using DRDA access.

Other DRDA partners at DRDA level 4 can also take advantage of the functions that are introduced in Version 8 of DB2 UDB for z/OS.

Data sharing

DB2 can support both Version 7 and Version 8 members in compatibility mode in a data sharing group. To support both releases, you must first apply the fallback SPE to all Version 7 members of the group. Release coexistence begins when you migrate the first data sharing member to Version 8. You must successfully migrate the first data sharing member to Version 8 before attempting to migrate the other data sharing members.

For the best availability, you can migrate the members to Version 8 one member at a time. When developing your migration plan, remember that most new functions that are introduced in Version 8 are not available to any members of the group until all members are migrated to Version 8 and until all members are in new-function mode.

TSO, CAF, and RRSAF logon procedures: You can attach to either release of DB2 with your existing TSO, CAF, or RRSAF logon procedures, without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you **must** update those procedures and jobs to point to the latest level of DB2 load libraries. If you forget to update those procedures and jobs before migrating to any release subsequent to Version 8, those procedures and jobs can no longer work in that subsequent release.

For a detailed list of considerations for a data sharing group with multiple DB2 releases, see Chapter 3 of *DB2 Data Sharing: Planning and Administration*.

Migration step 1: Premigration activities

This step is included as a reminder to perform the following tasks before migrating to Version 8 compatibility mode:
- "Save critical access paths (optional)"
- "Examine all new and changed values for DB2I panels"
- "Make adjustments for release incompatibilities"
- "Ensure that Version 7 sample objects are available" on page 321
- "EBCDIC and ASCII CCSID must be non-zero" on page 321
- "Perform premigration queries (DSNTIJPM)" on page 321

Save critical access paths (optional)

Sometimes changes between releases of DB2 cause unwanted access path changes. Consult with your performance analysts to determine which queries are especially critical and ensure that there is a PLAN_TABLE that contains the good access path.

Run EXPLAIN on your queries before migrating. Because EXPLAIN requires a rebind, your access paths might change. Therefore, extract the needed queries and then run EXPLAIN on them under a different application or program name. This action protects the existing application while the access path information is obtained. Then, after the access paths for the extracted queries are validated, you can update the APPLNAME or PROGNAME columns of the PLAN_TABLE to the correct name. For more information about using access path hints, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Examine all new and changed values for DB2I panels

During a migration, the DB2I default panels DSNEOP01 and DSNEOP02, are not initialized with the values that are specified during the installation CLIST process. The DB2I panel variables in the ISPF profile from the previous release are used on the current release. Any customized DSNEPROF members are migrated from Version 7 to Version 8. However, you must examine any new or changed default panel values to ensure that your customized values are still valid.

Make adjustments for release incompatibilities

There are some changes in Version 8 that might affect your DB2 operations after migrating.

Adjust application programs

You might need to adjust your application programs because of the release incompatibilities that this section describes.

Adjust trace applications: If you have trace applications that use statement-length fields, you might need to change them to use 4-byte statement length fields.

Adjust user-defined function calls for new built-in functions: Several new built-in functions are available. If you have user-defined functions, invoke them with a fully qualified name to avoid calling built-in functions that might have the same name. If the user-defined functions are not invoked with a fully qualified name and SYSIBM is earlier in the SQL path, the built-in function is selected instead of the user-defined function.

Changed defaults for utilities: The default values for several utilities' options have changed in Version 8. SORTKEYS is the default for the REORG, LOAD, and REBUILD utilities. SORTDATA is the default for the REORG utility.

Changed behavior for DISPLAY LOCATION command: If you specify an empty parameter for the DISPLAY LOCATION command, such as DISPLAY LOCATION(), the command fails and DB2 issues message DSN9010I. In Version 8, you must specify a parameter for this command.

Changed output for DISPLAY GROUP command: In Version 8, the DISPLAY GROUP command displays the status of your group in compatibility mode, enabling-new-function mode, and new-function mode.

Changed behavior for TRANSLATE function: If your query references the catalog, uses the TRANSLATE built-in function, and specifies a translate table, you might need to change the translate table. In some cases, the TRANSLATE functions might not have the same behavior as in Version 7. For example, some accented characters which had a single-byte EBCDIC value in Version 7 have a double-byte Unicode value. If you perform a TRANSLATE function on a string that contains such a character, the function will not return the expected results.

Changed output for DISPLAY GROUPBUFFERPOOL command: In Version 8, the DISPLAY GROUPBUFFERPOOL command displays both the operational coupling facility level and the actual coupling facility level. In Version 7, only the operational coupling facility level was displayed.

Changed parameter length for BLOB, CLOB, and DBCLOB functions: The lower limit for these functions is now 1 for consistency with the VARCHAR and VARGRAPHIC functions. You cannot invoke these built-in functions with an explicit parameter length of 0. If you specify 0, DB2 returns an error. If the input string is empty and an explicit length is not specified, the length attribute of the result is 1.

Changed input for GRAPHIC, VARGRAPHIC, and DBCLOB: The input string for the GRAPHIC, VARGRAPHIC, and DBCLOB functions cannot be BIT data, regardless of the encoding scheme of the data. If the input string is EBCDIC BIT data, DB2 returns an error.

Changed rules for procedure and function names: In Version 8, DB2 enforces the following rules for procedure and function names:

- If your routine is written in a language other than Java, the external name is a load module which must be less than or equal to 8 bytes. The external name must contain characters that are valid for a z/OS load module.
- A procedure name cannot consist of a single asterisk.

You must update your application programs to reflect these changes. If you do not, DB2 issues an error message. See *DB2 SQL Reference* for more information about these changes.

Truncation of CHAR data: Prior to Version 8, DB2 issued an error when applications invoked the CHAR function with string input data greater than 255 bytes. In Version 8, DB2 truncates the data to 255 bytes and issues a warning if non-blank characters are truncated.

SQLDA might contain truncated data: In Version 8, the length of many names has been extended. However, for compatibility with prior releases, the length of name fields in the SQLDA is not changing. Truncation of names that are stored in the SQLDA might occur with distinct name types. To avoid truncation of distinct type names in the SQLDA, you should not use distinct name types longer than 30 bytes. For more information, refer to *DB2 SQL Reference*.

LOCKPART clause is deprecated: The LOCKPART clause on ALTER or CREATE TABLESPACE is deprecated in Version 8, although the syntax is still supported for compatibility purposes. In previous releases, LOCKPART determined whether individual partitions would be locked. The previous default value for LOCKPART locked the entire table space with a lock on the last partition. In Version 8 new-function mode, individual portions of partitioned table spaces, including those that were created in Version 7 or earlier, are locked as they are accessed. In a data sharing environment, all members must be in enabling-new-function or new-function mode before this change takes effect.

UTLRSTRT is no longer supported: The subsystem parameter UTLRSTRT is no longer supported. When possible, DB2 attempts to restart online-restartable utilities, regardless of whether the RESTART keyword is specified. See *DB2 Utility Guide and Reference* for more information about automatically restarting utilities.

PKGLDTOL is no longer supported: The subsystem parameter PKGLDTOL is no longer supported. DB2 Version 8 requires the package or plan for applications with the following SQL statements:

- COMMIT
- CONNECT
- DESCRIBE TABLE
- RELEASE
- ROLLBACK
- SET CONNECTION
- SET *host-variable* = CURRENT SERVER
- VALUES CURRENT SERVER INTO *host-variable*

You must bind the DBRM into a plan or package.

Restriction on DB2 private protocol applications: DB2 limits the SQL statements that a private protocol application can include to statements that were added to DB2 before Version 8.

SQL reserved words: Version 8 has several new SQL reserved words. Refer to *DB2 SQL Reference* for the list, and adjust your applications accordingly.

Changed return code for message DSNU185: The return code for DSNU185 has changed from return code 8 to return code 0, allowing processing to continue. If you have applications that scan the return code of this message, you might need to modify them.

Input parameter markers of a prepared statement are always nullable: In previous versions of DB2, the SQLTYPE field could be set to nullable or non-nullable. In Version 8, the SQLTYPE field is always nullable. Use DESCRIBE to view the SQLTYPE field.

Savepoint names cannot begin with 'SYS': A savepoint name cannot begin with *SYS*. If your application has a savepoint with a name beginning with *SYS*, DB2 issues an error message.

Changed behavior for ALTER TABLESPACE: When issuing the ALTER TABLESPACE statement, partition options that are specified after an ALTER PARTITION clause will affect only the specified partition. If you do not specify an option following the specification of a partition, an error is issued.

Changed behavior for ALTER INDEX: If you do not specify an option following the ALTER PARTITION clause, a warning is issued.

EXTERNAL clause for ALTER PROCEDURE and ALTER FUNCTION requires NAME: In Version 7, the EXTERNAL clause did not require the NAME keyword followed by a value on the ALTER statement. In Version 8, you must specify NAME and a value if you specify EXTERNAL. Update your application programs to include the NAME keyword and a value when you specify the EXTERNAL clause on an ALTER statement. If you do not, DB2 issues an error message. See *DB2 SQL Reference* for more information.

Invalidation of statements that reference the catalog: In Version 8, DB2 may invalidate plans and packages that contain statements that reference the catalog. See job DSNTESQ for more information about these plans and packages.

SYSIBM.SYSDUMMY1 is recreated: During execution of job DSNTIJNE in Version 8 enabling-new-function mode processing, DB2 drops and re-creates the SYSIBM.SYSDUMMY1 table. If your plans and packages reference this table, they will be invalidated. DB2 automatically rebinds the invalidated plans and packages when the plans and packages are next references. An automatic rebind may change the access path.

Invalid uses of host variables are not supported: Validation of the attributes of host variables used in PREPARE statements is improved. You may need to update your application programs.

Example: If the defined length of the host variable is less than the length of the actual data, DB2 issues an error message. Update your application program to specify the correct length of the host variable.

Using the PARAMETER STYLE clause in CREATE PROCEDURE statement: In Version 7, you did not need to specify PARAMETER STYLE before an explicit parameter style such as SQL, DB2SQL, JAVA, GENERAL, or GENERAL WITH NULLS. In Version 8, you must specify PARAMETER STYLE ahead of any explicit parameter style, or DB2 issues an error message. If you omit the PARAMETER STYLE clause entirely, the default is PARAMETER STYLE SQL.

User IDs must have SYSOPR authority: In Version 7, DB2 commands that were issued from the z/OS console or TSO SDSF were previously associated with the SYSOPR user ID. In Version 8, these commands are associated with the primary user ID that issued them. You must grant SYSOPR authorization to these user IDs or public.

Update CCSIDs in DBINFO: If you have defined an external function or procedure with DBINFO, you might need to update the CCSIDs in DBINFO. In Version 7, CCSID fields in DBINFO were set to the CCSIDs of the invoking statement. In Version 8, a single set of three CCSIDs might not reflect the CCSIDs of a statement that invokes an external function or procedure defined by DBINFO. Adjust the CCSIDs in DBINFO, then recompile and rebind the routine that references them. See "Migration step 20: Ensure that routines use Version 8 DBINFO" on page 339 for more details.

START DB2 ACCESS(MAINT) restricts access to installation SYSADM or installation SYSOPR: DB2 Version 8 enforces the restriction of START DB2 ACCESS(MAINT) that limits DB2 access to those user IDs that have installation

SYSADM or installation SYSOPR authorization. Prior versions of DB2 did not enforce this restriction. For more information about the ACCESS(MAINT) option of START DB2, see *DB2 Command Reference*.

Ensure that Version 7 sample objects are available

If you no longer have the Version 7 IVP jobs, you need to run the Version 7 installation CLIST to regenerate them. If you dropped the Version 7 sample database (by running job DSNTEJ0), you need to run the Version 7 IVP jobs through phase 3 before you start the migration to Version 8 compatibility mode. If you don't have the Version 7 jobs available during migration, you will not have a DB2-supported IVP to verify a successful migration to Version 8 compatibility mode.

Ensure that no utility jobs are running

In Version 8, you can restart or terminate a utility only on the same release on which it was started. Any outstanding utilities prior to Version 8 cannot be restarted or terminated after you have migrated from Version 7 to Version 8 compatibility mode. To ensure that you do not have outstanding utility jobs, issue the DISPLAY UTILITY(*) command.

EBCDIC and ASCII CCSID must be non-zero

You must specify a non-zero value for EBCDIC and ASCII CCSIDs on installation panel DSNTIPF. The altering of CCSIDs can be very disruptive to a system. Converting to a CCSID that supports the euro symbol is potentially less disruptive because specific pre-euro CCSIDs map to specific CCSIDs for the euro. See "Converting to the euro symbol" on page 518 for the detailed steps. Converting to a different CCSID for other reasons, particularly when a DB2 subsystem has been operating with the wrong CCSID, could render data unusable and unrecoverable.

Recommendation: Never change CCSIDs on an existing DB2 subsystem without specific guidance from IBM Software Support.

Perform premigration queries (DSNTIJPM)

Job DSNTIJPM performs premigration queries to the Version 7 catalog. DSNTIJPM identifies user-defined DB2 catalog indexes that reside on user-managed storage. For each of these indexes, you need to define a shadow data set. DB2 uses the shadow data set during job DSNTIJNE when it runs REORG to convert the catalog table spaces to DB2 Version 8 new-function mode.

DSNTIJPM also generates SQL statements or utility statements to remove or correct incompatibilities. Job DSNTIJPM checks for the following incompatibilities:

- Use of data capture on DB2 catalog tables. The CATMAINT and CATENFM jobs disable data capture, which you must reactivate at a later time. For more information about data capture, see "Migration step 21: Enable change data capture" on page 339.
- Use of selective partition locking on partitioned table spaces. Selective partition locking is deprecated in Version 8, although it is still supported for compatibility.
- Partitioned table spaces that have a truncated limit key. A truncated limit key can result in an abend if the same limit key value is used for different partitions.
- Stored procedures that use LANGUAGE COMPJAVA. Stored procedures that use LANGUAGE COMPJAVA cannot be defined or run after migrating to DB2

Version 8. For more information about LANGUAGE COMPJAVA, see "LANGUAGE COMPJAVA no longer supported for stored procedures" on page 309.

- Stored procedures that use the DB2–established stored procedures address space. The DB2–established stored procedures address space is deprecated in Version 8. Existing stored procedures that use the DB2–established stored procedures address space will run in Version 8, but you cannot create or alter stored procedures using the NO WLM ENVIRONMENT clause. For more information about the deprecation of the DB2–established stored procedures address space, see "Support for DB2-established data space for cached dynamic statements is deprecated" on page 308.
- Use of the DSNWZPR module by the DB2–supplied stored procedure DSNWZP. In Version 8, DSNWZP requires a WLM stored procedure address space. Users of DSNWZPR must revert to the external module DSNWZP. For more information about DSNWZP, see "DSNWZP runs in WLM-established stored procedure address space" on page 309.
- The Version 7 sample database. To verify migration to Version 8 compatibility mode, run portions of the Version 7 sample jobs. The sample jobs require the Version 7 sample database. If the Version 7 sample database is missing, run the phase 1 and phase 2 sample jobs in Version 7 before migrating to Version 8 compatibility mode.
- Evidence of more than one code page within the same encoding scheme.
- Plans and packages for routines that need to be rebound because of a change in the DBINFO control block. For more information about DBINFO, see "Migration step 20: Ensure that routines use Version 8 DBINFO" on page 339.
- Type 1 indexes. If you do not remove type 1 indexes from your Version 7 catalog, you will not be able to migrate to Version 8 compatibility mode. J

Important: Because type 2 indexes often take more space than type 1 indexes, you should first determine if you need to allocate more space for the indexes. Refer to Section 1 of *DB2 Administration Guide* for more information. Based on these calculations, if your index space is not large enough, delete and redefine your data sets with a larger allocation. For more information about increasing your data set allocations, see *z/OS DFSMSdfp Storage Administration Reference*.

You can convert to type 2 indexes in either of the following ways:

- Use the CONVERT TO TYPE 2 option of the ALTER INDEX SQL statement. This method only works with catalog indexes, not directory indexes.
  1. Enter the SQL statement ALTER INDEX with the CONVERT TO TYPE 2 option.
  2. Run the utility REBUILD INDEX. See *DB2 Utility Guide and Reference* for more information on REBUILD INDEX.
     Recommendation: Run REBUILD INDEX on an index immediately after running ALTER INDEX because the index is unusable until it is rebuilt.

Migration step 2: Run the link checker on DB2 Version 7 table spaces (optional)

The link checker utility, DSN1CHKR, verifies the integrity of the DB2 directory and catalog table spaces. DSN1CHKR scans the specified table space for broken links, hash chains, and orphans (records that are not part of any link or chain). You need to run DSN1CHKR only on catalog and directory table spaces that contain links or

hashes. You must issue the STOP DATABASE command on these table spaces before running DSN1CHKR on them. These table spaces include:
- DSNDB06.SYSDBASE
- DSNDB06.SYSDBAUT
- DSNDB06.SYSGROUP
- DSNDB06.SYSPLAN
- DSNDB06.SYSVIEWS
- DSNDB01.DBD01

In addition, run DSN1COPY with the CHECK option on all catalog table spaces to
ensure that the table space pages are physically correct and that the catalog table
spaces are clustered. When you run this utility on segmented table spaces, you
might receive message DSN1985I. The segmented table spaces in the catalog and
directory are: DSNDB06.SYSPACKAGE, DSNDB06.SYSSTR, DSNDB06.SYSSTATS,
DSNDB06.SYSDDF, DSNDB06.SYSOBJ, DSNDB01.SYSUTILX, and DSNDB01.SPT01.
You can ignore this message. See the description of DSN1985I in *DB2 Messages* for
details.

Recommendation: Run DSN1COPY and DSN1CHKR with the catalog and the directory table spaces stopped, or with DB2 stopped. Also run the CHECK INDEX utility. For more information on DSN1CHKR and DSN1COPY, see Part 3 of *DB2 Utility Guide and Reference*. For more information about CHECK INDEX, see Part 2 of *DB2 Utility Guide and Reference*.

You should run the following query on your Version 7 catalog tables to ensure that you do not have a STOGROUP that is defined with both specific and non-specific volume IDs.

If the query returns any rows, the identified STOGROUPs have both specific and non-specific volume IDS. Table spaces in databases that use these STOGROUPs cannot be image copied or recovered until ALTER STOGROUP is used to remove volumes so that the STOGROUP has either specific or non-specific volume IDs.

This query is commented out in Version 8 member DSNTESQ of *prefix*.SDSNSAMP.

General-use Programming Interface

```
SELECT * FROM SYSIBM.SYSVOLUMES V1
         WHERE VOLID ¬= '*' AND
               EXISTS (SELECT * FROM SYSIBM.SYSVOLUMES V2
                                WHERE V1.SGNAME = V2.SGNAME AND
                                      V2.VOLID = '*')
```

End of General-use Programming Interface

Migration step 3: Determine which plans and packages are invalid after migration (optional)

Migrating to Version 8 compatibility mode renders some plans and packages invalid. Find out which ones are invalid by running the following queries on your Version 7 subsystem:

| **Product-sensitive Programming Interface**

```
| SELECT DISTINCT DNAME
| FROM SYSIBM.SYSPLANDEP
| WHERE BNAME IN
```

```
('SYSCOLAUTH'   ,'SYSCOLDISTSTATS','SYSCOLDIST_HIST','SYSCOLSTATS' ,
 'SYSCOLUMNS'   ,'SYSCOLUMNS_HIST','SYSCOPY'        ,'SYSDBAUTH'   ,
 'SYSDBRM'      ,'SYSFIELDS'      ,'SYSINDEXES'     ,'SYSINDEXPART',
 'SYSINDEXSTATS','SYSPACKAGE'     ,'SYSPACKAUTH'    ,'SYSPACKDEP'  ,
 'SYSPACKSTMT'  ,'SYSPKSYSTEM'    ,'SYSPLANAUTH'    ,'SYSRESAUTH'  ,
 'SYSROUTINES'  ,'SYSROUTINEAUTH' ,'SYSSTMT'        ,'SYSSTRINGS'  ,
 'SYSTABAUTH'   ,'SYSTABLEPART'   ,'SYSTABLESPACE'  ,'SYSTABLES'   ,
 'SYSTRIGGERS'  ,'SYSUSERAUTH'    ,'SYSVIEWS'       ,'SYSVLTREE'   ,
 'SYSVTREE')
AND BCREATOR = 'SYSIBM'
AND BTYPE IN ('I','T')
ORDER BY DNAME;

SELECT DISTINCT COLLID, NAME, VERSION
FROM SYSIBM.SYSPACKDEP, SYSIBM.SYSPACKAGE
WHERE BNAME IN
('SYSCOLAUTH'   ,'SYSCOLDISTSTATS','SYSCOLDIST_HIST','SYSCOLSTATS' ,
 'SYSCOLUMNS'   ,'SYSCOLUMNS_HIST','SYSCOPY'        ,'SYSDBAUTH'   ,
 'SYSDBRM'      ,'SYSFIELDS'      ,'SYSINDEXES'     ,'SYSINDEXPART',
 'SYSINDEXSTATS','SYSPACKAGE'     ,'SYSPACKAUTH'    ,'SYSPACKDEP'  ,
 'SYSPACKSTMT'  ,'SYSPKSYSTEM'    ,'SYSPLANAUTH'    ,'SYSRESAUTH'  ,
 'SYSROUTINES'  ,'SYSROUTINEAUTH' ,'SYSSTMT'        ,'SYSSTRINGS'  ,
 'SYSTABAUTH'   ,'SYSTABLEPART'   ,'SYSTABLESPACE'  ,'SYSTABLES'   ,
 'SYSTRIGGERS'  ,'SYSUSERAUTH'    ,'SYSVIEWS'       ,'SYSVLTREE'   ,
 'SYSVTREE')
AND LOCATION = ' '
AND BQUALIFIER = 'SYSIBM'
AND BTYPE IN ('I','T')
AND COLLID = DCOLLID
AND NAME = DNAME
AND CONTOKEN = DCONTOKEN
ORDER BY COLLID, NAME, VERSION;
```

End of Product-sensitive Programming Interface

These two queries are commented out in the Version 8 member DSNTESQ of *prefix*.SDSNSAMP. You can run these queries from SPUFI or from a dynamic SQL program like DSNTEP2.

After migration, you can explicitly rebind these plans and packages or let DB2 rebind them automatically. See Part 4 of *DB2 Application Programming and SQL Guide* for suggestions on rebinding these plans and packages.

Migration step 4: Check for consistency between catalog tables (optional)

To check for consistency between catalog tables, you can run the queries that are not commented out in member DSNTESQ of the *prefix*.SDSNSAMP library.

The DSNTESQ queries check the logical correctness of the DB2 catalog. You can execute the SQL statements in DSNTESQ from SPUFI or from a dynamic SQL program like DSNTEP2.

Before you run these queries, you should have already followed step 2 in migration. In this step, you run the DSN1CHKR utility and the CHECK INDEX utility, and the DSN1COPY utility with the CHECK option.

You can run the queries on the actual catalog tables or on “mirror” copies of the catalog tables. If you run the queries on the copies, use the comment lines in member DSNTESQ for guidance. By running queries on copies of the catalog table, you reduce contention on the catalog.

Migration step 5: Take image copies of the directory and catalog: DSNTIJIC

Attention: Create a copy of your Version 7 catalog and directory for backup purposes. If you do not create this copy, DB2 starts. However, if errors in the catalog or directory require you to fall back to Version 7, you risk losing some of your tables and data.

To create a copy of the Version 7 catalog and directory, run Version 7 job DSNTIJIC. Before you run DSNTIJIC, examine the job for:

- The tape unit name. The job lists the tape unit name as TAPE. If this is incorrect for your site, correct it. The name TAPE is also used as the unit name for the default archive log data sets.
- Expiration date or retention period. You can add a retention period or an expiration date to the job.
- The USER option on the JOB statement. Ensure that the user is authorized. This must be the same user that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP.

Job DSNTIJIC contains a list of all the DB2 directory and catalog table spaces. When you run DSNTIJIC, it invokes the DB2 image copy utility to copy these table spaces to tape. The copied table spaces allow you to recover the DB2 catalog and directory in case of a failure.

If job DSNTIJIC fails or abends, look for problems with the tape that is set up for image copy. If you do not find a problem, examine the log for problems. For example, look for incorrect size or I/O errors.

After migration, periodically run the Version 8 job DSNTIJIC against the Version 8 directory and catalog, perhaps daily or weekly. This action reduces the amount of time that is required for recovering the DB2 directory or catalog. The copied data and log data sets are needed for recovery.

If you are remigrating, you need to take one of the following actions:

- Change the names of the data sets in which the new image copies are to reside. (Migration image copies use the current data set names.)
- Run the MODIFY utility to remove the migration image copies. If you select this option, ensure that you are familiar with the MODIFY utility. See *DB2 Utility Guide and Reference* for more information.

If DSNTIJIC has been modified to copy table spaces to disk instead of tape, the job is limited to two disk volumes. To change the number of disk volumes, the job needs to be modified again, using volume serial numbers instead of VOL=REF=*.*jobstep*.

Migration step 6: Connect DB2 to TSO

Access to TSO is required to support the interactive component of DB2 (DB2I) and to allow batch applications to access DB2 when those batch programs are executed under the TSO terminal monitor program (TMP).

To attach DB2 to TSO:
1. Make DB2 load modules available to TSO and batch users.
2. Make DB2 CLISTs available to TSO and batch users.
3. Make panels, messages, and load modules available to ISPF and TSO.

These tasks are described in the following topics.

Save your TSO logon procedures and JCL from Version 7 in case you need to fall back from DB2 UDB for z/OS Version 8.

Make DB2 load modules available to TSO and batch users

If you included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, you can skip this step.

If you have not included *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in your LNKLST*xx*, add JOBLIB or STEPLIB statements to your logon procedures and JCL to ensure that you access the Version 8 load modules. Include *prefix*.SDSNEXIT before *prefix*.SDSNLOAD in your JOBLIB or STEPLIB concatenations.

You can attach to multiple releases of DB2 with your existing TSO or CAF logon procedures, without changing the load libraries for your applications. After you migrate completely to the latest level of DB2, you **must** update those procedures and jobs to point to the latest level of DB2 load libraries.

Make DB2 CLISTs available to TSO and batch users: # DSNTIJVC

Tailoring changes can modify these CLISTs: DSNEMC01, DSNH, DSNU, and DSNHC. The DSNTINST CLIST reads these CLISTs from *prefix*.SDSNCLST, edits them, and places them in *prefix*.NEW.SDSNTEMP. You can modify the default values. Refer to “Completing the CLIST processing” on page 243 for information on the items that you can modify.

The DSNEMC01 CLIST uses the values that are specified on installation panel DSNTIPF and stores the results in the ISPF profile member DSNEPROF. You can migrate any customized DSNEPROF members from Version 7 to Version 8 compatibility mode. However, you need to examine any new or changed default panel values to ensure that your customized values are still valid.

Job DSNTIJVC merges the tailored CLISTs from *prefix*.NEW.SDSNTEMP with unchanged CLISTs from *prefix*.SDSNCLST, and it places all CLISTs in *prefix*.NEW.SDSNCLST. DSNTIJVC also converts the DB2 CLISTs from a fixed-block record format to a variable-blocked format, with a record length of 84 and a block size of 3120.

If you use fixed-block CLIST libraries, modify the DSNTIJVC job as follows:
- Change the SYSIN DD to DUMMY.
- Change the allocation of *prefix*.SDSNCLST to match the data control block (DCB) attributes of your other CLIST libraries.

A CLIST that has been converted from fixed-block format to variable-block format cannot be used as input to the DSNTINST CLIST; use the unedited version of the SDSNCLST data set, as created by SMP/E.

To make the CLISTs available to TSO and batch users, you must either concatenate *prefix*.NEW.SDSNCLST with your existing CLIST libraries or copy *prefix*.NEW.SDSNCLST into an existing CLIST library.

If you need to rerun this job, first delete data set *prefix*.NEW.SDSNCLST, which is created by this job.

When corrective service is applied to a CLIST, SMP/E changes only the *prefix*.SDSNCLST data set. You need to redo any record format changes and reapply any special tailoring that is required. You also need to move the CLIST to *prefix*.NEW.SDSNCLST. Corrective service (program temporary fixes) for these CLISTs is sent with ++HOLD statements, calling your attention to the possibility of additional work.

Make panels, messages, and load modules available to ISPF # and TSO

Concatenate the DB2 ISPF libraries with the ISPPLIB, ISPMLIB, and ISPSLIB DD
statements in your logon procedures and in any of your CLISTs where they might
be allocated. These libraries are *prefix*.SDSNSPFP, *prefix*.SDSNSPFM, and
prefix.SDSNSPFS. You also need to concatenate the DB2 English DB2I panels in
prefix.SDSNPFPE or, if you are using Kanji panels, in *prefix*.SDSNPFPK to ISPPLIB.

DB2I uses the ISPF PROFILE and SHARED variable pools for most panel variable fields. You can easily re-enter a panel when panel variables have previously been specified. For the DB2 subcommands that permit lists of plan names, package names, DBRMs, and ENABLE and DISABLE statements, DB2I provides ISPF tables that contain all the user-specified variables for these subcommand keywords.

DB2I creates and maintains a set of ISPF tables in a user-defined TSO data set that is allocated to a DDNAME of DSNETBLS. Table 48 on page 266 shows the library table member names and their contents.

When allocating this data set, assign the following DCB attributes, where `n` is any integer:

```
DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(n*LRECL)
```

The following example shows how you might set up an ALLOCATE statement to create the data set:

```
ALLOC DA(DSNSPFT) NEW SP(1 1) TR DIR(10) +
DSORG(PO) RECFM(F B) LRECL(80) BLKSIZE(800)
F(DSNETBLS) REUSE
```

The following example shows how you might allocate an existing data set to the DSNETBLS DDNAME:

```
ALLOC DA(DSNSPFT)      F(DSNETBLS) REUSE
```

If you do not allocate the DSNSPFT data set and connect it to ISPF, DB2I allocates a temporary data set for the ISPF table library members at DB2I startup. DB2I deletes this temporary data set when the ISPF session is terminated.

DB2I uses ISPF table services to maintain individual ISPF tables within the DSNETBLS data set. For performance reasons, ISPF keeps this table library in an **open** state after an individual table has been updated. Attempts to **close** this data set using the TSO FREE command results in error message IKJ56861I.

For additional information about this TSO error message and how to **close** this data set, refer to *z/OS ISPF Messages and Codes*.

If you want to run the ISPF-CAF sample application that is provided with DB2, ensure that the data set *prefix*.RUNLIB.LOAD is included in the STEPLIB concatenation of the logon procedure or in the ISPLLIB concatenation list. For more information about the ISPF-CAF sample application, see "Running dynamic SQL and the ISPF/CAF application" on page 384.

Refer to 326 for more information about using your TSO and CAF logon procedures.

Migration step 7: Connect IMS to DB2 (optional)

Connecting DB2 to IMS requires coordination with your IMS support group. To connect the IMS attachment facility:
- Make DB2 load modules available to IMS.
- Define DB2 to IMS.
- Define new programs and transactions to IMS.
- Prepare IMS applications for DB2.

Depending on your site, you might also need to:
- Define DB2 plans.
- Generate a user language interface.

Chapter 11, "Connecting the IMS attachment facility," on page 439 describes these tasks and the requirements from a DB2 perspective.

Migration step 8: Connect CICS to DB2 (optional)

Connecting DB2 to CICS requires that you regenerate several CICS tables with additional entries. A macro is supplied with CICS to define the connection between CICS and DB2 by using a resource control table (RCT).

Ensure that you coordinate the attachment facility connection with your CICS support group. To connect the CICS attachment facility, you must:
1. Recalculate space requirements for the CICS attachment facility.
2. Define your CICS attachment facility parameters using the RCT.
3. Update the CICS system tables.
4. Update the CICS initialization JCL.
5. Coordinate DB2 and CICS security if necessary.
6. Prepare new CICS applications for DB2 if necessary.

CICS Transaction Server for z/OS DB2 Guide describes these tasks.

Migration step 9: Stop DB2 Version 7 activity

Before making DB2 UDB for z/OS Version 8 compatibility mode operational, ensure that all work is stopped on Version 7, including data sharing members if
| you have enabled data sharing. If you do not stop work on the Version 7
| subsystem or data sharing member that you are migrating, fallback procedures might fail.

To stop work on Version 7, perform the following steps:

1. Issue the following command, where *-DSN1* is the subsystem command prefix defined for DB2:

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   The QUIESCE keyword allows DB2 to complete the processing of currently executing programs. This might require some processing time.
2. Issue the following command to allow only the system administrators and system operators that were defined during installation to access DB2:

   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```

   If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. For example, if the bootstrap data set (BSDS) does not match the subsystem parameter values, ensure that the correct jobs were run for DSNTIJUZ. Ensure that you started DB2 with the correct subsystem parameter option.
3. Ensure that all work is complete.
   - Make sure no units of recovery remain. Issue the following command:

     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```

     Then use -DSN1RECOVER INDOUBT for any indoubt threads.
   - Ensure that no utility work remains. Issue the following command:

     ```
     -DSN1 DISPLAY UTILITY(*)
     ```

     Then, either allow utilities to complete before proceeding, or issue the following command to stop all utility processing:

     ```
     -DSN1 TERM UTILITY(*)
     ```
   - Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. You can determine existing restrictions by issuing this command:

     ```
     -DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
     ```

     A user with system administrator or system operator authority specified on installation panel DSNTIPP, must enter this command.

     Recover any table spaces and index spaces with write error ranges or deferred restart states.
4. To stop DB2, issue the following command:

   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```

   A user with system administrator (SYSADM) or system operator (SYSOPR) authority must enter this command.

| All utilities must be restarted or terminated on the version on which they were
| started. If you do not use data sharing, ensure that all utilities are completed or
| terminated on Version 7. Issue the following command:

```
| -DSN1 DISPLAY UTILITY(*)
```

| After you have determined the utilities that are running, you can let them
| complete processing or you can terminate the utility. To stop all utilities, issue the
| following command:

```
| -DSN1 TERM UTILITY(*)
```

Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. You can determine existing restrictions by issuing the following commands:

```
-DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
-DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
```

You must have system administrator or system operator privileges to issue this command.

If you have table spaces or index spaces that have write error ranges or deferred restart rates, issue a RECOVER command.

Migration step 10: Back up your DB2 Version 7 volumes (optional)

At this point, you can back up your Version 7 subsystem. To do this, take dumps of the DB2 subsystem data sets. You also can take dumps of the SMP/E data sets and the DB2 distribution and target libraries.

Migration step 11: Define DB2 initialization parameters: DSNTIJUZ

DSNTIJUZ generates the DB2 data-only subsystem parameter module, DSNZP*xxx* and the data-only load modules DSNHDECP and DSNHMCID. The subsystem parameter module consists of the expansion of seven macros that contain the DB2 execution-time parameters that you selected by using the ISPF panels. The names of these macros are DSN6ARVP, DSN6ENV, DSN6FAC, DSN6GRP, DSN6LOGP, DSN6SPRM, and DSN6SYSP.

Save your Version 7 subsystem parameter module so that it is available in case you need to fall back.

The DSNTINST CLIST performs calculations by using the values that you specified for some of the parameter values that you entered on the panels. These calculations appear in the macro descriptions.

DSNTIJUZ actions

In addition to defining the subsystem parameter module, job DSNTIJUZ:

- Link-edits the DSNHDECP module into the *prefix*.SDSNEXIT data set.
- Link-edits the assembled subsystem parameter module, DSNZP*xxx*, into the *prefix*.SDSNEXIT library.
- Assembles and link-edits the DSNHMCID data-only module which is needed for message conversion by DB2 applications and utilities. DSNHMCID is link-edited into both the *prefix*.SDSNEXIT and *prefix*.SDSNLOAD libraries.
- Uses the assembler, the DSNHDECM macro, and SMP/E to update values that you specified on installation panels DSNTIPF and DSNTIP4 and your subsystem name. The information is placed into a data-only load module, DSNHDECP, which resides in *prefix*.SDSNEXIT.
- Uses SMP/E in step DSNTIMQ to read in the edited version of DSNTIJUZ. This action is required to determine the appropriate includes and library names. After the initial run of step DSNTIMQ, rerunning this step is required only when changes have been made to DSNHDECP.
- Uses JCLIN to ensure that macro maintenance is placed into all the required load modules.

If DSNHMCID cannot exist in *prefix*.SDSNLOAD, take one of the following actions:

- Include *prefix*.SDSNEXIT ahead of *prefix*.SDSNLOAD in the system link list.
- Include *prefix*.SDSNEXIT ahead of *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statements for all DB2 applications, address space start-up procedures, TSO log-on procedures, CICS tasks, and IMS tasks that use DB2 .

Additional steps for DSNTIJUZ are as follows:

1. If you added a STEPLIB statement to the DB2 start procedures ahead of *prefix*.SDSNEXIT and *prefix*.SDSNLOAD, you can move the SYSLMOD output to that library.
2. If you changed the prefix for the DB2 distribution libraries, edit DSNTIJUZ to correct the data set names.
3. If you have not run the SMP/E ACCEPT job (DSNTIJAC) of FMID HDB8810, edit DSNTIJUZ so that the SMP/E temporary data set (SMPTLIB) is included in the concatenation for the ADSNLOAD DD statement in step DSNTIZQ. This action ensures that member DSNARIB is linked with DSNHDECP. The linkage editor issues a return code of 8, along with message IEW0342 for the following CSECTs:

DSNFSYSP	DSNJARVP	DSNJLOGP	DSNTSPRM
DSNVDIR1	DSNZMSTR	DSN3DIR1	

Subsystem parameter UNION_COLNAME_7, in macro DSN6SPRM, specifies what behavior for DB2 to use for result column names in UNION queries. The default value of NO results in Version 8 behavior. If the column name is the same across all sub-queries in the UNION, the result column name will be that column name. Otherwise, the result column is unnamed. When the value of UNION_COLNAME_7 is YES, DB2 uses Version 7 and earlier behavior. The column name that is returned in the SQLNAME field of an SQLDA following a DESCRIBE statement where the result table that is being described is the result of a union is the column name from the first subquery of the union operation.

Add a continuation character in column 72 if needed. To assemble and link the load module, run the first two steps of DSNTIJUZ. After DSNTIJUZ completes, you must either use the SET SYSPARM command or stop and start DB2 for the change to take effect.

Subsystem parameter COMCRIT, in macro DSN6SPRM, sets the Common Criteria environment. The default value is NO, which means the behavior of DB2 is unchanged. When the value of COMCRIT is YES, all tables that you create (other than created global temporary tables, declared global temporary tables, and auxiliary tables) must have multilevel security. If the AS SECURITY LABEL clause is missing from a table, an error occurs and the table is not created. Setting COMCRIT to YES will cause some of the current installation and migration processes to fail. You can change the value of COMCRIT online by using SET SYSPARM and you can audit COMCRIT with IFCID 0106.

Add a second BSDS

If your Version 7 system has only one BSDS, take one of the following actions:
- Manually add TWOBSDS=NO in the DSN6LOGP macro in job DSNTIJUZ.
- Add another BSDS to DB2 before you migrate.

Recommendation: Add a second BSDS because having two BSDSs makes recovery much easier in most situations. In cases that normally require recovery and restart, a second BSDS allows you to continue working. Also, the required storage is small, and the data set is relatively inactive.

To add a second BSDS:

1. Change your subsystem parameter to TWOBSDS=YES by using job DSNTIJUZ.
2. Define a second BSDS; use the VSAM BSDS definition in job DSNTIJIN as an example.
3. Add a //BSDS2 DD statement to the DSN1MSTR DB2 startup procedure.
4. Execute the RECOVER BSDS command to establish dual BSDS. For information on the RECOVER BSDS command, see Part 4 (Volume 1) of *DB2 Administration Guide*.

You might receive message GIM65001W when running steps DSNTLOG and DSNTIMQ, or you might receive a return code of 4 when running step DSNTIMQ. You can ignore these messages.

If DSNTIJUZ fails or abends, correct the problem and rerun the job, using the same subsystem parameter name.

For more information about access method services, see z/OS DFSMS Access Method Services for Catalogs.

Migration step 12: Establish subsystem security (optional)

DB2 includes means for controlling access to data within DB2. It also works together with outside security systems, such as RACF, that control access to the DB2 subsystem. See Part 3 (Volume 1) of *DB2 Administration Guide* for suggestions and instructions for including DB2 in your security system.

Because your Version 8 system reuses the data objects from your Version 7 system, you have probably already supplied the protection that those objects need. However, you probably want to protect the new (Version 8) DB2 data objects.

Migration step 13: Define DB2 Version 8 to z/OS: DSNTIJMV

This job does some of the steps that are required to identify DB2 to z/OS, including updating members of SYS1.PARMLIB and SYS1.PROCLIB. This job renames your Version 7 procedures so that they do not conflict with Version 8 procedures.

Because different sites have different requirements for identifying DB2 to z/OS, DSNTIJMV cannot anticipate all the necessary updates. For this reason, the updates that job DSNTIJMV makes in SYS1.PARMLIB and SYS1.PROCLIB are incomplete. You might have additional procedures of your own to rename, or you might have to provide procedures for both releases, using alias names to indicate the current release. You can complete these updates either by making the updates directly in SYS1.PARMLIB and SYS1.PROCLIB or by editing DSNTIJMV.

Recommendation: For SYS1.PROCLIB, submit the procedure-update section of DSNTIJMV, as necessary. However, before you make the updates, read this section of the chapter, and examine DSNTIJMV to study the updates it makes. Edit the updates directly in SYS1.PARMLIB instead of submitting the updates in the DSNTIJMV step.

Regardless of whether you make the updates directly or edit DSNTIJMV to make the updates, you might first want to review "Choosing link list options" on page 53.

DSNTIJMV actions

Job DSNTIJMV does the following updates to SYS1.PARMLIB and SYS1.PROCLIB
to help identify DB2 to z/OS:

1. Job DSNTIJMV updates the following SYS1.PARMLIB members:
   - IEFSSN*xx*

     This member contains an entry for every z/OS subsystem. Unless you change the DB2 subsystem name or the DB2 command prefix, you do not need to change this member. If you change the subsystem name or the command prefix, either change the current member or create a new member.

     Place the primary system's record (JES2 or JES3) record as the first line in an IEFSSN*xx* member. However, if you use SMS, place the SMS line before the primary system. There might also be other products that change position during system initialization. The DB2 line should come after SMS, the JES subsystem, and other vendor products.
   - IEAAPF*xx* or PROG*xx*

     Job DSNTIJMV updates IEAAPF*xx* to include the DB2 program libraries (*prefix*.SDSNEXIT,*prefix*.SDSNLOAD, *prefix*.SDXRRESL, and *prefix*.SDSNLINK) as libraries that are authorized by using the authorized program facility (APF).

     All libraries that are concatenated with *prefix*.SDSNEXIT and *prefix*.SDSNLOAD in STEPLIB and JOBLIB statements must be APF-authorized.
   - LNKLST*xx*

     Job DSNTIJMV updates this member to include the DB2 load module library, *prefix*.SDSNLINK, in the LNKLST*xx*. If you moved the modules from *prefix*.SDSNLINK into another library, edit DSNTIJMV to include that library in the LNKLST*xx*. If you have combined *prefix*.SDSNLINK and *prefix*.SDSNLOAD into one library, edit DSNTIJMV to include the combined library in the LNKLST*xx*. See *z/OS MVS Initialization and Tuning Guide* for restrictions on data sets that are concatenated in LNKLST.

   You can do additional editing for the SYS1.PARMLIB updates. If you are editing DSNTIJMV rather than making the changes directly, you have a choice: You can either include your additional entries for the SYS1.PARMLIB members (IEAAPF*xx* and LNKLST*xx*) at the end of the existing list of entries, or you can place them earlier in the list.

   If you include these entries at the end of the existing SYS1.PARMLIB list, ensure that commas (the continuation character) delimit each entry except the last entry.

   ECSA size is another SYS1.PARMLIB change to consider at this time. You specify ECSA size in the CSA parameter of the IEASYS00 parameter. Ensure that you have specified an adequate size for this subsystem (generally 2 MB plus the MAXIMUM ECSA value on installation panel DSNTIPJ if the CROSS MEMORY value is NO).

   The **IOP parameter** is another SYS1.PARMLIB change to consider at this time. DB2 can schedule synchronous read-write I/Os and prefetch read I/Os under the application address space's I/O scheduling priority. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about the effects of this type of I/O scheduling. To enable this type of I/O scheduling:

- Use the IOP parameter to set the I/O priority for the address space of a performance group. The IOP parameter is in the IEAIPSxx member of SYS1.PARMLIB.
- Enable z/OS I/O priority scheduling by specifying IOQ=PRTY in the IEAIPSxx member of SYS1.PARMLIB.

2. Job DSNTIJMV renames your Version 7 procedures so that they are not replaced by DB2 UDB for z/OS Version 8 procedures.
3. DSNTIJMV updates SYS1.PROCLIB to include the following Version 8 procedures:
   - System services address space startup procedure (*xxxx*MSTR)
   - Database services address space startup procedure (*xxxx*DBM1)
   - Distributed data facility address space startup procedure (*xxxx*DIST)
   - Stored procedures address space startup procedure (*xxxx*SPAS)
   - WLM environment for the DSNACICS stored procedure (*xxxx*CICS)
   - WLM sample procedure for stored procedures
   - IRLM address space startup procedure (IRLMPROC)
   - Program preparation procedures
   - Utilities procedure (DSNUPROC)

   If you specified a suffix on panel DSNTIPA1, that suffix is appended to data sets &USER..DBRMLIB.DATA.*suffix*, &USER..RUNLIB.LOAD.*suffix*, and &USER..SRCLIB.DATA.*suffix*. To override these data set names, edit the updates to SYS1.PROCLIB.

   The STEPLIB concatenation of the *xxxx*DBM1 address space procedure includes a commented-out DD for the IBM Language Environment runtime library (SCEERUN). If your system does not include the SCEERUN library in the systemlinklist, you must uncomment this DD.

Important: In Version 8, support for DB2-established address spaces is deprecated. Although existing stored procedures can still run in a DB2-established stored procedure address space, you should move your stored procedures to WLM environments as soon as possible. If you CREATE or ALTER an existing stored procedure, it cannot run in a DB2-established stored procedure address space. For more information about moving stored procedures, see Part 5 (Volume 2) of of *DB2 Administration Guide*.

Completing the step

During migration, DB2 UDB for z/OS Version 8 procedures replace your Version 7 procedures (which are renamed). If you changed the DB2 subsystem name, the name of the DB2 address space startup procedures also change. If you made any changes to your Version 7 procedures (such as data set names), make similar changes to the Version 8 procedures.

Before starting DB2, check the private area sizes in the SYS1.PROCLIB update section to ensure that you have enough user private area.

Also, examine the size of the private area on the DB2 startup procedures. If necessary, modify them to satisfy the requirements for EDM pool size, buffers, numbers of open data sets, and the amount of available private address space. For more information about private address spaces, see "Working storage calculation" on page 42.

If job DSNTIJMV runs successfully, it produces return codes of 0. Because a rename can fail without setting the return code, verify all renames.

Migration step 14: Define system data sets: DSNTIJIN

Job DSNTIJIN defines these non-VSAM data sets:
prefix.SRCLIB.DATA
prefix.RUNLIB.LOAD
prefix.DBRMLIB.DATA

The job also defines the new VSAM data sets for the table spaces and indexes of the catalog and directory.

Recommendation: For recovery purposes, keep system data sets on different disk volumes. Because these data sets are in use frequently, do not migrate them with DFSMShsm.

If DSNTIJIN runs successfully, it produces return codes of 0 for all steps. Check any VSAM messages carefully.

If job DSNTIJIN fails or abends, delete the allocated non-VSAM data sets, examine the VSAM messages, correct any indicated problems with DFSMSdfp, and then rerun job DSNTIJIN.

Migration step 15: Define user authorization exit routines: DSNTIJEX

Job DSNTIJEX builds the sample authorization exit routines, DSN3@SGN and DSN3@ATH and the user version of the access control authorization exit routine DSNX@XAC, from the source code in *prefix*.SDSNSAMP and places them in the *prefix*.SDSNEXIT library. Job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. You can modify DSNX@XAC, the access control authorization exit routine, and use DSNTIJEX to assemble and link-edit it. This exit routine allows you to bypass some or most of DB2 authorization checking and to specify your own authorization checking. For more information about this access control authorization exit routine, see Appendix B of *DB2 Administration Guide*. The DB2 CLIST tailors the JCL in DSNTIJEX to meet the requirements of your site.

The sample authorization exit routines are not the same as the default authorization exit routines that are supplied by DB2. By implementing the sample authorization exit routines, you can provide group names as secondary authorization IDs. For information about writing exit routines, see Appendix B of *DB2 Administration Guide*. For more information on controlling data access, see Part 3 (Volume 1) of *DB2 Administration Guide*.

You have the following options regarding exit routines:
- To use the default authorizations, skip job DSNTIJEX.
- To use the sample authorization exit routines, run job DSNTIJEX.
- To use your own authorization exit routines, modify job DSNTIJEX to reference the correct library, and then run it.

If job DSNTIJEX runs successfully, it produces a return code of 0 or 4.

If job DSNTIJEX fails or abends, correct the problem, and rerun the job.

DSNXSXAC is a copy of the default access control authorization exit routine that you can modify. This exit routine allows you to bypass some or most of DB2 authorization checking and to specify your own authorization checking. If you do not change the exit routine, you should delete this step.

DSNACICS is a stored procedure that invokes user exit routine DSNACICX, which you can use to modify CICS parameters that the DSNACICS caller specifies. If you do not need to modify the caller's parameter values, you can use the default DSNACICX exit routine. However, if you need to modify the caller's parameter values, you need to perform the following tasks:

1. Write a user exit routine in assembler, COBOL, C, or PL/I
2. Assemble or compile the source code
3. Link-edit the object code into the DB2 exit routine library

Installation job DSNTIJEX includes a step to assemble and link-edit the sample version of DSNACICX. You can use this step as a model for your program preparation job.

| If you will use the RACF/DB2 external security module (DSNXRXAC) as your
| DB2 access control authorization exit, modify job DSNTIJEX to refer to
| DSNXRXAC instead of DSNXSXAC.

Migration step 16: IPL z/OS

The load module library SDSNLINK contains the early code. If all of the required maintenance has been applied to your system, the early code is **upward** compatible with DB2 UDB for z/OS Version 8. Ensure that the early code pre-conditioning PTFs have been installed on your system before you migrate. The Version 8 early code is **downward** compatible with Version 7.

If you are at the appropriate service level for Version 7, you can plan ahead, do PARMLIB updates (which are necessary at least to update the APF authorization list), and IPL z/OS whenever convenient, before you begin your migration. The z/OS IPL is necessary because migration job DSNTIJMV makes changes to SYS1.PARMLIB that are not recognized by z/OS until the next IPL. The DSNTIJMV job makes the following changes to the SYS1.PARMLIB library:

- Creates new subsystem definitions in the IEFSSN*xx* member
- Creates new APF libraries in the IEAAPF*xx* member
- Creates new load module libraries in the LNKLST*xx* member

Complete these changes before performing the IPL.

You must IPL before or during migration, but IPLs are not necessary for fallback or remigration. For more information, refer to "Migration step 13: Define DB2 Version 8 to z/OS: DSNTIJMV" on page 332 and "Choosing link list options" on page 53.

After you IPL z/OS, message DSN3100I appears on the z/OS console, stating that DB2 is ready for the START command.

Migration step 17: Start DB2 Version 8

Perform the following steps to start DB2 UDB for z/OS, Version 8:

1. Start the IRLM.

   If you have not requested that DB2 automatically start the IRLM, you should start it before you start DB2. Use the following command, where *irlmproc* is the name that you assigned to the IRLM startup procedure:

   ```
   START irlmproc
   ```

The *irlmproc* is the value that you specified for the PROC NAME option on installation panel DSNTIPI.

If you specified YES for the AUTO START option on installation panel DSNTIPI, DB2 starts the IRLM automatically.

2. Start DB2 from the z/OS console with the following command, where -DSN1 is the subsystem command prefix that you defined for DB2, and *DSNZPxxx* is the name of the DB2 initialization parameter module:

   ```
   -DSN1 START DB2 PARM(DSNZPxxx)
   ```

   If you omit the PARM parameter, the name that is used is the one that you specified in the field PARAMETER MODULE on panel DSNTIPO. If you did not specify a parameter module on panel DSNTIPO, DB2 uses the default, DSNZPARM.

   If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, and possibly *ssnm*DIST, *ssnm*SPAS, and *irlmproc*, where *ssnm* is the DB2 subsystem name and *irlmproc* is the IRLM procedure name.

   If DB2 starts successfully, the series of RESTART messages that you receive concludes with these two messages:

   ```
   DSNR002I   RESTART COMPLETED
   DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
   ```

   In the next step, you migrate the DB2 catalog. Before the catalog is migrated, some catalog or directory table spaces are restricted. You can ignore the following messages that might occur during startup because the catalog and directory table spaces are restricted:

   - DSNT501I with reason code 00C900A6
   - DSNL700I with reason code 00C900A6 (if DDF is auto-started)
   - Abend 04E with reason code 00E70014 (during DDL registration)

   You can determine existing restrictions by issuing this command after you start DB2:

   ```
   -DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
   ```

   The preceding command might also generate message DSNT501I with reason code 00C900A6.

   If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. A common cause of startup failure is that the BSDS does not match the subsystem parameter values; ensure that the correct job was run for DSNTIJUZ. Also, check that the subsystem parameter member that you specified (or used by default) when you started DB2 is the one that the DSNTIJUZ job built. Check the JCL for the DB2 startup procedure.

3. Optionally, start TSO. If you want to use the TSO SUBMIT command to do housekeeping and migration verification, start TSO (if it is not already started).

4. Take one of the following actions to enable primary user IDs to issue DB2 commands from the z/OS console or TSO SDSF:

   - Grant SYSOPR authority to all primary user IDs that issue DB2 commands from the z/OS console or TSO SDSF. Issue the following command:

     ```
     GRANT SYSOPR TO userid
     ```

   - Define RACF classes to authorize DB2 commands. Use the following statements:

```
SETR CLASSACT(DSNADM)
RDEFINE DSNADM DSN1.SYSOPR UACC(NONE)
SETR RACLIST(DSNADM) REFRESH
PERMIT DSN1.SYSOPR CLASS(DSNADM) ID(userid) ACCESS(READ)
SETR RACLIST(DSNADM) REFRESH
```

You must run job DSNTIJTC to complete the tailoring of the DB2 catalog. In a data sharing environment, do not run DSNTIJTC after installing non-originating members. When you start DB2 Version 8 for the first time, you might receive message DSNT501I with reason code 00C900A6. You can ignore this message. Running job DSNTIJTC corrects the cause of this message.

Migration step 18: Tailor DB2 Version 8 catalog: DSNTIJTC

DSNTIJTC invokes the CATMAINT utility to migrate your Version 7 catalog to the Version 8 catalog. Before running this job, ensure that you have enough disk space, as described in “Work file database storage requirements” on page 22. If you do not have enough disk space, the job fails. See Table 54 on page 315 for a list of new and changed indexes that might affect your work file database space needs. Complete all premigration activities before running this job.

DSNTIJTC contains one step. DSNTIJTC creates new catalog and directory objects, adds columns to existing catalog tables, and creates and updates indexes on the catalog tables to accommodate new Version 8 objects. All IBM-supplied indexes are created or updated sequentially during the execution of DSNTIJTC.

A status message, DSNU777I, is issued at several points to indicate migration progress. Diagnostic error messages are issued when CATMAINT processing fails. If a problem is found during the SQL processing phase of migration, message DSNU778I is issued. If non-supported functions are encountered, message DSNU776I is issued. All of these messages are written to the SYSPRINT data set. See the message descriptions in *DB2 Messages* for details.

If job DSNTIJTC fails, save the output and verify that you are at the correct maintenance level. If you are not, you need to install the appropriate maintenance. If you are at the correct maintenance level, correct the problem, and rerun the job. If you run the job again and the job still fails, return to Version 7. Because CATMAINT failures roll back all Version 8 changes, the catalog and directory are in Version 7 format. Altered indexes are not rolled back. Determine if any index is in a pending status by using the CHECK INDEX utility. To return to Version 7 you need to:
- Rename procedures to use Version 7 libraries.
- Reconnect TSO, IMS, and CICS to Version 7 libraries.

See “Falling back” on page 344 for details on these steps.

If your Version 7 system is damaged, you need to perform a point-in-time recovery. Follow these step to recover your Version 7 system:
- Restore the Version 7 catalog and directory from image copies.
- Restore the BSDSs from archive logs that were made prior to migration.
- Flush the SCA (for data sharing environments only).
- Recover the catalog and directory indexes.

For more details about recovering data to a prior point of consistency, see Part 4 (Volume 1) of *DB2 Administration Guide*.

When you remigrate, run the DSNTIJTC job again. To execute DSNTIJTC, you must have installation SYSADM authority.

Migration step 19: Ensure that the catalog has no problems (optional)

Check the integrity of your DB2 UDB for z/OS Version 8 catalog and directory by following, in any order, these steps:

- Run CHECK INDEX on all the indexes in the catalog and directory by using job DSNTIJCX. In Version 8, DSNTIJCX will indicate that there are no indexes in the SYSALTER tablespace because these objects are not created until you commence enabling-new-function mode.
- Run the link checker (DSN1CHKR) to ensure that no existing links are broken. See "Migration step 2: Run the link checker on DB2 Version 7 table spaces (optional)" on page 322 for more information.
- Run the queries in member DSNTESQ of *prefix*.SDSNSAMP. Because SPUFI is not bound yet, you cannot use SPUFI to run these queries. One alternative is to use the Version 7 DSNTEP2 program to run the queries.
- Run the DSN1COPY utility with the CHECK option on the catalog table spaces.

Migration step 20: Ensure that routines use Version 8 DBINFO

If you defined an external function or procedure with DBINFO, you must update the routine body for the declaration of the CCSIDs in DBINFO. In Version 7, CCSID fields in DBINFO were set to the CCSIDs of the invoking statement. In Version 8, the position of the Unicode CCSID fields have been moved, and the CCSID fields in DBINFO are now set to the CCSIDs corresponding to the application encoding scheme in effect. This set of three CCSIDs might not reflect the CCSIDs of a statement that invokes the routine. See *DB2 Application Programming and SQL Guide* for more details about DBINFO.

If you have a routine that references the ASCII or Unicode CCSID fields in DBINFO, you must recompile and rebind it in Version 8 to correctly reference these fields. Additionally, you must recompile and rebind applications that invoke these routines at the same level as the routine. To identify routines that must be recompiled and rebound because they reference the ASCII or Unicode CCSID fields in DBINFO, issue the following query:

```
SELECT SCHEMA, NAME
FROM SYSIBM.SYSROUTINES
WHERE DBINFO = 'Y';
```

If the routine is written in C and includes the SQLUDF.H file, you can recompile and rebind it. If the routine does not include SQLUDF.H, or if it is written in another language, you must update the application code to meet the new structure of the CCSID information in DBINFO. See *DB2 Application Programming and SQL Guide* for more information about the structure of DBINFO. After you have updated the application code, recompile and rebind these routines. After you have recompiled and rebound the routines, you must recompile and rebind the applications that invoke these routines.

Exception: If your routine references only the EBCDIC CCSID field in DBINFO, you do not need to update, recompile, or rebind.

Migration step 21: Enable change data capture

During migration to Version 8 compatibility mode, DB2 disables change data capture on the following catalog tables:

SYSCOLDIST_HIST	SYSCHECKS	SYSCOLAUTH	SYSCOLUMNS_HIST	SYSCOLUMNS

SYSCOPY	SYSCOLSTATS	SYSDBAUTH	SYSDBRM	SYSCOLDIST
SYSCOLDISTSTATS	SYSCOLAUTH	SYSFIELDS	SYSINDEXPART	SYSINDEXES
SYSINDEXSTATS	SYSVLTREE	SYSPACKAUTH	SYSPACKDEP	SYSPACKAGE
SYSPACKSTMT	SYSPKSYSTEM	SYSPLANAUTH	SYSRESAUTH	SYSROUTINEAUTH
SYSROUTINES	SYSSTRINGS	SYSSTMT	SYSTABAUTH	SYSTABLES
SYSTABLEPART	SYSTRIGGERS	SYSUSERAUTH	SYSVIEWS	SYSVTREE

If you want to use change data capture on these tables, you must re-enable it. To re-enable change data capture, issue the following command:

```
ALTER TABLE SYSIBM.SYSTABLES DATA CAPTURE CHANGES;
```

Migration step 22: Prepare dynamic SQL program: DSNTIJTM

DSNTIJTM assembles, link-edits, binds, and runs DSNTIAD, a program that processes certain SQL statements dynamically. This job also assembles, link-edits, and binds DSNTWR, the load module for the DB2-supplied stored procedure WLM_REFRESH, which is created in DSNTIJSG.

Migration step 23: Bind SPUFI and DCLGEN and user-maintained database activity: DSNTIJSG

In migration mode, job DSNTIJSG:

- Rebinds the IBM-defined packages
- Redefines the DB2-supplied stored procedures

If you bound special SPUFI packages and plans in Version 7, you need to bind those packages again in Version 8. You do not need to bind the plan again. To bind the special SPUFI packages again, issue the following commands:

```
BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
```

If you did not bind special SPUFI packages and plans in Version 7, you can bind new ones now with the following commands:

```
BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('DSN!!0.SDSNDBRM')
BIND PLAN(SPCS1047) -
     PKLIST(SPCS1047.DSNESM68) -
     ISOLATION(CS) ENCODING(1047) ACTION(REPLACE)
BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
     LIBRARY('DSN!!0.SDSNDBRM')
BIND PLAN(SPRR1047) -
     PKLIST(SPRR1047.DSNESM68) -
     ISOLATION(RR) ENCODING(1047) ACTION(REPLACE)
```

For more information, see “Special packages and plans for SPUFI” on page 300.

See Appendix B of *DB2 Utility Guide and Reference* for information about invoking utilities from a stored procedure. Stored procedures can be enabled after installation or migration. See “Enabling stored procedures after installation” on page 285 for the specific steps that are required to accomplish this.

Requirements:

- Stored procedure DSNWZP now requires a WLM-managed stored procedures address space.
- In a data sharing environment, you must ensure that the resource limit facility (RLF) is inactive on all members in the data sharing group before running DSNTIJSG. To do this, issue the STOP RLIMIT command to each member.

Important: You must set NUMTCB=1 in your WLM environment for DSNWZP.

If DSNTIJSG runs successfully, it produces a return code of 0. It can also produce a return code of 4 because a step within this job attempts to delete a row from a table that might not exist at the time this job runs. DB2 issues is a bind warning for each plan. Expect the following messages:

```
DSNE932I  WARNING, ONLY IBM-SUPPLIED PLAN NAMES SHOULD BEGIN WITH DSN
DSNE932I  WARNING, ONLY IBM-SUPPLIED PACKAGE-IDS SHOULD BEGIN WITH DSN
DSNE932I  WARNING, ONLY IBM-SUPPLIED COLLECTION-IDS SHOULD BEGIN WITH DSN
```

When tailored for migration, DSNTIJSG binds the packages for use by the DB2 ODBC and JDBC metadata methods in Version 8 compatibility mode. DSNTIJSG does not create the database and global temporary tables needed by the metadata methods because these are presumed to have been migrated from DB2 Version 7. If these objects do not exist in your DB2 subsystem, or you otherwise do not use the DB2 ODBC and JDBC metadata methods, remove all BIND PACKAGE(DSNASPCC) statements from job step DSNTIRU before running this job.

When you migrate the first member of the data sharing group to Version 8, DSNTIJSG rebinds SPUFI in Version 8. The Version 7 members cannot use a Version 8 SPUFI. If you attempt to run an SQL statement in a data sharing member at Version 7 with SPUFI at Version 8, expect the following messages:

```
DSNT408I SQLCODE = -904, ERROR:  UNSUCCESSFUL EXECUTION
      CAUSED BY AN UNAVAILABLE RESOURCE. REASON 00E7009E
DSNT418I SQLSTATE = 57011 SQLSTATE RETURN CODE
```

If job DSNTIJSG fails or abends, ensure that the user that is specified on the JOB statement is authorized. Use the name that you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP. (The RESTART parameter on the JOB statement can be useful.)

Correct any other problems, and rerun DSNTIJSG. If you encounter resource shortages, review the parameters in job DSNTIJUZ, making any necessary modifications. Then stop DB2, rerun DSNTIJUZ, start DB2, and rerun DSNTIJSG from the last successful step.

Recommendation: Alter your DB2 Version 8 buffer pools that have frequent page reads or frequent page writes to use PGFIX YES if you have sufficient real storage available for the these buffer pools. Fixing the buffer pages in real storage once and keeping them fixed avoids the processing time that DB2 needs to fix and free pages each time there is an I/O. In some cases, this processing time can be as much as 10% for I/O intensive workloads. To use this option, issue the following command:

```
ALTER BPOOL(bpname) VPSIZE(vpsize) PGFIX(YES)
```

Where *bpname* is the name of the buffer pool and *vpsize* is the size of the virtual pool.

Migration step 24: Bind the packages for DB2 REXX Language Support: DSNTIJRX (optional)

This step is required if you plan to use DB2 REXX Language Support. Before you can use DB2 REXX Language Support, you must bind DB2 packages that DB2 REXX Language Support uses. Run job DSNTIJRX to do this. Before you run DSNTIJRX, make the following changes:

- Add a job statement.
- Change `DSN SYSTEM(DSN)` to `DSN SYSTEM(`*`ssid`*`)`, where *`ssid`* is the name of the DB2 subsystem on which you plan to use DB2 REXX Language Support.
- Change all instances of DSN!!0 to your DB2 data set name prefix.
- Change all instances of DSNTIA!! to the plan name for the DSNTIAD program.

Migration step 25: Verify views

During migration to Version 8 compatibility mode, some views might be marked with view regeneration errors. To determine which views were marked with view regeneration errors during migration, issue the following query:

```
SELECT CREATOR,NAME
FROM SYSIBM.SYSTABLES
WHERE TYPE = 'V'
AND STATUS = 'R'
AND TABLESTATUS = 'V'
```

If any views have view regeneration errors, issue the following query:

```
ALTER VIEW view REGENERATE
```

where *view* is the name of the view with regeneration errors.

Migration step 26: Take an image copy of the DB2 Version 8 compatibility mode catalog: DSNTIJIC

Create a copy of the DB2 UDB for z/OS Version 8 compatibility mode catalog and directory for back-up purposes. See “Migration step 5: Take image copies of the directory and catalog: DSNTIJIC” on page 325 for information on job DSNTIJIC.

Migration step 27: Verify your DB2 Version 8 compatibility mode system (optional)

Three steps verify your Version 8 subsystem. The first step, which is optional, runs selected Version 7 sample jobs. If the DB2 objects from the Version 7 sample jobs exist, you can follow the procedure under “Run the DB2 Version 7 sample jobs” on page 343. If you deleted the Version 7 sample objects, continue with step 2. The final step is testing your application test cases.

In Version 8, you cannot run the IVP jobs until DB2 is running in Version 8 new-function mode. Therefore, you should run the Version 7 IVP jobs to verify a successful migration to Version 8 compatibility mode. The Version 8 IVP jobs are created by the installation CLIST as part of Version 8 enabling-new-function mode, and the Version 8 IVP jobs should not be run until the DB2 subsystem is running in Version 8 new-function mode.

#

Run the DB2 Version 7 sample jobs

If all of the local DB2 objects from Version 7 still exist (that is, if you have not run job DSNTEJ0), follow these steps:

1. Change the JOBLIB statements to point to *prefix*.SDSNLOAD.
2. Ensure that the DSN8EAE1 module that you created when you originally ran the Version 7 sample jobs is copied to *prefix*.SDSNEXIT. DSN8EAE1 is an EDITPROC that is used by the employee sample table.
3. Edit the Version 7 sample jobs before executing them. Do **not** run all the Version 7 sample jobs. Run only the specific jobs and job steps that are listed in step 4.
4. Test the migration of the IVP phase 2 applications from Version 7:
   a. DSNTEJ2A: Perform all except the first two steps of job DSNTEJ2A.

      Expect a return code of 4 because table spaces DSN8D710.NEWDEPT and DSN8D710.NEWPHONE are placed in COPY-pending states.
   b. DSNTEJ2C: Execute only the RUN PROGRAM(DSN8BC3) PLAN(DSN8BH71) statement in step PH02CS04.
   c. DSNTEJ2D: Execute only the RUN PROGRAM(DSN8BD3) PLAN(DSN8BD71) statement in step PH02DS03.
   d. DSNTEJ2E: Execute only the RUN PROGRAM(DSN8BE3) PLAN(DSN8BE71) statement in step PH02ES04.
   e. DSNTEJ2F: Execute only the RUN PROGRAM(DSN8BF3) PLAN(DSN8BF71) statement in step PH02FS03.
   f. DSNTEJ2P: Run only step PH02PS05.
5. Test the migration of the IVP phase 3 applications from Version 7:
   a. Do not run job DSNTEJ3C or DSNTEJ3P.
   b. If you want to test the DB2 Version 7 ISPF-CAF applications under Version 8, place the Version 7 SDSNSPFP panel library ahead of the Version 8 SDSNSPFP panel library in the ISPPLIB concatenation. This placement is necessary so that the plans that are migrated from Version 7 can be used. Remove the Version 7 SDSNSPFP library from your ISPPLIB concatenation when you are finished testing the Version 7 IVP applications under Version 8.

      See the Version 7 *DB2 Installation Guide* for directions about how to start and run the Version 7 ISPF-CAF applications.

Do not run any other Version 7 sample jobs.

#

Run the Version 8 sample jobs

|
|
In Version 8, you cannot run the IVP jobs until DB2 is running in Version 8 new-function mode.

The detailed instructions for running your Version 8 sample jobs on your Version 8 subsystem are described in Chapter 10, "Verifying installation with the sample applications," on page 363.

#

Test Version 8 with your application test cases

After completing the DB2 Version 8 sample jobs, you should test your applications on the Version 8 subsystem. For information about testing application programs, see *DB2 Application Programming and SQL Guide*.

Migration step 28: Enable stored procedures (optional)

To enable stored procedures **after** you have completed the migration process, you can either run the installation CLIST in MIGRATE mode or edit the job DSNTIJUZ.

More information about enabling stored procedures is available in “Enabling stored procedures after installation” on page 285.

Falling back

Falling back is the process of returning to DB2 Version 7 after migrating your catalog and directory to DB2 UDB for z/OS Version 8 compatibility mode. You can fall back to Version 7 only after successfully migrating the catalog to Version 8 compatibility mode by using job DSNTIJTC. However, you cannot fall back to Version 7 or return to Version 8 compatibility mode after you enter enabling-new-function or new-function mode.

Fall back if you have a severe error while operating Version 8 compatibility mode and you want to return to operation on Version 7. After fallback, the catalog remains a Version 8 catalog.

Remigrating is the process of returning to Version 8 compatibility mode after falling back to Version 7.

Fallback considerations

You cannot use the new DB2 UDB for z/OS Version 8 facilities until you are in new-function mode.

Data sharing

There are additional considerations for falling back if any member of a data sharing group falls back. See *DB2 Data Sharing: Planning and Administration* for these details.

Frozen objects

Falling back does not undo changes that the migration process made to the catalog. DB2 uses the migrated catalog after fallback. Some objects in this catalog that have been affected by Version 8 function might become *frozen* objects after fallback. Frozen objects are unavailable, and they are marked with the release dependency marker L. If an object is marked with a release dependency, it remains marked forever. The release dependency marker is listed in the IBMREQD column of catalog tables. Table 55 lists the objects that are frozen when falling back to Version 7.

Table 55. Objects that are frozen when falling back to Version 7

RELEASE DEPENDENT MARK = L
• Plans, packages, or views that use any new syntax, objects, or bind options • DBRMs that are produced by a precompile in Version 8 with a value of YES for the NEWFUN option • User-defined functions created in Version 8 with the PARAMETER CCSID option • User-defined SQL procedures and functions created in Version 8 with the PARAMETER CCSID option

Plans and packages become frozen objects when they use new SQL syntax, use new BIND options and attributes, or reference frozen objects. When plans and packages become frozen objects, the automatic rebind process is adversely affected.

General-use Programming Interface

While operating in Version 7, you can determine if any of your objects are frozen by issuing some SELECT statements. Job DSNTESQ contains these statements.

End of General-use Programming Interface

Automatic rebind

After fallback, plans or packages that are bound in Version 8 are automatically rebound on their first execution in Version 7. See *DB2 Application Programming and SQL Guide* for more details about automatic rebind.

After fallback, if you try to use plans or packages that are frozen, the automatic rebind in Version 7 that takes place the first time that you try to run the plan in Version 7 fails. To make the plans and packages that were not automatically rebound on Version 7 available, change the SQL statements or remove the reference to a frozen object, precompile the application programs, and explicitly bind the plans and packages on Version 7.

Other fallback considerations

Before you fall back to Version 7, be aware of the following considerations:

DB2 Control Center: If you use the DB2 Control Center, refer to *DB2 Management*
Clients Package Program Directory for more information about falling back.

Buffer pools: DB2 Version 8 maintains the Version 7 virtual buffer pool and hiperpool definitions at migration so that they can be used if you fall back.

NEWFUN precompiler option: You cannot execute a plan or package that uses a DBRM that was produced by precompiling in DB2 Version 8 with a value of YES for the NEWFUN precompiler option. You cannot BIND a DBRM that was precompiled with a value of YES for the NEWFUN precompiler option on Version 7 or earlier.

DISPLAY GROUPBUFFERPOOL output: After fallback, the DISPLAY GROUPBUFFERPOOL command's output reverts to the Version 7 format and displays only the operational coupling facility level.

Running DB2-supplied stored procedure DSNWZP in a WLM-established stored procedure address space: In DB2 Version 8, DB2-supplied stored procedure DSNWZP is defined to run in a WLM-established stored procedure address space and to use external module DSNWZP. In DB2 Version 7, DSNWZP must use external module DSNWZPR to run in a WLM-established stored procedure address space. You must alter DSNWZP to use DSNWZPR after fallback. To do this, execute this SQL ALTER statement:

```
ALTER PROCEDURE SYSPROC.DSNWZP EXTERNAL NAME DSNWZPR;
```

Page-fixes for buffer pools: If you defined a buffer pool using PGFIX YES in Version 8, it is defined with PGFIX NO after fallback. When you remigrate to Version 8, the buffer pool is defined with PGFIX YES.

Fallback procedure

Because the structure of the DB2 UDB for z/OS Version 8 catalog is used in Version 7 after falling back, the fallback procedure involves only a few steps:

1. Stop Version 8 activity.

2. Terminate all utilities that are running on Version 8.
3. Reactivate Version 7.
4. Reconnect TSO, IMS, and CICS to Version 7.
5. Start Version 7.
6. Verify fallback.

You can save your Version 8 TSO logon procedures and JCL for remigration to Version 8.

Fallback step 1: Stop DB2 Version 8 activity

Ensure that no recovery is required on system databases.

To stop Version 8 work, perform the following steps:

1. Issue the following command:
   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```
   The QUIESCE keyword allows DB2 to complete processing of currently executing programs. This activity might require some processing time.
2. Issue the following command:
   ```
   -DSN1 START DB2 ACCESS(MAINT)
   ```
   This command allows only the install-defined system administrators and system operators to access DB2.

   If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. Check to see that you started DB2 with the correct subsystem parameter load module.
3. Ensure that all work is complete.
   - Ensure that no units of recovery remain. Issue the following command:
     ```
     -DSN1 DISPLAY THREAD(*) TYPE(*)
     ```
     Then use RECOVER INDOUBT for any indoubt threads.
   - Ensure that no utility work remains. Issue the following command:
     ```
     -DSN1 DISPLAY UTILITY(*)
     ```
     Then, either allow utilities to complete before proceeding, or stop all utility processing with the following command:
     ```
     -DSN1 TERM UTILITY(*)
     ```
   - Ensure that no table spaces and index spaces in the DB2 directory (DSNDB01) or the DB2 catalog (DSNDB06) have write error ranges or deferred restart states. Issue the following command:
     ```
     -DSN1 DISPLAY DATABASE(DSNDB01) SPACENAM(*) RESTRICT
     -DSN1 DISPLAY DATABASE(DSNDB06) SPACENAM(*) RESTRICT
     ```
     A user with install-defined system administrator or system operator authority also must enter this command.

     Recover any table spaces and index spaces with write error range or deferred restart states.
4. To stop DB2, issue the following command:
   ```
   -DSN1 STOP DB2 MODE(QUIESCE)
   ```
   A user with SYSADM or SYSOPR authority also must enter this command.

If IRLM does not stop automatically when DB2 stops, stop IRLM manually. To stop IRLM, issue the following command, where *irlmproc* is the name you assigned to the IRLM startup procedure:

```
STOP irlmproc
```

Fallback step 2: Reactivate DB2 Version 7 code: DSNTIJFV

This job renames procedures to activate Version 7 and deactivate DB2 UDB for z/OS Version 8. SYS1.PROCLIB is the default target for JCL procedures. Add statements to rename other procedures, such as your IMS, CICS, TSO logon procedures, and batch procedures. You might also need to rename procedures in your jobs from Version 7.

You might want two sets of procedures, such as DSN1*xxxx* and DSN2*xxxx*, at all times, with an alias for the current release level.

If DSNTIJFV runs successfully, it produces a return code of 0. Check to ensure that all renames execute successfully.

If DSNTIJFV fails or abends, rerun only the renames that failed. If some of the procedures already exist, check carefully to ensure that procedures for the two releases are not mixed.

Fallback step 3: Reconnect TSO, IMS, and CICS to DB2 Version 7

Re-establish your Version 7 logon procedures and JCL, as well as the CICS and IMS connections.

If you overwrote the load module during migration to DB2 UDB for z/OS Version 8, re-assemble the RCT with the Version 7 libraries.

If you did not overwrite the load module, change the STEPLIB statements for CICS and IMS jobs so that they refer to the Version 7 libraries.

Fallback step 4: Start DB2 Version 7

Perform the following steps to start Version 7:

1. Start the IRLM.

   If you have not requested that DB2 automatically start the IRLM, start it before you start DB2. Use the following command, where *irlmproc* is the name that you assigned to the IRLM startup procedure:

   ```
   START irlmproc
   ```

   This is the value that you specified for the PROC NAME option on installation panel DSNTIPI.

   If you specified YES for the AUTO START option on installation panel DSNTIPI, DB2 starts the IRLM automatically.
2. Start DB2 from the z/OS console by using the following command:

   ```
   -DSN1 START DB2,PARM(DSNZPxxx)
   ```

   In this command, *-DSN1* is the subsystem command prefix that you defined for DB2, and *DSNZPxxx* is the name of the Version 7 subsystem parameter module. If you used the default name, DSNZPARM, you can omit the PARM parameter.

If DB2 starts successfully, two to five address spaces also start. These address spaces are *ssnm*MSTR and *ssnm*DBM1, and possibly *ssnm*SPAS, *ssnm*DIST, and *irlmproc*, where *ssnm* is the DB2 subsystem name and *irlmproc* is the IRLM procedure name.

If DB2 starts successfully, the series of restart messages that you receive concludes with these two messages:

```
DSNR002I   RESTART COMPLETED
DSN9022I   DSNYASCP '-DSN1 START DB2' NORMAL COMPLETION
```

3. **If you have done distributed processing with your DB2 UDB for z/OS Version 8 subsystem**, check message DSNR005I for the number of indoubt threads after you start DB2.

   If you find no indoubt threads, continue falling back as if you had not done any distributed processing. If you find indoubt threads, issue the following command:

   ```
   -DSN1 DISPLAY THREAD(*) TYPE(INDOUBT)
   ```

   If the number of indoubt threads that are reported in the DSNV408I messages is equal to the number of threads that are reported in the DSNR005I message, continue falling back as if you had not done any distributed processing. If fewer indoubt threads are reported by DSNV408I messages than in message DSNR005I, proceed as follows:

   a. Stop Version 7.
   b. Determine which units of work are incomplete by scanning the DB2 recovery log with the DB2 Version 8 DSN1LOGP utility. Use the SUMMARY option of this utility.
   c. Examine the DSN1LOGP output to find all the DSN1162I messages that have a COORDINATOR name in a remote location. Each of these messages identify an indoubt DBAT. Record the LUWID that is displayed in each message.
   d. Decide whether to commit or abort each indoubt DBAT. One way to do this is by contacting the COORDINATOR location. If it is another DB2 subsystem, use the DISPLAY THREAD command to help you decide.
   e. If you have not already done so during migration, apply the fallback PTF that is supplied with Version 8.
   f. Start Version 7 again.
   g. Issue the RECOVER INDOUBT ACTION(*correct decision*) LUWID(*luwid*) command to resolve each indoubt DBAT.
4. **If you have not done distributed processing with your DB2 UDB for z/OS Version 8 subsystem**, check outstanding restrictions after you start DB2. Identify databases whose uses are restricted by issuing the following command:

   ```
   -DSN1 DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
   ```

   You can start some of these databases at this time. For more information on starting restricted databases, see Part 4 (Volume 1) of of *DB2 Administration Guide*.
5. If DB2 does not start properly, it usually abends with a reason code that indicates where the error occurred. To find the error, check the set of definitions for the associated resource. A common cause of startup failure is that the BSDS does not match the subsystem parameter values; ensure that the startup procedure is pointing to the correct BSDS and subsystem parameter. Also, check that the subsystem parameter member that you specified (or is used by default) when you started DB2 is the one that job DSNTIJUZ built. Check the JCL for the DB2 startup procedure.

6. Optionally, start TSO. If you want to use the TSO SUBMIT command to do housekeeping and fallback verification, you must start TSO (if it is not already started).

Fallback step 5: Verify fallback

At this point, you must perform some of your own testing to determine if the fallback was successful. You cannot run the DB2 UDB for z/OS Version 8 samples on Version 7.

Verify fallback by following these steps:

1. Run the Version 7 sample applications.
2. Test your own applications.
3. Retry the problem for which you decided to fall back.

Remigrating

Migrating after falling back (remigrating) is simpler than the migration process that is described in this chapter.

When remigrating, refer to “Migration considerations” on page 304 because many of those considerations apply to remigrations, too. Which considerations apply depends on the type of activity that took place on your Version 7 subsystem after falling back.

A plan or package is automatically rebound in Version 8 when it is executed for the first time after remigration if it was not explicitly bound in Version 7. However, if you specified NO for the AUTO BIND option on installation panel DSNTIPO, automatic binds are disabled. This means that the plan or package from your previous release is the one that runs in Version 8, so the plan or package does not benefit from Version 8 enhancements.

Recommendation: When remigrating, follow these steps

1. Run DSN1COPY with the CHECK option on the Version 7 catalog table spaces. Also, run DSN1CHKR on Version 7. For information about these utilities, see “Migration step 2: Run the link checker on DB2 Version 7 table spaces (optional)” on page 322. Finally, execute the queries in member DSNTESQ of *prefix*.SDSNSAMP.
2. Take an image copy of the Version 7 catalog by using the DSNTIJIC job.

   This step is not required, but it is recommended. See “Migration step 5: Take image copies of the directory and catalog: DSNTIJIC” on page 325 for more information.
3. Stop Version 7.
4. Reconnect TSO, IMS, and CICS to DB2 Version 8.

   Re-establish your Version 8 logon procedures and JCL, as well as your Version 8 CICS and IMS connections.
5. Rebuild Version 8 cataloged procedures.

   Rename the Version 8 procedures that were renamed by job DSNTIJFV during fallback. If job DSNTIJFV was not run, you need to rerun job DSNTIJMV. Comment out step 1 (DSNTIMP), which defines Version 8 to z/OS, and run the job. (You do not need to define Version 8 to z/OS a second time.)
6. Start DB2 Version 8.

   Ensure that you are using your Version 8 subsystem parameter load module.
7. Take an image copy of the Version 8 catalog by using the DSNTIJIC job.

For information about job DSNTIJIC, see "Migration step 5: Take image copies of the directory and catalog: DSNTIJIC" on page 325.

8. Return to using the Version 8 DSNWZP instead of DSNWZPR by submitting the following command:

   ```
   ALTER PROCEDURE SYSPROC.DSNWZP EXTERNAL NAME DSNWZP;
   ```

9. Verify your DB2 Version 8 system.

   For information about this procedure, see "Migration step 27: Verify your DB2 Version 8 compatibility mode system (optional)" on page 342.

Chapter 9. Enabling new-function mode

After completing the migration steps described in Chapter 8, “Migrating the DB2 subsystem to compatibility mode,” you must convert the DB2 catalog in order to use new Version 8 function.

Important: All members of a data sharing group must have migrated to Version 8 compatibility mode successfully before you begin the enabling-new-function mode process.

A point of consistency needs to be created for the catalog and directory before enabling new-function mode. The Quiesce Utility should be used to establish a point of consistency for the catalog and directory table spaces; note that DSNDB01.SYSUTILX should be quiesced by itself. Updates to the DB2 catalog and directory should be avoided while in enabling new-function mode.

Generally, you must complete the following steps to complete conversion of the catalog:

1. Run the installation CLIST using the enabling-new-function mode option. Several installation panels will be presented in which you indicate space requirements. Refer to Chapter 6, “Installing, migrating, and updating system parameters” for information about running the CLIST. During enabling-new-function mode processing, the CLIST generates installation jobs DSNTIJMC, DSNTIJNE, DSNTIJNF, DSNTIJNG, DSNTIJEN, DSNTIJNH, and DSNTIJNR, and customizes the Version 8 IVP jobs.
2. Create a copy of the DB2 UDB for z/OS Version 8 compatibility mode catalog and directory for back-up purposes. See “Migration step 5: Take image copies of the directory and catalog: DSNTIJIC” on page 325 for information on job DSNTIJIC.
3. Run installation job DSNTIJNE. Job DSNTIJNE performs several functions:
   - Saves the current RBA or LRSN in the BSDS.
   - Changes types and lengths of existing catalog columns.
   - Converts catalog data to Unicode.
   - Changes buffer pools for several catalog table spaces.
   - Changes catalog indexes with varying length columns to NOT PADDED.
   - Changes page size of several catalog table spaces.
4. Run installation job DSNTIJNF. DSNTIJNF puts the DB2 subsystem in new-function mode.
5. Run installation job DSNTIJMC. DSNTIJMC converts the stored procedures for JCBC and ODBC support for use in new-function mode.

   Important: Run this step only if the stored procedures were defined in DB2 Version 7 or earlier.
6. Run installation job DSNTIJNG. DSNTIJNG rebuilds the DSNHDECP module to specify new-function mode as the default. The precompiler will now accept program source code that uses new Version 8 function.

Attention: You cannot fall back from Version 8 new-function mode. Do not convert to Version 8 new-function mode until you are certain that you will not need to fall back to Version 7.

The following topics provide additional information:
- "Conversion considerations"
- "Compatibility mode" on page 354
- "Enabling-new-function mode" on page 357
- "New-function mode" on page 359
- "Returning from new-function mode to enabling-new-function mode" on page 361
- "Considerations for the BSDS" on page 361

Conversion considerations

Be aware of the following changes that might affect your conversion to Version 8 new-function mode.

Allocate more space for SYSSTATS table space: You may need to allocate more space for the SYSSTATS table space. In Version 8, the SYSCOLDISTSTATS and SYSCOLDIST catalog tables contain more data than in previous versions. Increase the size of the shadow data sets on panel DSNTIP01.

Increase VSAM data set size: You should increase the size of the catalog table space and index space VSAM data sets because of the increased length of names in the catalog in Version 8. Increase the size of the shadow data sets on panel DSNTIP01.

New utility for converting BSDSs: In Version 8, you can convert BSDSs with a new utility (DSNJCNVB) to support more data sets, up to 10000 per copy for archive logs and 93 per copy for active logs. You can run the conversion utility once DB2 is in new-function mode. Converting the BSDS allows support for a larger number of active log data sets and archive log data sets. For information about converting BSDSs, see "Considerations for the BSDS" on page 361.

Character conversions between Unicode CCSIDs and EBCDIC CCSIDs: The character conversions between Unicode CCSIDs 367, 1208, and 1200, and EBCDIC CCSIDs 37, 500, and 1047 must be defined.

CCSIDs for IFCID records: All SQL statements that are written into IFCID records are in Unicode UTF-8, not in EBCDIC. See "Tracing parameters panel: DSNTIPN" on page 149 for more information IFCID records.

SQL statements parsed in Unicode: DB2 parses SQL statements in Unicode UTF-8. If an application program uses a different CCSID (for example, an EBCDIC CCSID), DB2 converts the application program internally to Unicode for processing. This conversion occurs for the entire application program, including host language statements. Message DSNH330I might be issued if errors are encountered, such as an invalid code point, a mismatch between shift-in and shift-out characters, and an absence of half of a DBCS character.

String constants have new maximum lengths: To check the length of a string constant, DB2 uses the Unicode representation of the string constant even if the eventual destination of the constant does not use Unicode. This length might differ from the length that you entered, if you entered the string constant in a CCSID other than UTF-8. In some cases, a string that was valid in Version 7 might be flagged as too long in Version 8, and message DSNH102I or SQLCODE -102 is generated. This incompatibility can occur only if the string contains one or more characters whose Unicode representations require more bytes than their original representations and this expansion causes the string to grow beyond the maximum allowed length.

This incompatibility can exist in all modes of DB2 or disappear on entering new-function mode. The following strings might be flagged as too long in any mode of DB2:

- In ALTER INDEX or CREATE INDEX, VALUES (constant)
- In ALTER TABLE or CREATE TABLE, FIELDPROC program-name (constant)
- In ALTER TABLE or CREATE TABLE, CHECK (check-condition)
- In ALTER TABLE, CREATE TABLE, and DECLARE GLOBAL TEMPORARY TABLE, under DEFAULT, in a constant
- In COMMENT ON, IS string
- In LABEL ON, IS string
- SET CURRENT LOCALE LC_CTYPE
- SET CURRENT OPTIMIZATION HINT
- SET CURRENT SQLID
- In SIGNAL SQLSTATE, diagnostic-string-constant

The following strings might be flagged as too long in compatibility mode and enabling-new-function mode, but not in new-function mode, because new-function mode allows longer strings:

- CASE expression
- CALL procedure-name (expression)
- Expression in a predicate
- Expression in a WHERE clause
- Expression in a HAVING clause
- Expression in a SELECT clause in a subselect
- Join expression
- Expression specified in a built-in function

Convert PLAN_TABLE: To see new features of EXPLAIN output, use ALTER TABLE to add the new Version 8 columns to the existing PLAN_TABLE after you have converted your DB2 subsystem to Version 8 new-function mode. See *DB2 SQL Reference* for more information.

Dropped catalog tables: Two catalog tables are dropped during Version 8 enabling-new-function mode processing: SYSIBM.SYSLINKS and SYSIBM.SYSPROCEDURES. When you are in new-function mode, you can delete the VSAM data set to the SYSPROCEDURES index, DSNKCX01.

Invalidated statistics: During the enabling-new-function mode processing, SYSCOLUMNS.STATSTIME statistics become invalidated.

Run RUNSTATS to re-create statistics for the DB2 catalog tables: Enabling-new-function mode invalidates statistics on the catalog tables. Therefore, DB2 uses default statistics when it calculates access paths. Incorrect statistic values can cause problems with your DB2 subsystem. Run RUNSTATS on the seventeen catalog table spaces to gather new statistics.

Change data capture cannot be enabled on catalog tables during enabling-new-function mode: During enabling-new-function mode processing, change data capture is disabled on most catalog tables. You cannot re-enable change data capture until your DB2 subsystem is in Version 8 new-function mode.

Invalidated plans and packages: During the enabling-new-function mode processing, some plans and packages become invalidated. Queries on these tables cannot be resolved by rebinding. To determine which plans and packages are no longer valid, use the following queries:

```
SELECT DISTINCT DNAME
  FROM SYSIBM.SYSPLANDEP
 WHERE BCREATOR = 'SYSIBM'
  AND ((BTYPE = 'I' AND BNAME = 'DSNKCX01')
    OR (BTYPE = 'T' AND BNAME IN
       (SELECT T.NAME
          FROM SYSIBM.SYSTABLES AS T
         WHERE T.DBID =6
           AND T.OBID<>0
           AND T.NAME NOT IN ('SYSCOPY','SYSOBDS'))))
 ORDER BY DNAME;
```

```
SELECT DISTINCT COLLID, NAME,
  FROM SYSIBM.SYSPACKDEP, SYSIBM.SYSPACKAGE
 WHERE ((BTYPE = 'I' AND BNAME = 'DSNKCX01')
     OR (BTYPE = 'T' AND BNAME IN
        (SELECT T.NAME
           FROM SYSIBM.SYSTABLES AS T
          WHERE T.DBID =6
            AND T.OBID<>0
            AND T.NAME NOT IN ('SYSCOPY','SYSOBDS'))))
      AND LOCATION   = ' '
      AND BQUALIFIER = 'SYSIBM'
      AND COLLID     = DCOLLID
      AND NAME       = DNAME
      AND CONTOKEN   = DCONTOKEN
  ORDER BY COLLID, NAME, VERSION;
```

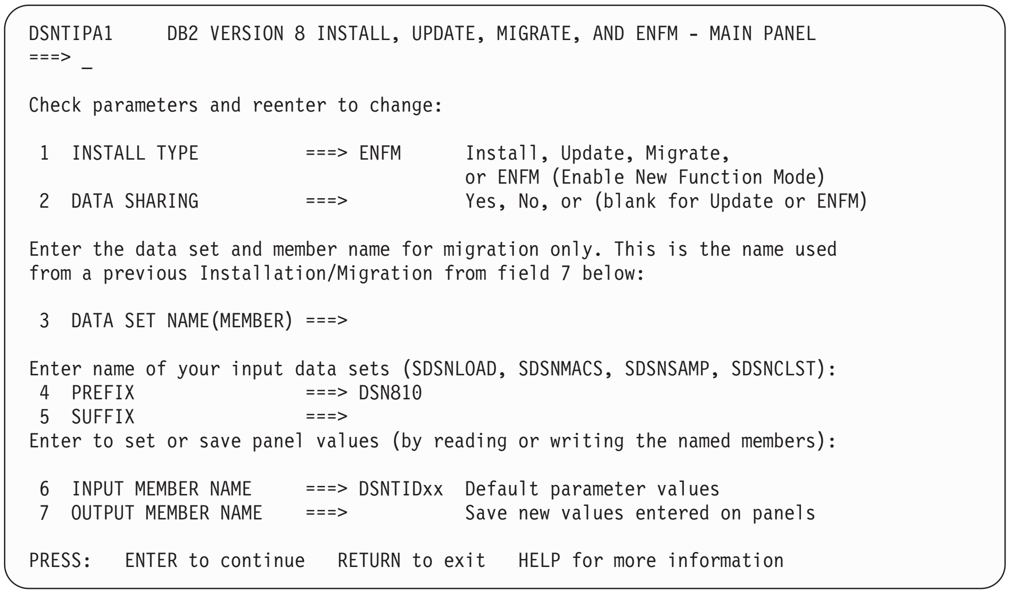
These two queries are commented out in the Version 8 member DSNTESQ of *prefix*.SDSNSAMP. You can run these queries from SPUFI or from a dynamic SQL program like DSNTEP2.

After conversion to new-function mode, you can explicitly rebind these plans and packages or let DB2 rebind them automatically. See Part 4 of *DB2 Application Programming and SQL Guide* for suggestions on rebinding these plans and packages.

The following topics describe the three Version 8 modes and the specific steps you must take to complete catalog conversion.

Compatibility mode

After you complete the migration process, DB2 is in Version 8 compatibility mode. During this time, you cannot use Version 8 new function. In order to progress to Version 8 enabling-new-function mode, you must run the installation CLIST again using the ENFM option on panel DSNTIPA1 as shown in Figure 42. More information about the fields in this panel can be found on “Main panel: DSNTIPA1” on page 94.



```
DSNTIPA1     DB2 VERSION 8 INSTALL, UPDATE, MIGRATE, AND ENFM - MAIN PANEL
===> _

Check parameters and reenter to change:

 1  INSTALL TYPE          ===> ENFM      Install, Update, Migrate,
                                         or ENFM (Enable New Function Mode)
 2  DATA SHARING          ===>           Yes, No, or (blank for Update or ENFM)

Enter the data set and member name for migration only. This is the name used
from a previous Installation/Migration from field 7 below:

 3  DATA SET NAME(MEMBER) ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DSN810
 5  SUFFIX                ===>
Enter to set or save panel values (by reading or writing the named members):

 6  INPUT MEMBER NAME     ===> DSNTIDxx  Default parameter values
 7  OUTPUT MEMBER NAME    ===>           Save new values entered on panels

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

Figure 42. DSNTIPA1: Install, update, and migrate DB2 - main panel

The value of the INPUT MEMBER NAME field is the value that you specified as the output member during migration to compatibility mode. In a data-sharing environment, it is the output member of the first group member to migrate to compatibility mode. In Version 8 enabling-new-function mode, the CLIST customizes the installation jobs DSNTIJNE, DSNTIJNF, DSNTIJNG, DSNTIJEN, DSNTIJNH, and DSNTIJNR as well as the Version 8 sample jobs.

The CLIST calculates preliminary space allocations for shadow data sets and displays these recommendations on panel DSNTIP00, as shown in Figure 43. This panel lists the 18 DB2 directory and catalog table spaces that are transformed during enabling-new-function mode processing. It also lists the CLIST-determined device types, volumes, and space allocations that are used for the shadow data set during enabling-new-function mode processing. When you run DSNTIJNE, the shadow data sets replace the current data sets for the table and index spaces being converted.

Update the device type, volume, and space settings as needed by overtyping in the appropriate fields. Your entries must conform to the following specifications:

- DASD DEVICE entries must be 1-8 alphanumeric characters (including A-Z, 0-9, $, #, and @). Shadow data sets must be on disk.
- VOL/SERIAL entries must be 1-6 alphanumeric characters.
- PRIMARY RECS and SECONDARY RECS entries must be 1-6 numeric characters, indicating the number of records.

You must leave the DATA SHARING and DATA SET NAME(MEMBER) fields blank when enabling new-function mode. The INPUT MEMBER NAME that you specify for conversion should be the same as the OUTPUT MEMBER NAME that you specified during migration to compatibility mode. In a data sharing environment, the INPUT MEMBER NAME that you specify for conversion should be the same as the OUTPUT MEMBER NAME used when migrating the first member of the data sharing group to Version 8.

After you have specified ENFM on panel DSNTIPA1, panel DSNTIPT appears. Choose the output SDNSAMP data set on installation panel DSNTIPT. If you use the same data set name, it is not deleted or reallocated. You can only change the value of SAMPLE LIBRARY on this panel. See "Data set names panel 1: DSNTIPT" on page 113 for more information.

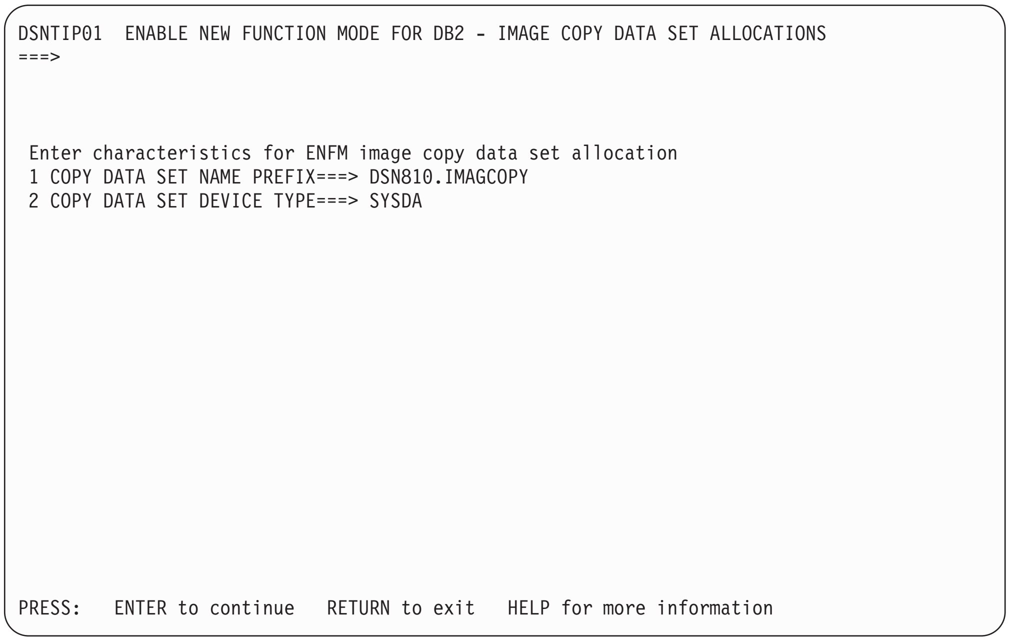
```
DSNTIP00     ENABLE NEW FUNCTION MODE FOR DB2 - SHADOW DATA SET ALLOCATIONS
===>

   OBJECT     DASD DEVICE      VOL/SERIAL   PRIMARY RECS   SECONDARY RECS
 1 SPT01      ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 2 SYSDBASE   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 3 SYSDBAUT   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 4 SYSDDF     ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 5 SYSGRAUT   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 6 SYSGROUP   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 7 SYSGRTNS   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 8 SYSHIST    ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
 9 SYSJAVA    ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
10 SYSOBJ     ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
11 SYSPKAGE   ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
12 SYSPLAN    ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
13 SYSSEQ     ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
14 SYSSEQ2    ==> SYSDA        ==> DSNV01   ==> 114        ==> 30
15 SYSSTATS   ==> SYSDA        ==> DSNV01   ==> 10         ==> 10
16 SYSSTR     ==> SYSDA        ==> DSNV01   ==> 10         ==> 10
17 SYSUSER    ==> SYSDA        ==> DSNV01   ==> 10         ==> 10
18 SYSVIEWS   ==> SYSDA        ==> DSNV01   ==> 10         ==> 10
19 INDEXES    ==> SYSDA        ==> DSNV01  Catalog and directory index shadows
PRESS:  ENTER to continue  RETURN to exit  HELP for more information
```

Figure 43. DSNTIP00: Enable New Function Mode for DB2 - shadow data set allocations

Panel DSNTIP01, which appears after panel DSNTIP00 and is shown in Figure 44 on page 357, lists the name and target device type for the image copy data sets.

Update the data set name and target volume settings as needed by overtyping in the appropriate fields. Your entries must conform to the following specifications:

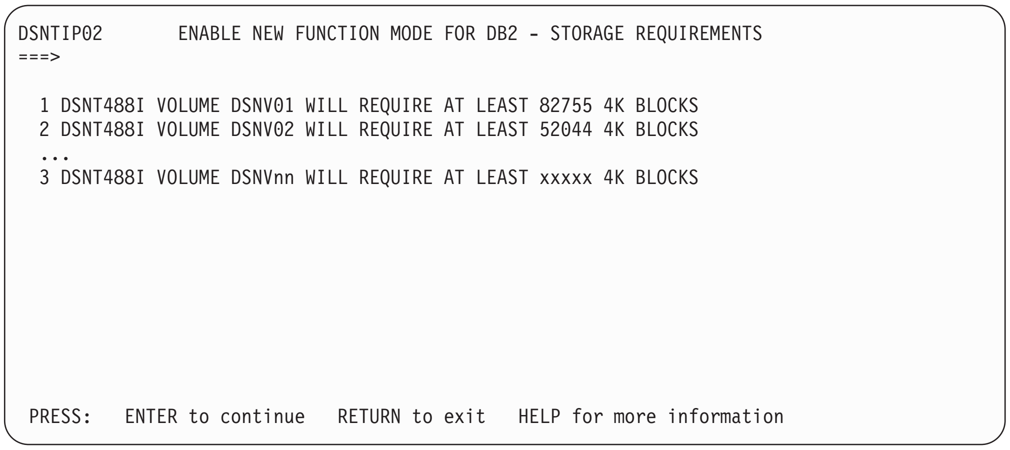
- The COPY DATA SET NAME PREFIX value must follow z/OS standards for data set names.
- The COPY DATA SET DEVICE TYPE value must be 1-8 alphanumeric characters, including A-Z, 0-9, $, #, and @. You can enter either tape or disk devices as targets for image copy data sets. If TAPE, 3480, or 3490 is specified for a table space, the CLIST customizes the enabling-new-function mode job with no SPACE parameter for the corresponding image copy data set; otherwise, the CLIST uses the same space settings as specified for that table space's shadow data set.



```
DSNTIP01  ENABLE NEW FUNCTION MODE FOR DB2 - IMAGE COPY DATA SET ALLOCATIONS
===>



 Enter characteristics for ENFM image copy data set allocation
 1 COPY DATA SET NAME PREFIX===> DSN810.IMAGCOPY
 2 COPY DATA SET DEVICE TYPE===> SYSDA


















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

Figure 44. DSNTIP01: Enable New Function Mode for DB2 - image copy data set allocations

Panel DSNTIP02, which appears after panel DSNTIP01 and is shown in Figure 45, displays a summary of space requirements that are based on the data that you entered on previous panels. To accept the space requirements that are displayed for each volume, press the ENTER key. Pressing the ENTER key also signals the CLIST to generate the enabling-new-function mode job and Version 8 sample jobs. To revise the image copy data set names and target devices by returning to panel DSNTIP01, press the RETURN key.



```
DSNTIP02       ENABLE NEW FUNCTION MODE FOR DB2 - STORAGE REQUIREMENTS
===>

  1 DSNT488I VOLUME DSNV01 WILL REQUIRE AT LEAST 82755 4K BLOCKS
  2 DSNT488I VOLUME DSNV02 WILL REQUIRE AT LEAST 52044 4K BLOCKS
  ...
  3 DSNT488I VOLUME DSNVnn WILL REQUIRE AT LEAST xxxxx 4K BLOCKS











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

Figure 45. DSNTIP02: Enable New Function Mode for DB2 - storage requirements

Enabling-new-function mode

Enabling-new-function mode takes place between compatibility mode and new-function mode. New function is not available during enabling-new-function mode.

Important: After you begin enabling-new-function mode, you cannot fall back to Version 7 or return to Version 8 compatibility mode.

You must perform several actions before beginning enabling-new-function mode processing:

- Ensure that the BP8K0, BP16K0, and BP32K buffer pools exist. If these buffer pools do not exist before you begin enabling-new-function mode, the processing fails. For data sharing groups, you must also define the GBP8K0, GBP16K0, and GBP32K buffer pools. For more information about catalog table space buffer pools, see *DB2 SQL Reference*.
- You should increase the size of the shadow data sets before entering enabling-new-function mode in order to accommodate longer names in the catalog.
- Take an image copy of the DB2 catalog and directory.

  Important: You must take an image copy of the DB2 catalog and directory before running DSNTIJNE for the first time.
- Create shadow data sets for user-defined indexes on the DB2 catalog that reside on user-managed storage.

  Important: You must create shadow data sets for user-defined indexes that are on user-managed storage before you run job DSNTIJNE. DB2 uses these shadow data sets in the REORG steps that convert the catalog table spaces to new-function mode. One of the queries in job DSNTIJPM identifies the affected indexes.

DSNTIJNE consists of the following job steps:

ENFM0000
: Terminate pending utilities for catalog conversion.

ENFM0001
: Places the DB2 subsystem in enabling-new-function mode. Creates the index and table in the SYSALTER tablespace. The first time that you run job DSNTIJNE, saves the RBA or LRSN of the system log in the BSDS. In a data sharing environment, records the LRSN in the BSDS of the member on which job DSNTIJNE runs.

ENFM0*nn*0
: Check the new-function mode status of the *xxxxxxxx* table space.

ENFM0*nn*1
: Clean up the work files for converting the *xxxxxxxx* table space.

ENFM0*nn*3
: Allocates shadow data sets for converting the *xxxxxxxx* table space.

ENFM0*nn*7
: Converts the *xxxxxxxx* table space to Unicode.

ENFM0*nn*8
: Creates an image copy of the *xxxxxxxx* table space.

ENFM0*nn*9
: Deletes work files used to convert the *xxxxxxxx* table space.

ENFM9900
: Terminates pending utilities for catalog conversion.

Take an image copy of the DB2 catalog and directory after job DSNTIJNE has completed processing.

If you need to stop enabling-new-function mode processing before job DSNTIJNE completes, you can do so by running job DSNTIJNH. This job stops enabling-new-function mode processing after it finishes converting the current job step. If you pause enabling-new-function mode processing, take an image copy of the DB2 catalog and directory.

Recommendation: Do not manually reorganize the tablespace or rebuild index jobs on the catalog before the completion of ENFM00*nn*7.

If you need to determine how much of the enabling-new-function mode process has been completed, you can use the DISPLAY GROUP DETAIL command.

Job DSNTIJNF contains the following job steps:

ENFM9700
Verifies that the DB2 catalog and directory conversion is completed.

DSNTIAP
Binds a package for a message routine that is used by SPUFI and DCLGEN, and binds that package into the plans for SPUFI and DCLGEN.

If you run the CLIST again after you have entered enabling-new-function mode and if you did not accept the calculated space allocations on panel DSNTIP00, the CLIST displays panel DSNTIP03 as shown in Figure 46. To use the calculated space allocations, press ENTER. To use the saved values from your input member, press RETURN.

```
DSNTIP03

 ENABLE NEW FUNCTION MODE:

 WARNING: Calculated values for ENFM space
 allocations will replace one or more values
 saved from a previous session

 PRESS:  ENTER to overwrite previous space settings
         RETURN to retain previous space settings
```

Figure 46. DSNTIP03

New-function mode

New-function mode begins after job DSNTIJNF completes successfully. When in new-function mode, the DB2 catalog has been converted to Unicode and all Version 8 function is available for use. You cannot fall back to Version 7 or return to Version 8 compatibility mode. A Version 8 new-function mode or Version 8 enabling-new-function mode DB2 subsystem cannot coexist with a Version 7 DB2 subsystem.

After job DSNTIJNF completes, perform the following actions:

1. Run job DSNTIJLR. Job DSNTIJLR deploys job DSNLEUSR, a stored procedure that encrypts the authorization ID and password. For more information about DSNLEUSR, see *DB2 Administration Guide*.
2. Run job DSNTIJNG. Job DSNTIJNG modifies your DSNHDECP module to allow the DB2 precompiler to accept Version 8 new-function SQL statements by default.
3. Include DSNTIAP in the plans if you bound special SPUFI packages and plans for SPUFI in Version 7. In DB2 Version 8 in new-function mode, SPUFI requires DSNTIAP. To bind the special SPUFI plans again to include DSNTIAP, issue the following commands:

```
BIND PACKAGE(TIAP1047) MEMBER(DSNTIAP) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPCS1047) -
     PKLIST(SPCS1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
     ISOLATION(CS) ENCODING(1047) ACTION(REPLACE)
BIND PLAN(SPRR1047) -
     PKLIST(SPRR1047.DSNESM68 -
            TIAP1047,DSNTIAP) -
     ISOLATION(RR) ENCODING(1047) ACTION(REPLACE)
```

For more information, see "Special packages and plans for SPUFI" on page 300.

Attention: In new-function mode, your DSNHDECP module must specify NEWFUN=YES. In a data-sharing environment, if you use more than one DSNHDECP module, you must modify and run the jobs that you use to maintain these DSNHDECP modules to specify NEWFUN=YES.

DSNTIJNG consists of the following job steps. Do not run DSNTIJNG until DSNTIJNF has completed successfully.

DSNTIZP
Assembles the DSNHDECP data-only load module.

DSNTIZQ
Link-edits the DSNHDECP load module.

DSNTIMQ
Performs SMP/E processing for DSNHDECP.

During enabling-new-function mode processing, DATA CAPTURE is set to NONE on all catalog tables except SYSCOPY. After you enter new-function mode, if you want to use DATA CAPTURE on these tables, you must manually re-enable it. To re-enable DATA CAPTURE, issue the following command:

```
ALTER TABLE SYSIBM.SYSTABLES DATA CAPTURE CHANGES;
```

Important: You cannot re-enable DATA CAPTURE until your DB2 subsystem is in Version 8 new-function mode.

Two DB2 catalog tables are deleted during enabling-new-function mode processing because they are not used in Version 8. When you are in new-function mode, you can delete the VSAM data set for index DSNKCX01.

Optionally, you can use job DSNTIJNR to convert a Version 7 RLST to a Version 8 RLST. You can continue to use an RLST that has been defined from a release earlier then Version 8 but you cannot take advantage of long name support. Do not convert your Version 7 RLST until you have committed to new-function mode.

Recommendation: Use the ALTER INDEX statement with the NOT PADDED option to modify user-defined indexes on catalog tables if either of the following conditions is true:

- A column of the user-defined index key was altered to VARCHAR during the enabling-new-function mode process.
- The length of a column of the user-defined index was extended during the enabling-new-function mode process.
- The user-defined index has a pre-existing VARCHAR column in the index.

After you modify the index so that the NOT PADDED option is in effect, you must rebuild the index.

Recommendation: Alter your DB2 Version 8 buffer pools that have frequent page reads or frequent page writes to use PGFIX YES if you have sufficient real storage available for the these buffer pools. Fixing the buffer pages in real storage once and keeping them fixed avoids the processing time that DB2 needs to fix and free pages each time there is an I/O. In some cases, this processing time can be as much as 10% for I/O intensive workloads. To use this option, issue the following command:

```
ALTER BPOOL(bpname) VPSIZE(vpsize) PGFIX(YES)
```

Where *bpname* is the name of the buffer pool and *vpsize* is the size of the virtual pool.

Returning from new-function mode to enabling-new-function mode

If you are in new-function mode but do not want users to use new Version 8 functions, you can return to enabling-new-function mode.

Attention: You cannot return to Version 8 compatibility mode or fall back to Version 7 after you have entered enabling-new-function or new-function mode.

To return to enabling-new-function mode, you must complete the following steps:

1. Run job DSNTIJEN.
   Important: This job sets the DB2 mode to enabling-new-function mode. The catalog and table space remain in Unicode.
2. Run job DSNTIJNG. You must ensure that the DSNHDECM invocation specifies NEWFUN=NO in this job.
3. Verify that DSNHDECM specifies NEWFUN=NO.
4. If you are in a data sharing environment and you use more than one DSNHDECP module, modify and run the jobs that you use to maintain these DSNHDECP modules to specify NEWFUN=NO.

When you are ready to return to new-function mode, you must complete the following steps:

1. Run job DSNTIJNF.
2. Run job DSNTIJNG. You must ensure that the DSNHDECM invocation specifies NEWFUN=YES in this job.
3. If you are in a data sharing environment and you use more than one DSNHDECP module, modify and run the jobs that you use to maintain these DSNHDECP modules to specify NEWFUN=YES.

Considerations for the BSDS

You can create a larger BSDS, which provides support for more active log and archive log data sets, by running the DSNJCNVB conversion utility. After conversion, the BSDS can support up to 93 active log data sets and 10000 archive log data sets per copy. You must be in new-function mode to run the conversion utility.

Once you are in new-function mode, take the following steps to prepare to run the utility:

1. Rename your existing BSDS copy 1 data set. Retain the original copy of the BSDS in order to restore it in case of a failure during conversion.
2. Allocate a larger BSDS data set using the VSAM DEFINE statement in installation job DSNTIJIN, using the original BSDS name.
3. Use VSAM REPRO to copy the original data set to the new, larger data set.
4. Repeat steps 1-3 for copy 2 if you are using dual-BSDSs.

Invoke the conversion utility, DSNJCNVB, by entering the following code on an EXEC statement within a JCL job:

```
//   EXEC PGM=DSNJCNVB
```

SYSUT1 and SYSUT2 DDNAMES specify the BSDS copy 1 and copy 2 data sets, respectively, as input to this program. (SYSUT2 is optional.) DB2 must be stopped to run the conversion utility.

Recommendation: Change the number of archive log data sets by running the DSNJU003 utility. For information about changing the maximum number of archive log data sets that can be allocated, see "Archive log data set parameters panel: DSNTIPA" on page 210.

Chapter 10. Verifying installation with the sample applications

| Use the sample applications to verify either installation or migration to
| new-function mode. During verification, run all sample applications under the same user ID; this user ID must have SYSADM authority. Otherwise, errors might occur.

Recommendation: If you are migrating, run portions of the sample applications from Version 7 after you finish migration. This verifies the migration and ensures
| that the old jobs work with DB2 UDB for z/OS Version 8 new-function mode. You can run the Version 8 sample applications next. For information about how to run the sample applications from the previous release, see "Migration step 27: Verify your DB2 Version 8 compatibility mode system (optional)" on page 342.

| **Attention:** Do not run the Version 8 IVP while in Version 8 compatibility mode or
| Version 8 enabling-new-function mode.

The installation verification procedure (IVP) consists of eight phases: seven verification phases and one cleanup phase that drops sample objects. Each of the seven verification phases tests one or more DB2 functions or attachment facilities. Certain phases of the verification procedure might not apply to the environment in which your DB2 subsystem operates, so you might not need to perform all phases. In some cases, the steps and return codes differ when you run the fallback release and Version 8 phases. These differences are noted under the proper phase.

To help you perform the verification procedure, DB2 provides several jobs that invoke sample applications. You run the same jobs regardless of whether you are
| installing DB2 for the first time or converting your Version 8 catalog to
| new-function mode.

Recommendation: Run the IVP jobs with DB2 started in unrestricted mode because
restricted mode (ACCESS(MAINT)) does not allow use of stored procedures and
user-defined functions.

The installation CLIST tailored and loaded these jobs into *prefix*.NEW.SDSNSAMP that you created during your installation or conversion to new-function mode. Sometimes the verification jobs access information from the untailored *prefix*.SDSNSAMP library that existed before installation or conversion to new-function mode.

If you are installing a data sharing group, run the installation verification procedure (IVP) after you install or convert the originating system. You do not need to run the IVP after you enable the originating system or after you install a new data sharing member. See the installation procedures in *DB2 Data Sharing: Planning and Administration* for more information about verifying that your data sharing definitions are correctly established and that Sysplex parallelism is enabled.

Do not delete the fallback release sample objects from your subsystem. You need
| them to verify the success of a migration to Version 8 compatibility mode and to fall back to Version 7 in case of a Version 8 failure. If you are migrating, you can run the fallback release sample programs on Version 8 compatibility mode without any preparation to test the migration process.

Most DB2 sample objects have unique names to differentiate them from objects of previous releases. This allows sample programs for multiple releases to coexist.

The JCL that is provided for CICS and IMS sets up transaction identifiers for the sample applications.

Attention: Due to the changes in supported programming languages and compilers, you might need to increase the region size of DB2 before running the IVP jobs. See "Data set names panel 2: DSNTIPU" on page 118 for more information about supported languages and compilers.

The following topics provide additional information:
- "Installation verification phases and programs"
- "Planning for verification" on page 369
- "Special considerations for COBOL programs" on page 370
- "Special considerations for C and C++ programs" on page 371
- "Special considerations for PL/I programs" on page 372
- "Phase 0: Deleting the sample objects (DSNTEJ0)" on page 372
- "Phase 1: Creating and loading sample tables" on page 373
- "Phase 2: Testing the batch environment" on page 378
- "Phase 3: Testing SPUFI, DRDA Access, dynamic SQL, and TSO" on page 383
- "Phase 4: Testing the IMS environment" on page 387
- "Phase 5: Testing the CICS environment" on page 390
- "Phase 6: Accessing data at a remote site" on page 394
- "Phase 7: Accessing LOB data" on page 407
- "Working with the sample applications" on page 411
- "Scenarios" on page 415
- "Edit exit routine" on page 434
- "Huffman compression exit routine" on page 434
- "Sample field procedure" on page 435
- "Dynamic SQL statements (DSNTESA, DSNTESQ)" on page 435
- "Dynamic SQL programs (DSNTIAD, DSNTEP2, DSNTIAUL)" on page 437

Installation verification phases and programs

Table 56 on page 365 shows the programs that are run in each phase of the verification procedure. These programs need to be run sequentially by phase because the output of some jobs is used as input for following jobs. You **must** use the same compiler for each job. For example, if you use IBMCOB for DSNTEJ2C, you must use IBMCOB for all COBOL verification programs.

Run Phase 0 (job DSNTEJ0) only if you want to remove all the verification processing that you have completed so that you can begin the verification procedure again. Phases 1-3 test the TSO and batch environments, including user-defined functions. Phase 4 is for IMS users only, and Phase 5 is for CICS users only. Phase 6 sets up the sample tables and stored procedures for distributed processing. Phase 7 tests the LOB feature with sample tables, data, and programs.

prefix.SDSNSAMP contains the program source. Details about how to print it are provided in "Printing the sample application listings" on page 411.

When you complete the verification procedure, save the verification objects; you need them when you migrate to the next release of DB2.

The jobs that are listed in Table 56 are designed to run with minimal interaction on your part. However, before running these jobs, make any modifications that are suggested either in this chapter or in "Completing the CLIST processing" on page 243.

After running the verification jobs, you can still fall back. See "Falling back" on page 344 for details.

Table 56. Relationship of IVP phases to programs

Phase	Job	Program	Description
0	DSNTEJ0	DSNTIAD	Remove sample applications and sample schema authorizations
1	DSNTEJ1	DSNTIAD	Create tables
		DSN8CA	Assembler interface to call attach facility
		DSN8EAE1	Edit exit routine
		DSN8HUFF	Huffman compression exit routine
		DSNUTILB	Utilities
	DSNTEJ1L	DSNTEP2	Dynamic SQL program
	DSNTEJ1P	DSNTEP2	Dynamic SQL program
	DSNTEJ1S[1]	DSNHSP	See note 1
	DSNTEJ1T[2]	DSNUPROC	See note 2
	DSNTEJ1U	DSNTIAD	Create Unicode table
		DSNUTILB	Load Unicode table
		DSNTEP2	Select Unicode table
2	DSNTEJ2A	DSNTIAD	Grant execution
		DSNTIAUL	Unload and load tables
		DSNUTILB	Utilities
	DSNTEJ2C	DCLGEN	Generate declarations
		DSNTIAD	Grant execution
		DSN8BC3	COBOL phone application
	DSNTEJ2D	DSNTIAD	Grant execution
		DSN8BD3	C phone application
	DSNTEJ2E	DSN8MDG	Prepare error message routine
		DSN8BECL	Prepare classes used by C++ phone application
		DSNTIAD	Grant execution
		DSN8BE3	C++ phone application
	DSNTEJ2F	DSNTIAD	Grant execution
		DSN8BF3	Fortran phone application
	DSNTEJ2P	DCLGEN	Generate declarations
		DSNTIAD	Grant execution
		DSN8BP3	PL/I phone application

Table 56. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
	DSNTEJ2U[3]	DSNTIAD	Register sample user-defined functions
		DSN8DUAD	C program for ALTDATE function (current date)
		DSN8DUCD	C program for ALTDATE function (given date)
		DSN8DUAT	C program for ALTIME function (current time).
		DSN8DUCT	C program for ALTTIME function (given time)
		DSN8DUCY	C program for CURRENCY function
		DSN8DUTI	C program for TABLE_NAME, TABLE_SCHEMA, and TABLE_LOCATION functions
		DSN8EUDN	C++ program for DAYNAME function
		DSN8EUMN	C++ program for MONTHNAME function
		DSNTEP2	Dynamic SQL program
		DSN8DUWF	User-defined table function sample
		DSN8DUWC	C program on client for user-defined table function sample
3		SPUFI	Sample SPUFI input
	DSNTEJ3C	DSNTIAD	Grant execution
		DSN8CC	COBOL interface to call attachment facility
		DSN8SCM	COBOL connection manager
		DSN8SC3	COBOL phone application
		DSN8HC3	COBOL organization application
	DSNTEJ3M	DSNTIAD	Drop sample MQT objects, create sample MQS objects, grant authority to MQT objects, and populate sample MQT tables
		DSNUTILB	Produce statistics for MQT
		DSNTEP2	Process EXPLAINS on sample MQT tables
	DSNTEJ3P	DSNTEP2	Dynamic SQL application
		DSNTIAD	Grant execution
		DSN8SPM	PL/I connection manager
		DSN8SP3	PL/I phone application
4	DSNTEJ4C	DSNTIAD	Grant execution
		DSN8IC*x*	Organization application
		DSN8MC*x*	Copy code
	DSNTEJ4P	DSNTIAD	Grant execution

Table 56. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
		DSN8IP*x*	Organization, project applications
		DSN8MP*x*	Copy code
5	DSNTEJ5A	DSNTIAC	CICS SQLCA formatter front-end
	DSNTEJ5C	DSNTIAD	Grant execution
		DSN8CC*x*	Organization application
		DSN8MC*x*	Copy code
	DSNTEJ5P	DSNTIAD	Grant execution
		DSN8CP*x*	Organization, project applications
		DSN8MP*x*	Copy code
6	DSNTEJ6	DSNTIAD	Update location column in the department table to the sample location entered at installation time
	DSNTEJ6D[4]	DSN8ED1	Compile, link-edit, bind, and run a sample application program that calls a stored procedure
	DSNTEJ6T[4]	DSN8ED2	Register, prepare, and bind the stored procedure sample application
	DSNTEJ6P[5]	DSN8EP1	Invoke the sample stored procedure
	DSNTEJ6R[7]	DSN8ED8	Compile, link-edit, bind, and run a sample application that calls DSNUTILU, the DB2 Utilities Unicode parser stored procedure
	DSNTEJ6S[5]	DSN8EP2	Create sample stored procedure
	DSNTEJ6U[6]	DSN8EPU	Compile, link-edit, bind, and run a sample application that calls DSNUTILS, a stored procedure for executing DB2 Utilities
	DSNTEJ6V[7]	DSN8EE0	C++ class that calls DSNTIAR
		DSN8EE1	C++ class that calls DSNUTILS
		DSN8EE2	C++ client for DSN8EE1
	DSNTEJ6W[7]	DSN8ED6	Calls the WLM_REFRESH stored procedure
	DSNTEJ6Z[11]	DSN8ED7	C class that calls DSNWZP stored procedure
	DSNTEJ61[8]	DSN8EC1	Create sample ODBA stored procedure
	DSNTEJ62[8]	DSN8EC2	Invoke sample ODBA stored procedure
	DSNTEJ63[9]	DSN8ES1	Run sample SQL procedure that calculates employee earnings.
	DSNTEJ64[9]	DSN8ED3	Call the sample SQL procedure, DSN8ES1 from a client
	DSNTEJ65[9]	DSN8ED4	Call the SQL procedure processor, DSNTPSMP
		DSN8ES2	Run SQL procedure that calculates employee bonuses

Table 56. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description
		DSN8ED5	Call SQL procedure, DSN8ES2
7	DSNTEJ7	DSNTIAD	Create sample LOB table
		DSNTIAD	Create synonyms, grant access to LOB tables
		DSNUTILB	Load sample LOB table
		DSNUTILB	Produce statistics for LOB table spaces
	DSNTEJ71	DSNTIAD	Grant access to plans
		DSN8DLPL	Populate sample LOB table with BLOB data
		DSN8DLTC	Verify contents of LOB table
	DSNTEJ73	DSNTIAD	Grant access to plans
		DSN8DLRV	C employee resume application
	DSNTEJ75[10]	DSNTIAD	Grant access to plans
		DSN8DLPV	C employee photo application
	DSNTEJ76	DSN8CLPL	Create COBOL LOB sample table
		DSN8CLTC	Create synonyms, grant access to COBOL LOB tables
	DSNTEJ77	DSN8CLRV	Load sample COBOL LOB table
	DSNTEJ78	DSN8CLPV	Produce statistics for COBOL BLOB table spaces

Table 56. Relationship of IVP phases to programs (continued)

Phase	Job	Program	Description

Note:

1. Job DSNTEJ1S, which contains the sample JCL to run the schema processor, is not a part of the sample applications to verify installation. For more information about the schema processor, see Part 2 (Volume 1) of of *DB2 Administration Guide*.
2. Job DSNTEJ1T, which adds rows to SYSIBM.SYSSTRINGS for character conversion purposes, is not a part of the sample applications to verify installation.
3. Job DSNTEJ2U is not created unless you specify C/C++ for z/OS at installation time.
4. Jobs DSNTEJ6T and DSNTEJ6D are not edited by the CLIST unless the following conditions all apply:
   - A REMOTE LOCATION name is entered on panel DSNTIPY.
   - A LOCATION NAME is entered on panel DSNTIPR.
   - A non-blank value is specified for WLM ENVIRONMENT on panel DSNTIPX.
   - You specify AUTO or COMMAND for DDF STARTUP OPTION on panel DSNTIPR.
5. Jobs DSNTEJ6P and DSNTEJ6S are not edited by the CLIST unless the following conditions all apply:
   - A REMOTE LOCATION name is entered on panel DSNTIPY.
   - A LOCATION NAME is entered on panel DSNTIPR.
   - A non-blank value is specified for WLM ENVIRONMENT on panel DSNTIPX.
   - You specify AUTO or COMMAND for DDF STARTUP OPTION on panel DSNTIPR.
6. Jobs DSNTEJ6R and DSNTEJ6V are not edited by the CLIST unless you specify AUTO or COMMAND for the DDF STARTUP option on panel DSNTIPR and provide a value for WLM ENVIRONMENT on DSNTIPX.
7. Jobs DSNTEJ61 and DSNTEJ62 are not edited by the CLIST unless the following conditions all apply:
   - A REMOTE LOCATION name is entered on panel DSNTIPY.
   - A LOCATION NAME is entered on panel DSNTIPR.
   - A non-blank value is specified for WLM PROC NAME on panel DSNTIPX.
   - You specify AUTO or COMMAND for DDF STARTUP OPTION on panel DSNTIPR.
8. Jobs DSNTEJ63, DSNTEJ64, and DSNTEJ65 are not edited by the CLIST unless the following conditions all apply:
   - A LOCATION NAME is entered on panel DSNTIPR.
   - A non-blank value is specified for WLM ENVIRONMENT on panel DSNTIPX.
   - You specify AUTO or COMMAND for DDF STARTUP OPTION on panel DSNTIPR.
9. Jobs DSNTEJ75 and DSNTEJ78 are not edited by the CLIST unless the GDDM MACLIB and GDDM LOAD MODULES fields on panel DSNTIPW are non-blank.
10. Job DSNTEJ6Z is not edited by the CLIST unless a non-blank value is specified for WLM ENVIRONMENT on panel DSNTIPX, and you specify AUTO or COMMAND for DDF STARTUP OPTION on panel DSNTIPR.

Planning for verification

Before performing any of the verification phases, you must make certain decisions about your verification strategy. DB2 system administrators and system administrators for ISPF, TSO, batch, IMS, and CICS must be involved in these decisions. With these system administrators:

- Determine the verification phases that you plan to perform.

  Examine the description of each verification phase in this chapter, and determine which phases apply to your needs.

- Identify any phases that you want to modify before you perform them.

  Verification is designed to run with little interaction on your part. This chapter does not discuss how to modify any of the phases, but you can adapt any of the seven phases to your needs. If this is your intent, identify and describe any modifications you plan to make.
- Establish additional testing steps to complete the verification.

  The verification phases and the jobs that you run to perform them are valuable tools for testing DB2. They are not a substitute for a thorough subsystem test. You must plan and perform your own additional testing to complete the verification. To help you assess which additional tests might be necessary, examine the sample applications that are provided with DB2.
- Start any DB2 databases that are not currently started.

Special considerations for COBOL programs

IBM tested the DB2 COBOL samples with the compiler options that are shown in this section. If you have a problem executing the DB2 COBOL samples, ensure that your compiler options are consistent with the Enterprise COBOL options in Table 57. Remember that if you are using CICS, the options that you need to use depend on the CICS environment. To verify that you are using the correct options in your CICS environment, refer to *CICS Application Programming Guide*.

Table 57. Enterprise COBOL options

ADV	NOEXIT	NOTYPECHK
BUFSIZE(4096)	NOFASTSRT	NOVBREF
DATA(31)	NOFLAGMIG	NOWORD
FLAG(I)	NOFLAGSTD	NOXREF
INTDATE(ANSI)	NOIDLGEN	NUMPROC(NOPFD)
LANGUAGE(EN)	NOLIB	OBJECT
LINECOUNT(60)	NOLIST	OUTDD(SYSOUT)
NOADATA	NOMAP	PGMNAME(LONGUPPER)
NOAWO	NONAME	QUOTE
NOCMPR2	NONUMBER	RENT[3]
NOCOMPILE(S)	NOOFFSET	RMODE(AUTO)
NOCURRENCY	NOOPTIMIZE	SIZE(MAX)
NODBCS	NOSEQUENCE	SOURCE
NODECK	NOSSRANGE	SPACE(1)
NODUMP	NOTERM[1] or TERM[2]	TRUNC(STD)
NODYNAM	NOTEST	ZWB

Notes:

[1] Refers to jobs DSNTEJ2C, DSNTEJ3C, and DSNTEJ4C only.

[2] Refers to job DSNTEJ5C only.

[3] See the CICS documentation for actual options to use.

If you would like information on the language CLIST (DSNH), refer to the *DB2 Command Reference*.

For more detailed instructions, see *Enterprise COBOL for z/OS Programming Guide* and *z/OS Language Environment Programming Guide*.

Special considerations for C and C++ programs

IBM tested the DB2 C and C++ samples with the following compiler options. If you have a problem executing the DB2 C and C++ samples, ensure that your compiler options are consistent with the options in table Table 58 or Table 59.

Table 58. C language options

NOAGGR	NOEVENTS	NOHWOPTS	NOMEMORY	NOPPONLY
NOALIAS	EXECOPS	NOINLINE[3]	NESTINC(255)	REDIR[1]
ARGPARSE[1]	NOEXPMAC	LIST	OBJECT	NORENT
NOCHECKOUT[2]	NOEXPORTAL	NOLOCALE	NOOE[1]	NOSEARCH
NOCSECT	FLAG(I)	NOLONGNAME	NOOFFSET	NOSHOWINC
NODECK	NOGONUMBER	NOLSEARCH	NOOPTIMIZE	SOURCE
NODLL	HALT(16)[1]	MAXMEM(2000)[4]	PLIST(HOST[1]	SPILL(128)[4]
NOSSCOMM	START	TARGET(LE)	TERMINAL	NOTEST[5]
NOUPCONV	XREF			

Notes:

[1] This option is used by IBM C/C++ for z/OS.
[2] NOPPTRACE, PPCHECK, GOTO, ACCURACY, PARM, NOENUM, NOEXTERN, TRUNC, INIT, NOPORT, GENERAL.
[3] AUTO, NOREPORT, 100, 1000.
[4] This option is used only by IBM AD/Cycle C/370 V1R2.
[5] SYM, BLOCK, LINE, NOPATH.

Table 59 contains the C++ language options.

Table 59. C++ language options

ARGPARSE	NOIDL[1]	NESTINC(255)[2]	NOSHOWINC	TERMINAL
NOATTRIBUTE	NOINFO	OBJECT	NOSOM	NOTEST
NOCSECT	NOINLRPT	NOOE[2]	SOMEINIT	XREF
EXECOPS	LANGLVL(EXTENDED)	NOOFFSET	NOSOMGS	HALT(16)
NOEXPMAC	NOLIST	OPTIMIZE(0)	SOURCE	MEMORY[2]
NOEXPORTALL	NOLOCALE[2]	PLIST(HOST)[2]	NOSRCMSG	NOSEQUENCE
FLAG(I)	LONGNAME	NOPPONLY	START[2]	TEMPINC
NOGONUMBER	MARGINS	REDIR	TARGET(LE)[2]	

Notes:

[1] This option is used by IBM C/C++ for MVS/ESA V3R1 and subsequent releases.
[2] This option is used by IBM C/C++ for z/OS and subsequent releases.

The installation CLIST customizes C++ compiler parameters in sample job DSNTEJ2E if you have specified C++ for MVS/ESA V3R2 or a subsequent release on panel DSNTIPU.

If you want to run your samples applications with C++ for z/OS, but you selected an earlier version of that language on the installation panel, make the changes shown in Table 60.

Table 60. Required changes to run samples applications with C++ for MVS/ESA V3R2 when an earlier version of C++ is selected on the installation panel

In sample	Change all occurrences of	To
Job DSNTEJ2E	`PARM.CP='SOURCE XREF,MARGINS',`	`PARM.CP='/CXX SOURCE XREF,MARGINS',`
Procedure DSNHCPP2	`//CP EXEC PGR=CBC320PP, COND=((4,LT,PC1),(4,LT,PC2)), REGION=4096K`	`//CP EXEC PGR=CBC320PP, COND=((4,LT,PC1),(4,LT,PC2)), REGION=4096K,PARM='/CXX'`

Special considerations for PL/I programs

IBM tested the DB2 PL/I samples with the compiler options that are shown in Table 61. If you have a problem executing the DB2 PL/I samples, ensure that your compiler options are consistent with these options.

Table 61. PL/I options

CHARSET(60,EBCDIC)	LINECOUNT(55)	LMESSAGE	MARGINS(2,72,0)	NOAGGREGATE
NOATTRIBUTES	NOCOMPILE(S)	NOCOUNT	NOESD	NOFLOW
NOGONUMBER	NOGOSTMT	NOGRAPHIC	NOIMPRECISE	NOINCLUDE
NOINTERRUPT	NOLIST	NOMAP	NOMARGINI	NONEST
NONUMBER	NOOFFSET	NOOPTIMIZE	NOSTORAGE	NOSYNTAX(S)
NOTERMINAL	NOXREF	OBJECT	ODECK	SEQUENCE(73,80)
SIZE(506756)	SOURCE	STMT		

The installation CLIST tailors the PL/I sample programs for the following compilers, depending on the values that you entered in the PL/I COMPILER MODULE field on panel DSNTIPU:

- Enterprise PL/I
- IBM PL/I for MVS & VM

To use IMS from Enterprise PL/I or IBM PL/I for MVS & VM, you must use the SYSTEM(IMS) compile-time option when compiling your application program. In addition, you must also specify entry point PLISTART when your application is link-edited. For more information, see *PL/I for MVS & VM Programming Guide* or *IBM PL/I MVS & VM Programming Guide*.

Phase 0: Deleting the sample objects (DSNTEJ0)

Phase 0 consists of one job, DSNTEJ0. It frees all plans, drops all objects, and deletes data sets so that Phase 1 can be run again. Run Phase 0 (job DSNTEJ0) only if you want to remove all the verification processing that you have done so far so that you can begin the verification procedure again. When you complete the verification procedure, save the verification objects; you need them when you migrate to the next release of DB2.

If a sample application abends while running a utility, ensure that the utility is terminated before attempting to rerun the job. For information about the -TERM UTILITY command, see *DB2 Command Reference*.

Even when DSNTEJ0 runs successfully, some of the FREE, DROP, and DELETE commands often fail because the object was not created earlier. You can ignore these errors, even though they might generate return codes of 8 or 12. Check other errors.

If DSNTEJ0 runs successfully, it produces the return codes that are shown in Table 62.

Table 62. DSNTEJ0 return codes

Step	PROCSTEP	Return code
PH00S01		0000, 0008, or 0012
PH00S02		0000, 0004, or 0008
PH00S03	DSNUPROC	0004 or 0008
PH00S04		0000, 0004, or 0008

Table 62. DSNTEJ0 return codes (continued)

Step	PROCSTEP	Return code
PH00S05		0000, 0004, or 0008
PH00S06		0000, 0004, or 0008
PH00S07		0000, 0004, or 0008

If this job fails or abends, ensure that the user that is specified on the JOB statement is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups. Then correct any other problems, and rerun the job from the last successful step.

If the subsystem data sets were deleted before the DB2 sample objects are deleted, you must delete the data sets by using access method services commands or TSO commands. In all of the following examples, *vcatalog* is the catalog alias name that you specified for the CATALOG ALIAS field on installation panel DSNTIPA2. The *y* is either I or J.

The following access method services commands, which can be executed under TSO, delete the Version 8 sample data sets:

```
DELETE 'vcatalog.DSNDBD.DSN8D81A.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81L.*.y0001.*'
DELETE 'vcatalog.DSNDBD.DSN8D81P.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81U.*.y0001.*'
DELETE 'vcatalog.DSNDB04.STAFF.y0001.A001'
DELETE 'vcatalog.DSNDB04.TESTSTUF.y0001.A001'
DELETE 'vcatalog.DSNDBC.DSN8D81E.*.y0001.*'
DELETE 'vcatalog.DSNDBC.DSN8D81Y.*.y0001.*'
```

Phase 1: Creating and loading sample tables

This phase consists of four jobs: DSNTEJ1, DSNTEJ1L, DSNTEJ1P, and DSNTEJ1U. DSNTEJ1 invokes program DSNTIAD, which creates objects during the verification procedure. Run DSNTIEJ1 before running any other sample jobs.

DSNTEJ1L and DSNTEJ1P prepare and invoke program DSNTEP2, which lists the contents of the sample tables. The difference between the jobs is that DSNTEJ1P requires the PL/I compiler and allows you to customize DSNTEP2.

DSNTEJ1U creates, populates, and tests a sample Unicode database.

Job DSNTEJ1

Job DSNTEJ1 consists of the steps that are listed in Table 63.

Table 63. Steps in job DSNTEJ1

Step	Function
1-4	Creates all objects (storage group, databases, table spaces, tables, indexes, and views) that are used by the samples.
5	Drops synonyms.

Table 63. Steps in job DSNTEJ1 (continued)

Step	Function
6	Creates synonyms and grants authorization on objects to PUBLIC AT ALL LOCATIONS. This step creates synonyms for the sample tables, indexes, and views, so that the currently running authorization ID can execute the sample application and grant appropriate authority. The sample dynamic SQL program DSNTIAD processes the DB2 object definitions in this step and several others.
7	Uses the ASMCL procedure to create DSN8EAE1, an edit exit routine.
8	Assembles and link-edits DSNHUFF.
9	Assembles and link-edits DSN8FPRC, a sample field procedure.
10	Prepares the sample call attachment facility assembler interface. You must link-edit ISPLINK, the ISPF interface module, with this CAF sample load module. To do this, ensure that the link-edit SYSLIB statement that retrieves the ISPF load module library in procedure DSNHASM is not commented out.
11	Creates the sample utility list.
12	Loads the programming-related tables by using the LOAD utility.
13	Loads the sample tables by using the LOAD utility.
14	Checks data for referential integrity.
15	Establishes a quiesce point by using both log and image copies.
16	Makes an image copy of all the sample tables by using the COPY utility.
17	Establishes another quiesce point by using only image copies.
18	Reorganizes a table space and compiles statistics on all table spaces by using the REORG and RUNSTATS utilities.
19	Performs a REORG TABLESPACE with SHRLEVEL CHANGE.
20	Loads the sample tables by using the LOAD utility.
21	Sets the CURRENT RULES register and adds a check constraint using ALTER TABLE.
22	Checks data for referential integrity.
23	Checks data for check integrity.
24-27	Performs the operations in steps 15-18 except for the REORG on partition 3 of the Employee table space.
28	Unloads data from a partitioned table.
29	Reduces the partition key on the fourth partition of table space DSN8S81E.
30	Processes an online schema change to add a fifth partition to table space DSN8S81E.
31	Reorganizes a partitioned table space.
32	Performs various online schema changes, including lengthening a character field and converting an integer field to a decimal field.

If DSNTEJ1 runs successfully, it produces the return codes that are shown in Table 64 on page 375.

Table 64. DSNTEJ1 return codes

Step	PROCSTEP	Return code
PH01S01		0000
PH01S02		0000
PH01S03		0000
PH01S04		0000
PH01S05		0000
PH01S06		0000
PH01S07	ASM	0000
	LKED	0000
PH01S08	ASM	0000
	LKED	0000
PH01S09	ASM	0000
	LKED	0000
PH01S10	PC	0004
	ASM	0000
	LKED	0000
PH01S11	IEBGENER	0000
PH01S12	DSNUPROC	0000
PH01S13	DSNUPROC	0004
PH01S14	DSNUPROC	0000
PH01S15	DSNUPROC	0000
PH01S16	DSNUPROC	0000
PH01S17	DSNUPROC	0000
PH01S18	DSNUPROC	0000 or 0004
PH01S19	DSNUPROC	0000
PH01S20	DSNUPROC	0004
PH01S21		0004
PH01S22	DSNUPROC	0004
PH01S23		0000
PH01S24	DSNUPROC	0000
PH01S25	DSNUPROC	0000
PH01S26	DSNUPROC	0000
PH01S27	DSNUPROC	0000
PH01S28	DSNUPROC	0000
PH01S29		0000 or 0004
PH01S30		0000
PH01S31	DSNUPROC	0004
PH01S32		0000

DB2 issues the following message for every SQL statement, except for the drop synonym and insert statements:

```
DSNT400I SQLCODE = 0, SUCCESSFUL EXECUTION
```

If the synonyms in the DROP SYNONYM statements are not defined, SQL return codes of -204 result. The INSERT statements violate a check constraint on the EMP table. This results in an SQL return code of -545.

Job DSNTEJ1L

DSNTEJ1L link-edits the DSNTEP2 object deck (DSNTEP2L) to create an executable load module DSNTEP2. DSNTEP2 is described in Appendix D of *DB2 Utility Guide and Reference*. DSNTEJ1L link-edits the DSNTEP4 object deck (DSNTEP4L) to create an executable load module DSNTEP4.

DSNTEJ1L also binds and runs programs DSNTEP2 and DSNTEP4. DSNTEP2 lists the sample database tables and views. DSNTEP2 is a dynamic PL/I program that accepts SQL statements. DSNTEP2 produces a listing of the results of SELECT statements. DSNTEP4 is identical to DSNTEP2, except that it uses multi-row fetch

Job DSNTEJ1L requires the Language Environment link-edit and run-time libraries. DSNTEJ1L does not require the PL/I compiler.

If DSNTEJ1L runs successfully, it produces the return codes that are shown in Table 65.

Table 65. DSNTEJ1L return codes

Step	PROCSTEP	Return code
PH01PS01		0000
PH01PS02		0000 or 0004
PH01PS03		0000
PH01PS04		0000 or 0004

You can compare the output from this job with the sample output for DSNTEJ1L, which is found in member DSN8TJ1L in your *prefix*.SDSNIVPD data set.

If you run DSNTEJ1P before DSNTEJ1L, you can expect step PH01PS02 of job DSNTEJ1L to produce a return code of 0004 and the following message:

```
SQLWARNING ON GRANT   COMMAND, EXECUTE FUNCTION
  RESULT OF SQL STATEMENT:
  DSNT404I SQLCODE = 562, WARNING:  A GRANT OF A PRIVILEGE WAS IGNORED
           BECAUSE THE GRANTEE ALREADY HAS THE PRIVILEGE FROM THE GRANTOR
```

If either DSNTEJ1 or DSNTEJ1L fails or abends, ensure that the user that is specified in the JOB statements is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups.

Then, correct any other problems. Before rerunning DSNTEJ1, run DSNTEJ0 to drop the sample data. If you rerun DSNTEJ1L, rerun it from the last successful step.

Job DSNTEJ1P

If you have run DSNTEJ1L, you do not need to run DSNTEJ1P because they produce the same results. The major difference is that DSNTEJ1P uses the PL/I compiler and allows you to customize DSNTEP2 and DSNTEP4. DSNTEP2 and DSNTEP4 are described in Appendix D of *DB2 Utility Guide and Reference*. DSNTEJ1P precompiles, compiles, and link-edits PL/I program DSNTEP2. This program then lists the sample database tables and views. It is a dynamic PL/I program that accepts SQL statements. It produces a listing of the results of SELECT statements.

If DSNTEJ1P runs successfully, it produces the return codes that are shown in Table 66.

Table 66. DSNTEJ1P return codes

Step	PROCSTEP	Return code
PH01PS01	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0000
PH01PS02		0000 or 0004
PH01PS03	PPLI	0000
	PC	0000
	PLI	0004
	PLKED	0004
	LKED	0000
PH01PS04		0000 or 0004

You can compare the output from this job with the sample output for DSNTEJ1P, which is found in member DSN8TJ1P in your *prefix*.SDSNIVPD data set.

If you run DSNTEJ1L before DSNTEJ1P, you can expect step PH01PS02 of job DSNTEJ1P to produce a return code of 0004 and the following message:

```
SQLWARNING ON GRANT   COMMAND, EXECUTE FUNCTION
  RESULT OF SQL STATEMENT:
  DSNT404I SQLCODE = 562, WARNING:  A GRANT OF A PRIVILEGE WAS IGNORED
           BECAUSE THE GRANTEE ALREADY HAS THE PRIVILEGE FROM THE GRANTOR
```

If either DSNTEJ1 or DSNTEJ1P fails or abends, ensure that the user that is specified in the JOB statements is an authorized ID. If the name that you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP is a primary authorization ID, use this name. If the sample authorization exit routine and RACF are installed, and if the SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups.

Then, correct any other problems. Before rerunning DSNTEJ1, run DSNTEJ0 to drop the sample data. If you rerun DSNTEJ1P, rerun it from the last successful step.

Job DSNTEJ1U

DSNTEJ1U creates a database, table space, and table with CCSID Unicode. DSNTEJ1U loads data into the table from a data set that contains a full range of characters in an EBCDIC Latin-1 code page, which results in a mix of single and double-byte characters in the Unicode table. It then runs DSN1PRNT on the table to dump the hex image, revealing which characters are stored in single bytes and which in double bytes.

If DSNTEJ1U runs successfully, it produces the return codes that are shown in Table 67 on page 378.

Table 67. DSNTEJ1U return codes

Step	PROCSTEP	Return code
PH01US01		0000
PH01US02		0000
PH01US03	DSNUPROC	0000
PH01US04		0000

Phase 2: Testing the batch environment

This phase consists of several jobs. Run the jobs to test the program preparation procedures for various languages.

If any of the Phase 2 jobs fail or abend, be sure that the user specified in the JOB statements is authorized. Use the name you specified for either the SYSTEM ADMIN 1 option or the SYSTEM ADMIN 2 option on installation panel DSNTIPP. Then correct any other problems, and rerun the jobs from the last successful step.

Job DSNTEJ2A

DSNTEJ2A tests the assembler program preparation procedures. This job prepares and invokes program DSNTIAUL, which demonstrates the use of dynamic SQL in assembler to unload the data from tables or views. It also generates LOAD utility statements so the data can be loaded into another table. DSNTEJ2A then uses the LOAD utility to put data into copies of the unloaded tables. DSNTIAUL is described in Appendix D of *DB2 Utility Guide and Reference*.

If DSNTEJ2C runs successfully, it produces the return codes that are shown in Table 68.

Table 68. DSNTEJ2A return codes

Step	PROCSTEP	Return code
PREPUNL	PC ASM LKED	0000 0000 or 0004 0000 or 0004
BINDUNL		0000
DELETE		0000
CREATE		0000
UNLOAD		0000
EDIT		0000
LOAD	DSNUPROC	0004

You can compare the output from this job with the sample output for DSNTEJ2A found in member DSN8TJ2A in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2C

Job DSNTEJ2C tests the COBOL program preparation procedures. This job runs the phone application.

The phone application processes a table of telephone numbers, executing various types of SELECT statements and producing the corresponding listings. It can also update a phone number. This program is discussed in detail in “The phone application scenario” on page 420.

To run DSNTEJ2C with a type of COBOL other than the one you selected on panel DSNTIPY, see "Special considerations for COBOL programs" on page 370. Be aware that you must use the same type of COBOL to prepare DSNTEJ2C, DSNTEJ3C, DSNTEJ4C, AND DSNTEJ5C.

If DSNTEJ2C runs successfully, it produces the return codes that are shown in Table 69.

Table 69. DSNTEJ2C return codes

Step	PROCSTEP	Return code
PH02CS01		0000
PH02CS02	PC	0004
	COB	0000 or 0004
	PLKED[1]	0004
	LKED	0000 or 0004
PH02CS03	PC	0000
	COB	0000 or 0004
	PLKED[1]	0004
	LKED	0000
PH02CS04		0000

You can compare the output from this job with the sample output for DSNTEJ2C found in member DSN8TJ2C in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2D

Job DSNTEJ2D tests the C program preparation procedures. You must have sequence numbering on to run this job from an ISPF session. The C job runs only the phone application. See the phone application description under DSNTEJ2C on page 378.

If DSNTEJ2D runs successfully, it produces the return codes that are shown in Table 70.

Table 70. DSNTE2D return codes

Step	PROCSTEP	Return code
PH02DS01	PC	0004
	C	0000
	PLKED	0000 or 0004
	LKED	0004
PH02DS02	PC	0000
	C	0000
	PLKED	0000 or 0004
	LKED	0000 or 0004
PH02DS03		0000

You can compare the output from this job with the sample output for DSNTEJ2D found in member DSN8TJ2D in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2E

Job DSNTEJ2E tests the C++ program preparation procedures. You must have sequence numbering on to run this job from an ISPF session. The C++ job runs only the phone application, "Job DSNTEJ2C" on page 378.

If DSNTEJ2E runs successfully, it produces the return codes that are shown in Table 71 on page 380.

Table 71. DSNTE2E return codes

Step	PROCSTEP	Return code
PH02ES01	PC	0004
	C	0000
	PLKED	0000 or 0004
	LKED	0004
PH02ES02	PC	0000
	CP	0000
	PLKED	0000 or 0004
	LKED	0004
PH02ES03	PC	0004
	CP	0000
	PLKED	0000 or 0004
	LKED	0000
PH02ES04		0000

You can compare the output from this job with the sample output for DSNTEJ2E found in member DSN8TJ2E in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2F

Job DSNTEJ2F tests the Fortran program preparation procedures. The FORTRAN job runs only the phone application. See the phone application description under DSNTEJ2C on page 378.

If DSNTEJ2F runs successfully, it produces the return codes that are shown in Table 72.

Table 72. DSNTE2F return codes

Step	PROCSTEP	Return code
PH02FS01	PC	0004
	ASM	0000
	LKED	0004
PH02FS02	PC	0000
	FORT	0000
	LKED	0000
PH02FS03		0000

You can compare the output from this job with the sample output for DSNTEJ2F found in member DSN8TJ2F in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2P

Job DSNTEJ2P tests the PL/I program preparation procedures. The PL/I job runs the phone application.

If DSNTEJ2P runs successfully, it produces the return codes that are shown in Table 73.

Table 73. DSNTEJ2P return codes

Step	PROCSTEP	Return code
PH02PS01		0000
PH02PS02	PPLI	0000
	PC	0004
	PLI	0004
	LKED	0004

Table 73. DSNTEJ2P return codes (continued)

Step	PROCSTEP	Return code
PH02PS03	PPLI PC PLI LKED	0000 0000 0004 0000
PH02PS04		0000
PH02PS05		0000

You can compare the output from this job with the sample output for DSNTEJ2P found in member DSN8TJ2P in your *prefix*.SDSNIVPD data set.

Job DSNTEJ2U

DSNTEJ2U prepares and tests several sample user-defined functions, as well as a driver program to exercise them.

In order for the Install CLIST to generate Job DSNTEJ2U, you must complete the following steps during installation:

- Specify that the host has access to C/C++ for z/OS on Install Panel DSNTIPU
- Specify the name of the default WLM environment on install panel DSNTIPX

The sample user-defined functions are:

Function	Description
ALTDATE	Returns the current date in a user-specified format or converts a user-specified date from one format to another.
ALTIME	Returns the current time in a user-specified format or converts a user-specified time from one format to another.
CURRENCY	Formats a floating point number as a currency value.
DAYNAME	Returns the day of the week for a user-specified date in ISO format.
MONTHNAME	Returns the month for a user-specified date in ISO format.
TABLE_LOCATION	Returns the location name of a table, view, or undefined object found after resolving aliases for a user-specified object. See *DB2 SQL Reference* for more information.
TABLE_NAME	Returns the name of a table, view, or undefined object found after resolving aliases for a user-specified object. See *DB2 SQL Reference* for more information.
TABLE_SCHEMA	Returns the schema name of a table, view, or undefined object found after resolving aliases for a user-specified object. See *DB2 SQL Reference* for more information.
WEATHER	Returns sample weather data obtained from a TSO data set by way of demonstrating the usefulness of a user-defined function table function.

For more information about user-defined functions, see Part 2 of *DB2 Application Programming and SQL Guide*.

If you do not have C++ installed, skip steps PH02US08 and PH02US09. Also remove all statements that refer to DAYNAME and MONTHNAME from part DSNTESU in the *prefix*.SDSNSAMP library.

Job DSNTEJ2U consists of the steps that are listed in Table 75.

Table 74. Steps in job DSNTEJ2U

Step	Function
1	Drops all specific sample user-defined functions.
2	Creates and registers all sample user-defined functions. Grants EXECUTE authority to PUBLIC for the sample user-defined functions.
3-10	Prepares the eight external programs used by specific user-defined functions.
11	Binds the package for the TABLE_NAME, TABLE_SCHEMA, and TABLE_LOCATION functions. These are the only sample functions that issue SQL statements. This step also grants EXECUTE authority on these two samples to PUBLIC.
12	Invokes DSNTEP2 to exercise the sample user-defined functions.
13	Prepares DSN8DUWF, the external module for the sample user defined table function, WEATHER.
14	Prepares DSN8DUWC, a sample client function for statically invoking the WEATHER user-defined table function.
15	Binds the package and plan for DSN8DUWC and grants the necessary authorities.
16	Invokes DSN8DUWC.

If DSNTEJ2U runs successfully, it produces the return codes that are shown in Table 75.

Table 75. DSNTEJ2U return codes

Step	PROCSTEP	Return code
PHO2US01		0000
PH02US02		0000 or 0004
PH02US03 - PH02US07	PC C PLKED LKED	0004 0000 0004 0000
PH02US08 - PH02US09	PC CP PLKED LKED	0004 0000 0004 0000
PH02US10	PC C PLKED LKED	0000 0000 0004 0000
PH02US11		0000
PH02US12		0004

Table 75. DSNTEJ2U return codes (continued)

Step	PROCSTEP	Return code
PH02US13	PC C PLKED LKED	0004 0000 0004 0000
PH02US14	PC C PLKED LKED	0000 0000 0004 0000
PH02US15		0000 or 0004
PH02US16		0000

You can compare the output from this job with the sample output for DSNTEJ2U found in member DSN8TJ2U in your *prefix*.SDSNIVPD data set.

Phase 3: Testing SPUFI, DRDA Access, dynamic SQL, and TSO

Phase 3 allows you to test SPUFI and DRDA access, run dynamic SQL statements, run the phone application in TSO, and bind packages at the local and remote locations.

DSNTESC, the sample PLAN_TABLE, creates a Version 8 PLAN_TABLE with an
index optimized for access path hints. It also demonstrates how to migrate your
PLAN_TABLEs from earlier releases of DB2. If you are migrating from Version 7,
DSNTESC is customized by the DB2 installation CLIST to migrate your Version 7
sample PLAN_TABLE to Version 8. Migration of the sample PLAN_TABLE from a
prior release should not be performed more than once. DSNTESC also includes
SQL to create and migrate a sample DSN_STATEMNT_TABLE and a sample
DSN_FUNCTION_TABLE.

SPUFI (SQL Processor Using File Input) is a facility of DB2I. “Testing SPUFI” provides instructions for testing it. You can only run SPUFI under ISPF. You can run dynamic SQL whether or not you have ISPF.

Testing SPUFI

You can test SPUFI by following the steps below:

1. Log on to TSO.
2. Enter ISPF (this might be done for you, depending on your site’s standard practice).
3. On the DB2I defaults panel, change the DB2 name to the DB2 subsystem name you entered on panel DSNTIPM during installation. Then select DB2I on the ISPF Primary Option Menu.
4. Select SPUFI on the DB2I menu.
5. Enter the library name ’*prefix*.NEW.SDSNSAMP(DSNTESA)’ as input to SPUFI on line 1, the DATASET NAME parameter. If your site uses the comma as a decimal point, the library name entered must be for the tailored version of job DSNTESA that was modified by the installation CLIST.
6. Define an output data set name on line 4, the output DATASET NAME parameter of the panel. This allows you to review the output.
7. Press ENTER, and examine the results. These SQL statements require a significant amount of DB2 processing; you could have to wait for the output.

Run steps 5, 6, and 7 three times:

- Once with member DSNTESA, which uses a set of SQL statements to create a short-lived table space and table (as discussed in "Dynamic SQL statements (DSNTESA, DSNTESQ)" on page 435)
- Once with member DSNTESC, which creates a table space and creates a PLAN_TABLE, DSN_FUNCTION_TABLE, and DSN_STATEMNT_TABLE in that table space (as discussed in Part 5 (Volume 2) of *DB2 Administration Guide*). If migrating from DB2 Version 7, the installation CLIST tailors DSNTESC to migrate your Version 7 PLAN_TABLE to Version 8 using a schema name of DSN8710. If your Version 7 PLAN_TABLE has a different schema name, edit DSNTESC to use your schema name.
- Once with member DSNTESE, which retrieves the EXPLAIN information.

If any step fails or abends, be sure that the DB2 subsystem name is specified in the DB2 NAME field on the DB2I Defaults panel.

If you must drop either a Version 7 or Version 8 PLAN_TABLE, remove the appropriate comments from the job to issue the DROP statements.

Also, make sure that the user ID you are using is authorized. If the name you specified for either SYSTEM ADMIN 1 or SYSTEM ADMIN 2 on installation panel DSNTIPP is a primary authorization ID, use this name. If the sample authorization exit and RACF are installed, and both SYSTEM ADMIN 1 and SYSTEM ADMIN 2 are known to DB2 as secondary authorization IDs, you can run these jobs under a user ID in either of these RACF groups. Then correct any other problems and rerun the scenario from the last successful step.

Running dynamic SQL and the ISPF/CAF application

The Phase 3 jobs install the ISPF/CAF sample application. This sample consists of an assembler or COBOL call attachment facility (CAF) interface, a connection manager program, the phone application, and the distributed application using DRDA access. Job DSNTEJ1 prepares the assembler interface, and job DSNTEJ3C prepares the COBOL interface. The connection manager program and the phone application each exist in COBOL and PL/I. Job DSNTEJ3C prepares the COBOL version; job DSNTEJ3P prepares the PL/I version. The distributed application using DRDA access is written in COBOL.

To run DSNTEJ3C with a type of COBOL other than the one you specified on field
2 of panel DSNTIPY, see "Special considerations for COBOL programs" on page
370. Remember that you must use the same type of COBOL to prepare DSNTEJ2C,
DSNTEJ3C, DSNTEJ4C, and DSNTEJ5C.

Jobs DSNTEJ3C and DSNTEJ3P

To prepare for the distributed sample application, DSNTEJ3C binds a package at the local and remote subsystems. The remote subsystem is at the location specified on installation panel DSNTIPY. This allows you to access data at either site. Both the local and remote systems must be running DB2 Version 8.

Because DSNTEJ3C does a remote bind, you must set up your local and remote systems for remote communication before running this job. The sample jobs DSNTEJ1 and DSNTEJ6 must have been run on the remote system. For concurrent installations at 2 DB2 locations, designate one location as the requester and the other location as the server. For information on how to set up DB2 for remote communication, see Chapter 12, “Connecting distributed database systems,” on page 449.

If DSNTEJ3C runs successfully, it produces the return codes that are shown in Table 76.

Table 76. DSNTEJ3C return codes

Step	PROCSTEP	Return code
PH03CS01	PC	0004
	COB	0000 or 0004
	PLKED[1]	0004
	LKED	0000
PH03CS02	PC	0004
	COB	0000 or 0004
	PLKED[1]	0004
	LKED	0000
PH03CS03	PC	0000
	COB	0004
	PLKED[1]	0004
	LKED	0000
PH03CS04	PC	0000 or 0004
	COB	0000 or 0004
	PLKED[1]	0004
	LKED	0000
PH03CS05		0000
PH03CS06		0000 or 0004

Step PH03CS06 can give a return code of 0004 if sample job DSNTEJ1 was not run on the remote system. For testing, you should run job DSNTEJ1 on the remote system. If DSNTEJ3P runs successfully, it produces the return codes that are shown in Table 77.

Table 77. DSNTEJ3P return codes

Step	PROCSTEP	Return code
PH03PS01	PPLI	0000
	PC	0004
	PLI	0000
	LKED	0000
PH03PS02	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0000
PH03PS03		0000

The three steps in job DSNTEJ3P, steps PH03PS01, PH03PS02, and PH03PS03, prepare the ISPF/CAF sample application.

Job DSNTEJ3M

Job DSNTEJ3M creates, populates, and processes a database that demonstrates the use of DB2 materialized query tables (MQTs).

Before you run job DSNTEJ3M, you must create a PLAN_TABLE. To create this table, specify the member DSNTESC as an input data set name to SPUFI.

If DSNTEJ3M runs successfully, it produces the return codes that are shown in Table 78.

Table 78. DSNTEJ3M return codes

Step	PROCSTEP	Return code
PH03MS00		0000
PH03MS01		0000
PH03MS02		0000
PH03MS03		0000 or 0004
PH03MS04		0000
PH03MS05		0000
PH03MS06		0000
PH03MS07		0000

Starting an application in an ISPF/TSO environment

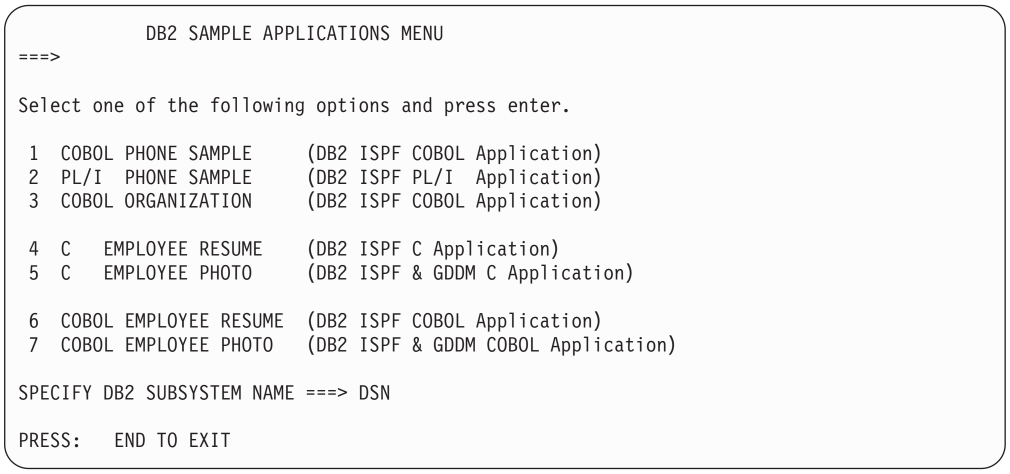
You must have access to ISPF load module libraries to run the ISPF/CAF sample application. See "Make panels, messages, and load modules available to ISPF and TSO" on page 265 for more information on this procedure. To start the application, enter a CALL command for option 6 of the ISPF primary option menu. To start the COBOL phone sample version of the connection manager, enter:

```
CALL 'prefix.RUNLIB.LOAD(DSN8SCM)'
```

To start the PL/I phone sample version of the connection manager, enter:

```
CALL 'prefix.RUNLIB.LOAD(DSN8SPM)'
```

After you enter one of these commands, DB2 displays the sample applications panel, shown in Figure 47.



```
            DB2 SAMPLE APPLICATIONS MENU
===>

Select one of the following options and press enter.

 1  COBOL PHONE SAMPLE     (DB2 ISPF COBOL Application)
 2  PL/I  PHONE SAMPLE     (DB2 ISPF PL/I  Application)
 3  COBOL ORGANIZATION     (DB2 ISPF COBOL Application)

 4  C   EMPLOYEE RESUME    (DB2 ISPF C Application)
 5  C   EMPLOYEE PHOTO     (DB2 ISPF & GDDM C Application)

 6  COBOL EMPLOYEE RESUME  (DB2 ISPF COBOL Application)
 7  COBOL EMPLOYEE PHOTO   (DB2 ISPF & GDDM COBOL Application)

SPECIFY DB2 SUBSYSTEM NAME ===> DSN

PRESS:   END TO EXIT
```

Figure 47. Initial panel of the ISPF/CAF application

Choosing option 1 or 2 on the sample applications panel during Phase 3 invokes either the COBOL or the PL/I version of the phone application. For more information about the phone application, see "The phone application scenario" on page 420. Choosing option 3 on the sample applications panel during Phase 6 invokes the COBOL organization application, which uses DRDA access to distributed data. For more information, see "The distributed organization application scenario" on page 423.

Choosing options 4, 5, 6, and 7 on the sample applications panel during Phase 7 invokes the ″Employee Resume″ and ″Employee Photo″ applications, which processes LOB data. Options 4 and 5 access the C language sample applications, and options 6 and 7 access the COBOL language sample applications. You must run job DSNTEJ73 before you can access option 4. You must run job DSNTEJ75 before you can access option 5. You must run job DSNTEJ77 before you can access option 6. You must run job DSNTEJ78 before you can access option 7. For more information, see “Employee resume and photo scenarios” on page 431.

Phase 4: Testing the IMS environment

Phase 4 installs the sample IMS transactions for both COBOL and PL/I. In the PL/I version, the phone application discussed in Phase 2 is also installed as an online transaction. For more information on the phone application, refer to “The phone application scenario” on page 420.

Jobs DSNTEJ4C and DSNTEJ4P

Job DSNTEJ4C is for COBOL; DSNTEJ4P is for PL/I. Both jobs perform the following functions:
- Precompile, compile, and link-edit the IMS online applications.
- Bind the IMS online applications.
- Create the message format service (MFS) panels for the online applications.
- Run the required PSBGEN and ACBGEN.

Select the proper job and define the applications and transactions to IMS. Member DSN8FIMS in *prefix*.SDSNSAMP contains information to assist in the definition step.

The verification transactions are single mode, single segment, and nonconversational.

Recommendation: Use SSM error option R because the program handles any errors. A resource translation table is not required.

Invoke the transaction by using the FORMAT command. The programs accept several lines of input on the first panel and display the results after you press ENTER.

To use a type of COBOL other than the one you specified on field 2 of panel
DSNTIPY, see “Special considerations for COBOL programs” on page 370.
Remember, you must use the same type of COBOL to prepare DSNTEJ2C,
DSNTEJ3C, DSNTEJ4C, and DSNTEJ5C.

If DSNTEJ4C runs successfully, it produces the return codes that are shown in Table 79.

Table 79. DSNTEJ4C return codes

	Step	PROCSTEP	Return code
#	PH04CS01	PC	0004
#		COB	0000
#		PLKED[1]	0004
#		LKED	0004
#	PH04CS02	PC	0000
#		COB	0000
#		PLKED[1]	0004
#		LKED	0004

Table 79. DSNTEJ4C return codes (continued)

Step	PROCSTEP	Return code
PH04CS03	PC	0000
	COB	0000
	PLKED[1]	0004
	LKED	0004
PH04CS04		0000
PH04CS05		0000
PH04CS06	S1	0000
	S2	0004
PH04CS07	S1	0000
	S2	0004
PH04CS08	C	0000
	L	0000
PH04CS09	G	0000

For DSNTEJ4C, the warning code that is expected from the precompiler step PH04CS01 is:

```
DB2 SQL PRECOMPILER     MESSAGES
DSNH0531 W   NO SQL STATEMENTS WERE FOUND
```

If DSNTEJ4P runs successfully, it produces the return codes that are shown in Table 80.

Table 80. DSNTEJ4P return codes

Step	PROCSTEP	Return Code
PH04PS01	PPLI	0000
	PC	0004
	PLI	0004
	LKED	0004
PH04PS02	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0004
PH04PS03	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0004
PH04PS04	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0004
PH04PS05	PPLI	0000
	PC	0004
	PLI	0004
	LKED	0004
PH04PS06	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0004
PH04PS07	PPLI	0000
	PC	0000
	PLI	0004
	LKED	0004
PH04PS08		0000
PH04PS09		0000

Table 80. DSNTEJ4P return codes (continued)

Step	PROCSTEP	Return Code
PH04PS10	S1	0000
	S2	0004
PH04PS11	S1	0000
	S2	0004
PH04PS12	S1	0000
	S2	0000 or 0004
PH04PS13	S1	0000
	S2	0000 or 0004
PH04PS14	S1	0000
	S2	0000 or 0004
PH04PS15	S1	0000
	S2	0000 or 0004
PH04PS16	C	0000
	L	0000
PH04PS17	G	0000
PH04PS18	C	0000
	L	0000
PH04PS19	G	0000
PH04PS20	C	0000
	L	0000
PH04PS21	G	0000

For DSNTEJ4P, the warning code expected from the precompiler step PH04PS01 is:

```
DB2 SQL PRECOMPILER     MESSAGES
DSNH0531 W   NO SQL STATEMENTS WERE FOUND
```

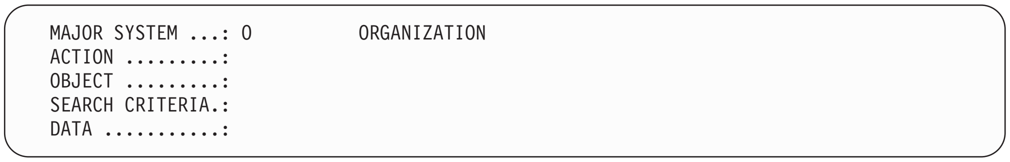
If either job DSNTEJ4C or job DSNTEJ4P fails or abends, rerun the jobs from the last successful step.

Starting an application in an IMS environment

After logging on to IMS, you can start the organization or project application by entering a FORMAT command. The FORMAT commands are:

- /FORMAT DSN8IPGO, which starts the PL/I organization version
- /FORMAT DSN8ICGO, which starts the COBOL organization version.

When you enter either of these two commands, the panel that is shown in Figure 48 is displayed.



```
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
```

Figure 48. Organization version of FORMAT command display

When the following command is entered, the panel that is shown in Figure 49 on page 390 is displayed.

```
/FORMAT DSN8IPFO
```

starts the PL/I projects version.

```
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
```

Figure 49. Project version of FORMAT command display

Using the phone application in IMS

When you use IMS, information is interactively processed. To begin, clear the screen and type in a FORMAT command. The FORMAT command is:

```
/FORMAT DSN8IPN0
```

starts PL/I phone application.

When the FORMAT command is entered, the panel shown in Figure 50 is displayed.

```
---------------------------- TELEPHONE DIRECTORY ----------------------------


                          LAST  NAME ==>

                          FIRST NAME ==>


LAST NAME           :  *  FOR LIST OF ENTIRE DIRECTORY
                       %  FOR GENERIC LIST (EX. K% = ALL  K - NAMES)
FIRST NAME(OPTIONAL):  %  FOR GENERIC LIST
```

Figure 50. Starting the phone application

Phase 5: Testing the CICS environment

Phase 5 tests the CICS environment. It installs the sample applications for COBOL and PL/I, and it prepares the CICS SQLCA formatter front-end.

Job DSNTEJ5A

DSNTEJ5A assembles and link-edits DSNTIAC, the CICS SQLCA formatter front-end. It also assembles and links the RCT and optionally adds the sample definitions to the CSD. Use DSNTIAC as an alternative to DSNTIAR when you want CICS services to do storage handling and program loading. If you are using CICS Version 4 or CICS Transaction Server 1.1, you need to modify job DSNTEJ5A to use steps DSN8FRCT and DSN8FRDO. You might need to tailor step DSN8FRDO for your system.

If DSNTEJ5A runs successfully, it produces the return codes that are shown in Table 81.

Table 81. DSNTEJ5A return codes

Step	PROCSTEP	Return code
PH05AS01		0000
PH05AS02		0000
PH05AS03		0000

Jobs DSNTEJ5C and DSNTEJ5P

Job DSNTEJ5C installs the sample application transactions in COBOL and prepares the organization application. Job DSNTEJ5P installs the transactions in PL/I and prepares the organization, project, and phone applications.

To use a type of COBOL other than the one you specified on field 2 of panel
DSNTIPY, see “Special considerations for COBOL programs” on page 370.
Remember, you must use the same type of COBOL to prepare DSNTEJ2C,
DSNTEJ3C, DSNTEJ4C, and DSNTEJ5C.

Both phase 5 jobs perform the following functions:
- Compile and link-edit the CICS online applications
- Bind the CICS online applications
- Create the BMS maps for the online applications.

Select the proper job, and define transactions, programs, and BMS maps to CICS.

prefix.SDSNSAMP members DSN8FPPT, DSN8FPCT, and DSN8FRCT contain the respective PPT, PCT, and RCT entries required for the phase 5 applications. These members help you perform the definition step. Make sure that the subsystem ID (SUBID) in the RCT entry matches your DB2 subsystem ID.

If DSNTEJ5C runs successfully, it produces the return codes that are shown in Table 82.

Table 82. DSNTEJ5C return codes

Step	PROCSTEP	Return code
MAPG	ASSEM	0000
MAPD	ASSEM	0000
DSNH		0000 or 0004
BIND		0000
MAPGP	ASSEM	0000
MAPGL		0000
MAPDP	ASSEM	0000
MAPDL		0000

If DSNTEJ5C fails or abends, rerun the job from the last successful step. To receive more prepare-time detail from DSNTEJ5C, change the parameters TERM(LEAVE) and PRINT(LEAVE) to TERM(TERM) and PRINT(TERM). See the discussion of the DSNH CLIST in *DB2 Command Reference* for more information.

If DSNTEJ5P runs successfully, it produces the return codes that are shown in Table 83.

Table 83. DSNTEJ5P return codes

Step	PROCSTEP	Return code
PH05PS01	ASSEM	0000
PH05PS02	ASSEM	0000
PH05PS03	ASSEM	0000
PH05PS04	ASSEM	0000

Table 83. DSNTEJ5P return codes (continued)

Step	PROCSTEP	Return code
PH05PS05	ASSEM	0000
PH05PS06		0004
PH05PS07	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS08		0004
PH05PS09	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS10		0004
PH05PS11	PPLI PC PLI LKED	0004 0000 0004 0004
PH05PS12		0004
PH05PS13	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS14	ASSEM	0000
PH05PS15	ASSEM	0000
PH05PS16		0004
PH05PS17	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS18		0004
PH05PS19	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS20		0004
PH05PS21	PPLI PC PLI LKED	0000 0000 0004 0004
PH05PS22		0000 or 0004
PH05PS23		0000
PH05PS24	ASSEM	0000
PH05PH25		0000
PH05PS26	ASSEM	0000
PH05PH27		0000
PH05PS28	ASSEM	0000
PH05PH29		0000
PH05PS30	ASSEM	0000
PH05PH31		0000
PH05PS32	ASSEM	0000
PH05PH33		0000
PH05PS34	ASSEM	0000

Table 83. DSNTEJ5P return codes (continued)

Step	PROCSTEP	Return code
PH05PH35		0000
PH05PS36	ASSEM	0000
PH05PH37		0000

If DSNTEJ5P fails or abends, rerun the job from the last successful step. You might find it convenient to break up DSNTEJ5P and run only the unsuccessful steps.

Starting an application in a CICS environment

After logging on to CICS, you can start an organization or project application by entering a CICS transaction code. The CICS transaction codes are:
- D8PP, which starts the PL/I project version
- D8PS, which starts the PL/I organization version
- D8CS, which starts the COBOL organization version.

When these transaction codes are entered, the panels that are shown in Figure 51 and Figure 52 are displayed.

```
              ACTION SELECTION
MAJOR SYSTEM ...: O          ORGANIZATION
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
SELECT AN ACTION FROM FOLLOWING LIST

   A     ADD (INSERT)
   D     DISPLAY (SHOW)
   E     ERASE (REMOVE)
   U     UPDATE (CHANGE)
```

Figure 51. Initial panel for the organization application in CICS

Figure 52 shows the initial panel for the CICS project application.

```
              ACTION SELECTION
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........:
OBJECT .........:
SEARCH CRITERIA.:
DATA ...........:
SELECT AN ACTION FROM FOLLOWING LIST

   A     ADD (INSERT)
   D     DISPLAY (SHOW)
   E     ERASE (REMOVE)
   U     UPDATE (CHANGE)
```

Figure 52. Initial panel for the project application in CICS

Refer to “Specifying values in the sample application panels” on page 411 for the criteria you need to enter to run the organization and project applications.

Using the phone application in CICS

When you use CICS, information is interactively processed. To begin, clear the screen and type in the transaction code:

```
D8PT
```

You can change the transaction codes when you install DB2. Check with your system administrator to find out if they have been changed from those shown.

Using CICS storage-handling facilities

To use the CICS storage-handling facilities when running the CICS sample applications, change your DSNTIAR calls to DSNTIAC calls in DSN8MC*xx* and DSN8MP*xx*. Then rerun job DSNTEJ5C or job DSNTEJ5P. The calls should look like this:

```
CALL DSNTIAC(EIB,COMMAREA,SQLCA,MSG,LRECL)
```

You must also define DSNTIAC and DSNTIA1 in the CSD.

Phase 6: Accessing data at a remote site

You can use this phase to verify that the features of DRDA access and DB2 private protocol access are working correctly. During this optional phase, you access data at a remote site using multiple sample applications:

- The DRDA access application (DSNTEJ6 in conjunction with DSNTEJ3C)
- The DB2 private protocol access application (user maintained)
- The stored procedure without result set sample (DSNTEJ6S and DSNTEJ6P)
- The stored procedure with result set sample (DSNTEJ6T and DSNTEJ6D)
- The stored procedure for invoking utilities (DSNTEJ6U, DSNTEJ6V, and DSNTEJ6Z)
- The stored procedure for invoking WLM_REFRESH (DSNTEJ6W)
- The stored procedure for IMS Open Database Access (DSNTEJ61 and DSNTEJ62)
- The SQL procedure batch sample (DSNTEJ63 and DSNTEJ64)
- The SQL procedures processor invocation sample (DSNTEJ65)

The installation CLIST prepares samples DSNTEJ6, DSNTEJ6S, DSNTEJ6P, DSNTEJ6T, DSNTEJ6D, DSNTEJ63, DSNTEJ64, and DSNTEJ65 for you only if you specify **YES** or **AUTO** on the DDF startup option of panel DSNTIPR. If you specify **NO**, the installation CLIST does not prepare these samples. In this case, you should not try to run these phase 6 samples.

The installation CLIST prepares the stored procedures samples only if you specify a WLM-established stored procedure JCL PROC name (field 1) on panel DSNTIPX. If you replace the default value with blanks on field 1 of this panel, you cannot start the WLM-established stored procedures address space until you update the subsystem parameter.

The installation CLIST tailors the phase 6 sample jobs according to the information you specify in field REMOTE LOCATION (field 1) of panel DSNTIPY. The guidelines for this field are:

- If the field is blank, the installation CLIST only customizes phase 6 sample jobs DSNTEJ63, DSNTEJ64, DSNTEJ65, and DSNTEJ6U, and DSNTEJ6V. Jobs DSNTEJ63, DSNTEJ64, and DSNTEJ65 use the local location name when the REMOTE LOCATION field is blank. Jobs DSNTEJ6U and DSNTEJ6V do not use a remote location name.
- If the value in the field is the same as the location name for the DB2 subsystem you are installing (field 2 of panel DSNTIPR), the stored procedures samples are prepared and customized for local use. However, the DRDA access sample is not prepared. This includes DSNTEJ6 and the DRDA access component of DSNTEJ3C.

- If the value in the field is different from the DB2 location name, the installation CLIST prepares the phase 6 samples assuming that the remote location is the server and that the local system is the client.

If you are installing and testing two DB2 subsystems concurrently, you must designate one as the server and the other as the client. If you change these designations during your testing, your results will be unpredictable. Verify that your VTAM APPL statement has the parameter SYNCLVL=SYNCPT defined. This allows updates at several locations.

DRDA Access sample

The distributed application using DRDA access is executed as part of Phase 6. The application is prepared in Phase 3 as part of DSNTEJ3C. Before this application can be run correctly as a DRDA access sample, you must run job DSNTEJ6 at both the local and remote sites to tailor the DEPT sample table for use in a distributed environment.

To set up your samples testing for concurrent installations at two DB2 locations, follow these guidelines:

- Designate one location as the requester (the client) and the other location as the server.
- Run the client version of DSNTEJ6 at the client site only; do **not** run the client version of DSNTEJ6 at the server.
- Edit the server version of DSNTEJ6 at the remote server site; do **not** run the server version of DSNTEJ6 at the client.
- Locate the following text in the server version of DSNTEJ6 within step PH06S01:

  ```
  UPDATE DEPT SET LOCATION = (your remote location name) WHERE DEPTNO = 'F22';
  UPDATE DEPT SET LOCATION = (your  location name) WHERE LOCATION = '  ';
  ```

  This text should be replaced with:

  ```
  UPDATE DEPT SET LOCATION = (your location name) WHERE DEPTNO = 'F22';
  UPDATE DEPT SET LOCATION = (your remote location name) WHERE LOCATION = '  ';
  ```

Job DSNTEJ6

Job DSNTEJ6 consists of the following step:

Step	Function
1	Updates the location column in the department table to the sample location entered on installation panel DSNTIPY

If DSNTEJ6 runs successfully, it produces the return code that is shown in Table 84.

Table 84. DSNTEJ6 return codes

Step	PROCSTEP	Return code
PH06S01		0000

After job DSNTEJ6 has completed successfully, start the distributed application scenario by following the instructions in “Starting an application in an ISPF/TSO environment in phase 6” on page 397.

DB2 Private Protocol Access sample

To test distributed processing that uses DB2 private protocol access, create a job that performs the following functions:

Step	Function
1	Removes objects VPHONE and VEMPLP, which point to local tables, by dropping VPHONE and VEMPLP views or dropping VPHONE and VEMPLP aliases
2	Sets up sample table access by creating aliases VPHONE and VEMPLP, which point to views DSN8810.VPHONE and DSN8810.VEMPLP at a remote location

Notes:
1. It is assumed that the views DSN8810.VPHONE and DSN8810.VEMPLP and their underlying tables exist at the remote location. If they do not exist, run job DSNTEJ1 to create them.
2. Step 1 always has one set of SQL statements that fail; it either drops the views or the aliases, but not both.
3. If you want to point back to the local sample tables after executing this job, run a job that drops the aliases VPHONE and VEMPLP.

If you specified DRDA as the default database protocol on panel DSNTIP5, you must specify DBPROTOCOL(PRIVATE) to test DB2 private protocol access. After running this job, re-run Phase 2 jobs DSNTEJ2C through DSNTEJ2P. The Phase 2 phone application jobs now reference data at the remote location through aliases.

Sample JCL statements for performing these functions are shown in Figure 53.

```
//JOBLIB  DD  DISP=SHR,
//            DSN=prefix.SDSNLOAD
//*
//*       STEP 1 : SET UP THE SAMPLE TABLE ACCESS
//*
//STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
 DSN SYSTEM(DSN)
 RUN  PROGRAM(DSNTIAD) PLAN(DSNTIA81) PARM('RC0') -
      LIB('prefix.RUNLIB.LOAD')
 END
//SYSIN    DD *
  DROP VIEW DSN8810.VPHONE;
  DROP VIEW DSN8810.VEMPLP;
  COMMIT;
  DROP ALIAS VPHONE;
  DROP ALIAS VEMPLP;
  COMMIT;
//*
//*       STEP 2 : SET UP THE SAMPLE TABLE ACCESS
//*
//STEP2 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
 DSN SYSTEM(DSN)
 RUN  PROGRAM(DSNTIAD) PLAN(DSNTIA81) -
      LIB('prefix.RUNLIB.LOAD')
 END
//SYSIN    DD *
  CREATE ALIAS VPHONE FOR SAMPLOC.DSN8810.VPHONE;
  CREATE ALIAS VEMPLP FOR SAMPLOC.DSN8810.VEMPLP;
//*
```

Figure 53. Sample JCL statements for DB2 private protocol access

Starting an application in an ISPF/TSO environment in phase 6

You must have access to ISPF load module libraries in order to run the ISPF/CAF sample application. See "Make panels, messages, and load modules available to ISPF and TSO" on page 265 for more information on this procedure. To start the application, enter a CALL command from option 6 of the ISPF primary option menu.

To start the COBOL sample version of the connection manager, enter:

```
CALL 'prefix.RUNLIB.LOAD(DSN8SCM)'
```

After you enter this command, DB2 displays the sample applications panel, shown in Figure 47. Choosing option 3 on the sample applications panel during Phase 6 invokes the COBOL organization application, which uses DRDA access for distributed data. For more information, see "The distributed organization application scenario" on page 423.

Stored procedure samples

The stored procedure sample applications demonstrate different ways that a stored procedure can be used by a client to issue DB2 commands to a DB2 server. There are several applications discussed:

- One sample without a result set
- One sample with a result set
- Two samples using the utilities stored procedure DSNUTILS
- One sample using the utilities stored procedure DSNUTILU
- One sample for IMS Open Database Access (ODBA) support
- Two samples for SQL procedures
- One sample to list the settings of subsystem parameters
- One sample to invoke a stored procedure that refreshes the WLM environment

Most of the applications prepare and run two programs; one prepares a stored procedure, and one executes a client program that calls the stored procedure and returns some response.

The stored procedure sample jobs are edited only when you specify selected fields on the installation panels. For details on which fields are required for the sample jobs, see the notes in Table 56 on page 365. To run these applications, you must first start the WLM-established stored procedures address space (SPAS). See "Routine parameters panel: DSNTIPX" on page 229 for information on generating the JCL to start the SPAS. Before starting the SPAS, you must have Language Environment and a Language Environment-compatible version of PL/I, C, or COBOL installed (depending on the job) to run the stored procedure sample jobs.

Stored procedure sample without result set

This application consists of two jobs: DSNTEJ6S and DSNTEJ6P. Job DSNTEJ6S must be run before job DSNTEJ6P. To run these jobs, you must have the PL/I product installed on your client and server systems in addition to Language Environment. This application prepares and runs:

- A stored procedure that uses the instrumentation facility interface to issue DB2 commands.
- A client program that receives DB2 command text, calls the stored procedure to issue the commands, receives the responses from the stored procedure in a parameter that is passed back, and prints the results.

For concurrent installations at two DB2 locations:

- Run the server version of DSNTEJ6S on the server system only; do **not** run the client version of DSNTEJ6S on the server
- Run the client version of DSNTEJ6P on the client system only; do **not** run the server version of DSNTEJ6P on the client

Job DSNTEJ6S

Job DSNTEJ6S compiles and link-edits the sample stored procedure DSN8EP2. It also updates the SYSIBM.SYSROUTINES catalog table with information about the stored procedure. If you have SQL statements in your stored procedure, you must remove the comment character in the JCL from the step that binds the stored procedure package.

You must run job DSNTEJ6S at the DB2 server location.

If DSNTEJ6S runs successfully, it produces the return codes that are shown in Table 85 on page 399.

Table 85. DSNTEJ6S return codes

Step	PROCSTEP	Return code
PH06SS01		0000
PH06SS02		0000
PH06SS03	PPLI PC PLI LKED	0000 0004 0004 0000

If SQLCODE -592 is received during the execution of DSNTEJ6T, ensure that SPA is active.

Job DSNTEJ6P

Job DSNTEJ6P compiles, link-edits, binds, and runs a sample program, DSN8EP1, that invokes the sample stored procedure.

Before you run DSNTEJ6P, run DSNTEJ6S to create the sample stored procedure.

If DSNTEJ6P runs successfully, it produces the return codes that are shown in Table 86.

Table 86. DSNTEJ6P return codes

Step	PROCSTEP	Return code
PH06PS01	PPLI PC PLI LKED	0000 0000 0004 0000
PH06PS02		0004
PH06PS03		0000

Output from a successful execution of DSNTEJ6P lists each DB2 command executed, followed by the messages generated by the DB2 command processor.

You can compare the output from this job with the sample output for DSNTEJ6P found in member DSN8TJ6P in your *prefix*.SDSNIVPD data set.

Stored procedure sample with result set

This application consists of two jobs; DSNTEJ6T and DSNTEJ6D. Job DSNTEJ6T must be run before job DSNTEJ6D. You must have C for z/OS installed, in addition to Language Environment to run DSNTEJ6D and DSNTEJ6T. This application prepares and runs:

- A stored procedure that uses the instrumentation facility interface to issue DB2 commands.
- A client program that receives DB2 command text, calls the stored procedure to issue the commands, receives the responses from the stored procedure in a result set, and prints the results.

For concurrent installations at two DB2 locations:

- Run the server version of job DSNTEJ6T on the server side only; do **not** run it on the client side
- Run the client version of job DSNTEJ6D on the client side only; do **not** run it on the server side

Job DSNTEJ6T

Job DSNTEJ6T registers, prepares, and binds the sample stored procedure, DSN8ED2, on the server. It also defines a created temporary table to receive the IFI output that is returned as a result set.

If DSNTEJ6T runs successfully, it produces the return codes that are shown in Table 87.

Table 87. DSNTEJ6T return codes

Step	PROCSTEP	Return code
PH06TS01		0000
PH06TS02		0000
PH06TS03	PC C PLKED LKED	0004 0000 0004 0000
PH06TS04		0000 or 0004

If SQLCODE -592 is received during the execution of DSNTEJ6T, ensure that SPA is active.

Job DSNTEJ6D

Job DSNTEJ6D compiles, link-edits, binds, and runs sample program DSN8ED1, that invokes the sample for using stored procedure result sets.

Before you run DSNTEJ6D, run DSNTEJ6T to create the sample stored procedure.

If DSNTEJ6D runs successfully, it produces the return codes that are shown in Table 88.

Table 88. DSNTEJ6D return codes

Step	PROCSTEP	Return code
PH06DS01	PC C PLKED LKED	0000 0000 or 0004 0000 or 0004 0000
PH06DS02		0004
PH06DS03		0000

Output from a successful execution of DSNTEJ6D lists each DB2 command executed, followed by the messages generated by the DB2 command processor.

You can compare the output from this job with the sample output for DSNTEJ6D found in member DSN8TJ6D in your *prefix*.SDSNIVPD data set.

Sample callers of utilities stored procedures

The DSNUTILS stored procedure enables execution of DB2 utilities from a DB2 application program using the SQL CALL statement. When called, DSNUTILS dynamically allocates the specified data sets, creates the utility input stream (SYSIN), invokes DB2 utilities (DSNUTILB), deletes all rows currently in the created temporary table (SYSIBM.SYSPRINT), captures the utility output stream (SYSPRINT), and puts this output into the created temporary table

(SYSIBM.SYSPRINT). The DSNUTILU stored procedure is very similar to DSNUTILS, but accepts parameters in Unicode.

The DSNUTILS and DSNUTILU stored procedures must run as a WLM-managed stored procedure.

Job DSNTEJ6U

Job DSNTEJ6U compiles, link-edits, binds, and runs sample PL/I program DSN8EPU, which invokes the DSNUTILS stored procedure to execute a utility.

Before you run DSNTEJ6U, verify that the DSNUTILS stored procedure was successfully created in job DSNTIJSG. DSNTEJ6U requires a WLM procedure. See Appendix B of *DB2 Utility Guide and Reference* for an example of how to customize a WLM procedure for DSNUTILS.

If DSNTEJ6U completes successfully, it produces the return codes that are shown in Table 89.

Table 89. DSNTEJ6U return codes

Step	PROCSTEP	Return code
PH06US01	PC	0000
	PLI	0004
	LKED	0000
PH06US02		0004
PH06US03		0000

Output from a successful execution of DSNTEJ6U lists the parameters specified followed by the messages generated by the DB2 DIAGNOSE DISPLAY MEPL utility. For more details on utilities and invoking utilities as a stored procedure, see Appendix B of *DB2 Utility Guide and Reference*.

You can compare the output from this job with the sample output for DSNTEJ6U found in member DSN8TJ6U in your *prefix*.SDSNIVPD data set.

Job DSNTEJ6R

Job DSNTEJ6R compiles, link-edits, binds, and runs sample C program DSN8ED8, which invokes the DSNUTILU stored procedure to execute a utility. For convenience on TSO, the utility control statement inputted to DSN8ED8 is encoded in EBCDIC and converted to Unicode before being passed to DSNUTILU.

Before you run DSNTEJ6R, verify that the DSNUTILU stored procedure was successfully created in job DSNTIJSG. The DSNUTILU stored procedure must run as a WLM-managed stored procedure. See Appendix B of *DB2 Utility Guide and Reference* for an example of how to customize a WLM procedure for DSNUTILU.

If DSNTEJ6R completes successfully, it produces the return codes that are shown in Table 90.

Table 90. DSNTEJ6R return codes

Step	PROCSTEP	Return code
PH06RS00		0000
PH06RS01	PC	0000
	C	0000

Table 90. DSNTEJ6R return codes (continued)

Step	PROCSTEP	Return code
	PLKED	0004
	LKED	0000
PH06RS02		0000 or 0004
PH06RS03		0000

Output from a successful execution of DSNTEJ6R lists the parameters specified followed by the messages generated by the DB2 DIAGNOSE DISPLAY MEPL utility. For more details on using Unicode-encoded utility statements and invoking utilities as a stored procedure, see Appendix B of *DB2 Utility Guide and Reference*.

You can compare the output from this job with the sample output for DSNTEJ6U found in member DSN8TJ6U in your *prefix*.SDSNIVPD data set.

Job DSNTEJ6V

Job DSNTEJ6V compiles, link-edits, binds, and runs sample C++ program DSN8EE1, which invokes the DSNUTILS stored procedure to execute a utility. DSNTEJ6V requires a WLM procedure. See Appendix B of *DB2 Utility Guide and Reference* for an example of how to customize a WLM procedure for DSNUTILS.

If DSNTEJ6V completes successfully, it produces the return codes that are shown in Table 91.

Table 91. DSNTEJ6V return codes

Step	PROCSTEP	Return code
PH06VS01	PC	0004
	CP	0000
	PLKED	0004
	LKED	0004
PH06VS02	PC1	0000
	CP1	0000
	PC2	0000
	CP2	0000
	PLKED	0004
	LKED	0000
PH06VS03		0000 or 0004
PH06VS04		0000

A successful execution of DSNTEJ6V unloads rows and columns from the PROJ sample table. For more details about utilities and invoking utilities as a stored procedure, see Appendix B of *DB2 Utility Guide and Reference*.

You can compare the output from this job with the sample output for DSNTEJ6V found in member DSN8TJ6V in your *prefix*.SDSNIVPD data set.

Job DSNTEJ6W

Job DSNTEJ6W is a JCL job that creates and initializes a sample SAF resource profile, and prepares, binds, and executes DSN8ED6. DSN8ED6 is a C language caller of WLM_REFRESH that accepts the WLM environment name and, optionally, the subsystem ID and an SQLID to be in effect when WLM_REFRESH is invoked.

The installation CLIST customizes DSNTEJ6W to run in and recycle the same WLM environment, which is the environment you specified in the WLM ENVIRONMENT field on installation panel DSNTIPX.

If you are not authorized to create special resource profiles, have your system security administrator perform the first step of this job. The SQLID used to run DSNTEJ6W needs READ access to this profile.

For more information about WLM, see *DB2 Application Programming and SQL Guide*.

If DSNTEJ6U completes successfully, it produces the return codes that are shown in Table 92.

Table 92. DSNTEJ6W return codes

Step	PROCSTEP	Return code
PH06WS01		0000
PH06WS02	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH06WS03		0000 or 0004
PH06WS04		0000

Job DSNTEJ6Z

Job DSNTEJ6Z generates a report of current subsystem parameter settings. This report is generated by DSN8ED7, a C-language caller of stored procedure DSNWZP. You must have TRACE and MONITOR1 privileges to run DSNWZP.

If DSNTEJ6U completes successfully, it produces the return codes that are shown in Table 93.

Table 93. DSNTEJ6Z return codes

Step	PROCSTEP	Return code
PH06ZS01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH06ZS02		0000 or 0004
PH06ZS03		0000

A successful execution of DSNTEJ6Z provides a report as shown in Figure 54 on page 404.

```
 DSN8ED7: Sample DB2 Configuration Setting Report Generator

 Macro    Parameter  Current     Description/         Install   Fld
 Name     Name       Setting     Install Field Name   Panel ID  No.
 ____________________________________________________________________
 DSN6SYSP AUDITST    0000000000  AUDIT TRACE          DSNTIPN      1
 DSN6SYSP CONDBAT    0000000064  MAX REMOTE CONNECTED DSNTIPE      4
 DSN6SYSP CTHREAD    00030       MAX USERS            DSNTIPE      2
 DSN6SYSP DLDFREQ    00005       LEVELID UPDATE FREQ  DSNTIPL     14
 DSN6SYSP PCLOSEN    00005       SWITCH CHKPTS        DSNTIPL     12
...
```

Figure 54. DSNTEJ6Z report format

Sample ODBA stored procedure

IMS Open Database Access (ODBA) support allows a DB2 stored procedure to directly connect to an IMS DBCTL system and issue DL/I calls to access IMS databases. A stored procedure can issue database DL/I requests via a new callable interface. For more information on IMS ODBA, see *DB2 Application Programming and SQL Guide*. ODBA requires IMS Version 6.

This application consists of two jobs: DSNTEJ61 and DSNTEJ62. Job DSNTEJ61 must be run before DSNTEJ62. You must have COBOL for MVS and VM and Language environment installed to run DSNTEJ61 and DSNTEJ62. You must start a WLM-established stored procedure address space to run DSNTEJ61 and DSNTEJ62. You need to update the startup procedure for the WLM-established stored procedure address space to add the ODBA data set names to the STEPLIB and DFSRESLB concatenations. An example of a data set name for ODBA is IMSVS.RESLIB.

For more information about setting up a WLM-established stored procedure address space, see *DB2 Application Programming and SQL Guide*.

Job DSNTEJ61

Job DSNTEJ61 prepares a sample stored procedure DSN8EC1, that uses ODBA. DSN8EC1 can add, update, delete, and display telephone directory records from the IMS sample database, DFSIVD1. DSN8EC1 shows how the AERTDLI API is used to issue IMS DL/I calls.

Before running DSNTEJ61, read the Dependencies information of the job prolog to verify that the server site is correctly configured. If DSNTEJ61 runs successfully, it produces the return codes that are shown in Table 94.

Table 94. DSNTEJ61 return codes

Step	PROCSTEP	Return code
PH061S01		0000
PH061S02		0000
PH061S03	PC COB PLKED LKED	0004 0000 0004 0000

Job DSNTEJ62

Job DSNTEJ62 prepares and invokes the sample client program DSN8EC2, which calls stored procedure DSN8EC1. DSN8EC2 calls the stored procedure DSN8EC1 multiple times to add, delete, and display telephone directory records.

Before running DSNTEJ62, perform the manual editing described in the Dependencies information in the job prolog. If DSNTEJ62 runs successfully, it produces the return codes that are shown in Table 95.

Table 95. DSNTEJ62 return codes

Step	PROCSTEP	Return code
PH062S01	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH062S02		0000 or 0004
PH062S03		0000

Sample SQL procedures

The two applications for SQL procedures are:

- DSNTEJ63 and DSNTEJ64

  Job DSNTEJ63 must be run before DSNTEJ64. Job DSNTEJ63 prepares a sample SQL procedure. Job DSNTEJ64 prepares and executes a C program that calls the sample SQL procedure.
- DSNTEJ65

  Job DSNTEJ65 has three parts:
  - Prepares a C program that calls the DB2 SQL procedures processor (DSNTPSMP).
  - Prepares a sample SQL procedure.
  - Prepares and executes a C program that calls the SQL procedure.

  Use DSNTEJ65 to verify that the DB2 SQL procedures processor, DSNTPSMP, is working correctly.

C and Language Environment are required for jobs DSNTEJ63, DSNTEJ64, and DSNTEJ65. For job DSNTEJ65, you must start a WLM-established stored procedures address space for DSNTPSMP. You must update the start-up procedure for the WLM-established stored procedure address space as listed in the prolog of sample start-up procedure DSN8WLMP. For more information about setting up a WLM-established stored procedure address space, see Part 6 of *DB2 Application Programming and SQL Guide*.

Job DSNTEJ63

Job DSNTEJ63 prepares the sample SQL procedure, DSN8ES1, which accepts a department number and returns salary and bonus data.

Before running DSNTEJ63, perform the manual editing described in the Dependencies information in the job prolog. If DSNTEJ63 runs successfully, it produces the return codes that are shown in Table 96 on page 406.

If you specified a remote location name on installation panel DSNTIPY, DSNTEJ63 should be run on the remote server site.

Table 96. DSNTEJ63 return codes

Step	PROCSTEP	Return code
PH063S01		0000
PH063S02	PC PCC C PLKED LKED	0004 0004 0000 0004 0000
PH063S03		0000 or 0004

Job DSNTEJ64

Job DSNTEJ64 prepares and executes DSN8ED3, a sample routine that calls the sample SQL procedure, DSN8ES1. You must run job DSNTEJ63 before running job DSNTEJ64.

Before running DSNTEJ64, perform the manual editing described in the Dependencies information in the job prolog. If DSNTEJ64 runs successfully, it produces the return codes that are shown in Table 97.

Table 97. DSNTEJ64 return codes

Step	PROCSTEP	Return code
PH064S01	PC C PLKED LKED	0000 0000 0004 0000
PH064S02		0000 or 0004
PH064S03		0000

You can compare the output from this job to the sample output for DSNTEJ64, which is found in member DSN8TJ64 in data set named *prefix*.SDSNIVPD.

Job DSNTEJ65

Job DSNTEJ65 demonstrates program preparation of an SQL procedure using the DB2 SQL procedures processor, DSNTPSMP. If you specified a remote location name on installation panel DSNTIPY, then jobs DSNTPSMP and DSNTIJSG, and the job DSNTIJRX should be run on the remote server site to bind DB2 REXX language support. The major components of job DSNTEJ65 are:

- DSN8WLMP is a sample start-up procedure for a WLM-established stored procedures address space in which DSNTPSMP runs. DSN8WLMP is located in *prefix*.SDSNSAMP.
- DSN8ED4 is a sample C program that calls the DB2 SQL procedures processor.
- DSN8ES2 is a sample SQL procedure that calculates employee bonuses.
- DSN8ED5 is a sample C program that calls the SQL procedure DSN8ES2.

Before running DSNTEJ65, perform the manual editing described in the Dependencies information in the job prolog. You must also manually tailor DSN8WLMP, the sample WLM startup procedure for DSNTPSMP. If DSNTEJ65 runs successfully, it produces the return codes that are shown in Table 98 on page 407.

Table 98. DSNTEJ65 return codes

Step	PROCSTEP	Return code
PH065S01	PC C PLKED LKED	0000 0000 0004 0000
PH065S02		0000 or 0004
PH065S03		0000 or 0004
PH065S04		0000 or 0004
PH065S05	PC C PLKED LKED	0000 0000 0004 0000
PH065S06		0000 or 0004
PH065S07		0000

You can compare the output from this job to the sample output for DSNTEJ65, which is found in member DSN8TJ65 in the data set named *prefix*.SDSNIVPD.

Phase 7: Accessing LOB data

This optional phase demonstrates how to set up and use a DB2 LOB application. This phase creates an extension to the Employee sample database to manage employee resumes and photo images. You run these jobs in this phase:

- DSNTEJ7: Creates the Employee resume and photo table, then loads the resumes.
- DSNTEJ71, DSNTEJ76: Populates the photo images and then validates that the resume and photo data is stored correctly.
- DSNTEJ73, DSNTEJ77: Prepares an ISPF application for viewing employee resume data.
- DSNTEJ75, DSNTEJ78: Prepares a GDDM application for viewing employee photo images.

Job DSNTEJ75 and DSNTEJ78 are not tailored by the installation CLIST unless you specify non-blank values for GDDM MACLIB and GDDM LOAD MODULES on panel DSNTIPW.

After you run these jobs, you can use ISPF and GDDM to view the sample employee resume and photo data.

Job DSNTEJ7

Job DSNTEJ7 demonstrates how to create a LOB table with all the accompanying LOB table spaces, auxiliary tables, and indexes. It also demonstrates how to use the DB2 LOAD utility to load a CLOB column of fewer than 32 KB.

Job DSNTEJ7 consists of the following steps:

Step	Function
1	Drops sample LOB objects
2	Creates sample Employee Resume and Photo LOB table
3	Creates aliases for the table, then grants access to it
4	Uses the DB2 LOAD utility to load CLOB data

Step	Function
5	Generates run-time statistics on the table

If DSNTEJ7 runs successfully, it produces the return codes that are shown in Table 99.

Table 99. DSNTEJ7 return codes

JOBSTEP	PROCSTEP	Return Code
PH07S01		0000
PH07S02		0000
PH07S03		0000 or 0004
PH07S04	DSNUPROC	0000
PH07S05	DSNUPROC	0000

Job DSNTEJ71

Job DSNTEJ71 compiles, link-edits, and binds two sample applications that manipulate LOB data. The DSN8DLPL sample application demonstrates how to use LOB locators to populate a LOB column larger than 32KB. The DSN8DLTC sample application validates the contents of the LOB table, verifying that it was populated correctly.

If DSNTEJ71 runs successfully, it produces the return codes that are shown in Table 100.

Table 100. DSNTEJ71 return codes

JOBSTEP	PROCSTEP	Return Code
PH071S01	PC C PLKED LKED	0000 0000 0004 0000
PH071S02	PC C PLKED LKED	0000 0000 0004 0000
PH071S03		0000
PH071S04		0000
PH071S05		0000

You can compare the output from this job with the sample output for DSNTEJ71 found in member DSN8TJ71 in your *prefix*.SDSNIVPD data set.

Job DSNTEJ73

Job DSNTEJ73 compiles the DSN8DLRV sample application, which demonstrates how to use built-in functions like POSSTR and SUBSTR in order to traverse a CLOB column and break out data from it.

If DSNTEJ73 runs successfully, it produces the return codes that are shown in Table 101 on page 409.

Table 101. DSNTEJ73 return codes

JOBSTEP	PROCSTEP	Return Code
PH073S01	PC	0004
	C	0000
	PLKED	0004
	LKED	0000
PH073S02	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH073S03		0000 or 0004

After running DSNTEJ73, you can view sample employee resumes by invoking DSN8DLRV, as described in "Starting an application in an ISPF/TSO environment in phase 7" on page 411.

Job DSNTEJ75

Job DSNTEJ75 runs sample program DSN8DLPV, which demonstrates how to manipulate BLOB data (employee photo images).

If DSNTEJ75 runs successfully, it produces the return codes that are shown in Table 102.

Table 102. DSNTEJ75 return codes

JOBSTEP	PROCSTEP	Return Code
PH075S01	PC	0000
	C	0000
	PLKED	0004
	LKED	0000
PH075S02		0000 or 0004

After running DSNTEJ75, you can view sample employee photo images by invoking DSN8DLPV, as described in "Starting an application in an ISPF/TSO environment in phase 7" on page 411.

Job DSNTEJ76

Job DSNTEJ76 compiles, link-edits, and binds two sample COBOL applications that manipulate LOB data. The DSN8CLPL sample application demonstrates how to use LOB locators to populate a LOB column that is larger than 32 KB. The DSN8CLTC sample application validates the contents of the LOB table, verifying that it was populated correctly.

If DSNTEJ76 runs successfully, it produces the return codes that are shown in Table 103.

Table 103. DSNTEJ76 return codes

JOBSTEP	PROCSTEP	Return code
PH076S01	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000
PH076S02	PC	0000
	COB	0000
	PLKED	0004
	LKED	0000

Table 103. DSNTEJ76 return codes (continued)

JOBSTEP	PROCSTEP	Return code
PH076S03		0000 or 0004
PH076S04		0000
PH076S05		0000

After running DSNTEJ76, you can compare the output with the sample output for DSNTEJ76 found in member DSN8TJ76 in your *prefix*.SDSNIVPD data set.

Job DSNTEJ77

Job DSNTEJ77 compiles the DSN8CLRV sample COBOL application, which demonstrates how to use built-in functions, such as POSSTR and SUBSTR, to traverse a CLOB column and break out data from it. To run DSN8CLRV, you must run the first step of job DSNTEJ73 to compile DSN8SDM.

If DSNTEJ77 runs successfully, it produces the return codes that are shown in Table 104.

Table 104. DSNTEJ77 return codes

JOBSTEP	PROCSTEP	Return Code
PH077S01	PC COB PLKED LKED	0000 0000 0004 0000
PH077S02		0000 or 0004

After running DSNTEJ77, you can view sample employee photo resumes by invoking DSN8CLRV as described in “Starting an application in an ISPF/TSO environment in phase 7” on page 411.

Job DSNTEJ78

Job DSNTEJ78 runs sample COBOL program DSN8CLPV, which demonstrates how to manipulate BLOB data (employee photo images).

If DSNTEJ78 runs successfully, it produces the return codes that are shown in Table 105.

Table 105. DSNTEJ78 return codes

JOBSTEP	PROCSTEP	Return Code
PH078S01	PC COB PLKED LKED	0000 0000 0004 0000
PH078S02		0000 or 0004

After running DSNTEJ78, you can view sample employee photo images by invoking DSN8CLPV as described in “Starting an application in an ISPF/TSO environment in phase 7” on page 411.

Starting an application in an ISPF/TSO environment in phase 7

You must have access to ISPF load module libraries in order to run the Employee Resume and Photo sample applications. See “Make panels, messages, and load modules available to ISPF and TSO” on page 265 for more information on this procedure. To start the application, enter a CALL command from option 6 of the ISPF primary option menu.

To start the Employee Resume and Photo sample applications, enter:

```
CALL 'prefix.RUNLIB.LOAD(DSN8SDM)'
```

After you enter this command, DB2 displays the sample applications panel, shown in Figure 47.

Choosing option 4 or option 6 on the sample applications panel during Phase 7 invokes the ″Employee Resume″ sample application, which processes CLOB data. Choosing option 5 or option 7 on the sample applications panel during Phase 7 invokes the ″Employee Photo″ sample application, which processes BLOB data. For more information, see “Employee resume and photo scenarios” on page 431.

Working with the sample applications

This topic describes the sample applications. The names of the sample applications have changed for Version 8. Check to make sure you have the authority to run the Version 8 sample programs. For information on granting and revoking DB2 privileges, see Part 3 (Volume 1) of *DB2 Administration Guide*.

Brief scenarios describe how to display, update, add, and delete information using the sample applications. Another scenario describes how to view or change information using a combination of organization and project applications. This scenario contains problem-solving exercises based upon creating and staffing a new department with new projects.

The output from the install verification steps discussed here appears in your *prefix*.SDSNIVPD data set.

Printing the sample application listings

Most of the DB2 sample applications are contained in *prefix*.SDSNSAMP. The source statements contained in *prefix*.SDSNSAMP can be printed using ISPF facilities, IEBPTPCH, or local facilities. The modules making up the SQLCA formatter routine (DSNTIAR, DSNTIAC, DSNTIA1, and DSNTIAM) are not in the *prefix*.SDSNSAMP library. They are provided in object form in *prefix*.SDSNLOAD.

You might not want to print all members of *prefix*.SDSNSAMP because some of the members are large and contain unprintable data. An alternative is to precompile and compile the wanted program by specifying a cross-reference to the precompiler and compiler. This provides a cross-reference for program variables and is current.

Specifying values in the sample application panels

You are prompted for the following information when you run the interactive sample applications:

Sample Applications	Distributed Sample Applications
• MAJOR SYSTEM	• ACTION
• ACTION	• OBJECT
• OBJECT	• SEARCH CRITERIA
• SEARCH CRITERIA	• LOCATION
• DATA	• DATA

These categories must be regarded as a family of values that, used together, specify the task to be performed. For MAJOR SYSTEM, ACTION, OBJECT, and SEARCH CRITERIA, a character code of one or two characters is used as a form of shorthand to indicate the desired criteria. The system provides a list of these codes with their meanings. A valid location name of 1 to 16 characters is used for location. The value for data must be consistent with the data type and length of search criteria. For information on valid location names, see Chapter 12, "Connecting distributed database systems," on page 449.

Major system specifies the major application area. In the sample application, there are two major systems: organization and project. These major systems are implemented in separate transactions to keep the plan sizes reasonable. If you are running the DB2 distributed sample program, organization is the only system; therefore, this criterion is not used.

Action specifies what you want to do with the object (specified on another line of the panel). You can display, update, add (insert), or erase (delete) information about the specified object.

Object specifies the object about which you want information. Normally, the action is associated with the object. Examples of objects are information about an employee (EM) or information about the relationship among departments (DS).

Objects can be specified with the following codes for the organization application:

DE	Department—general department and manager information for department specified
DS	Department structure—hierarchy information for department specified
EM	Employee—information concerning employee specified.

Objects can be specified with the following codes for the project application:

PS	Project structure—information on projects and subprojects
AL	Activity listing—information concerning the different activities that makes up a project
PR	Project—general project information
AS	Activity staffing—information about the employees staffed for activities of specified projects
AE	Activity estimate—information concerning the estimated staffing and time requirements of specified projects.

You are able only to add, update, or erase information about the selected object, although you can search and display based on other criteria. Items that are added or updated can be changed on the screen. Other fields are protected.

Search criteria helps to locate the specific item of information upon which to act. The following codes can be specified for the search criteria field for the organization application:

DI	Department number

DN	Department name
EI	Employee number
EN	Employee name
MI	Manager number
MN	Manager name.

The following codes can be specified for the search criteria for the project application:

DI	Department number
DN	Department name
EI	Employee number
EN	Employee name
PI	Project number
PN	Project name
RI	Responsible person number
RN	Responsible person name.

Location is used only for the distributed application. It describes the location where the action is to take place. If this criterion is left blank, then the local location is assumed.

Data further identifies the search criteria target. The data value specified must be consistent with the data type and length of the search criteria code. If the search criterion is an employee name (EN), manager name (MN), or responsible person name (RN), the value of data must be a person's last name. (See Specifying data values for additional information on how data values can be specified.)

Data values can be specified using either primary selection or secondary selection. *Primary selection* is the data value itself. Only one set of data values fulfills the request. *Secondary selection* allows multiple sets of data values to fulfill the request. A brief summary of the sets of data values appear on the screen. Each summary has an associated line number. To display additional information about a certain line, enter the line number in the DATA field. Secondary selection allows the application to display a set of values and then provides a prompt to select a specific DATA value. For example, you can display information about a *department (DE)* (the OBJECT) with a *department number (DI)* (the SEARCH CRITERIA) with a DATA *value* of D11.

Allowable combinations

The codes cannot be combined indiscriminately. For instance, manager number (MI) is a valid search criterion for a department (DE), but employee number (EI) and project number (PI) cannot be used to locate a department.

You can retrieve data by having the panels prompt you for the proper values. It is not necessary to enter the values one line at a time. If you already know all the values you want, they can be entered at the same time. If the values are only partially entered, you must start with ACTION and enter each value in sequence, not skipping over any values. For example, if you know all the values except OBJECT, only ACTION can be entered. You are prompted for OBJECT. Then you can enter OBJECT, SEARCH CRITERIA, and DATA.

Specifying data values

An entry on the DATA field specifies the choice of SEARCH CRITERIA. The values available for DATA are not limited to a select few as are the values for ACTION and OBJECT. There is a wider choice of DATA values and a variety of ways to express them.

If you know only part of a DATA value (for example, you know the department number begins with D), you can specify it as a *pattern.* The pattern can contain any character string with a special meaning, such as:

- The underscore character, _, represents any single character.
- The percent character, %, represents any string of zero or more characters.

These two special characters can be used in conjunction with other characters to specify a DATA value. Table 106 demonstrates three ways to use these characters to create a DATA value.

Table 106. Searching for data values

Data Value	Search Criteria	Description
%SMITH%	EN (Employee name)	Searches for any last name that contains the word SMITH; for example, BLACKSMITH, SMITHSONIAN, or NESMITHA
E_1	DI (Department number)	Searches for any department number with E in position 1 and 1 in position 3; for example, E71, E21, or EB1
%	Any	All values qualify

The values entered on the SEARCH CRITERIA and DATA fields can choose only one item to be displayed. However, the more usual case is that several items are displayed as a list. When this is the case, a secondary selection can be made by choosing the line number of the item of interest.

Function keys

The bottom line of the panel displays the function keys that are active for that panel:

Function key 2—Resend: If the panel is blanked out (for example, you pressed the CLEAR key) or you want to refresh the panel, press function key 2 to return (resend) the display you were viewing to the terminal.

Function key 3—End: To terminate the application, press function key 3 to clear the screen and continue with other transactions.

Function key 8—Next: Sometimes a display of information is too large to fit on one panel. Press function key 8 to scroll forward (the lines move upward).

Function key 10—Left: Press function key 10 to move the field of vision up one level in the department structure. For instance, in Figure 58 on page 417, Department E01 is shown on the left, and its subdepartments are shown on the right. When you press function key 10, the screen scrolls so that Department E01 is moved from the left side of the panel to the right side and the department to which it reports appears on the left. All other departments that report to the department now on the left also appear along with Department E01 on the right. Function key 10 performs this function only for IMS and CICS samples.

Scenarios

This topic discusses five scenarios for using the sample applications:
- “The project application scenario”
- “The organization application scenario” on page 416
- “The phone application scenario” on page 420
- “The distributed organization application scenario” on page 423
- “Employee resume and photo scenarios” on page 431

How you invoke these applications depends on the environment you are working in; instructions appear in the following places:
- “Starting an application in an ISPF/TSO environment” on page 386 and 397
- “Starting an application in an IMS environment” on page 389
- “Starting an application in a CICS environment” on page 393.

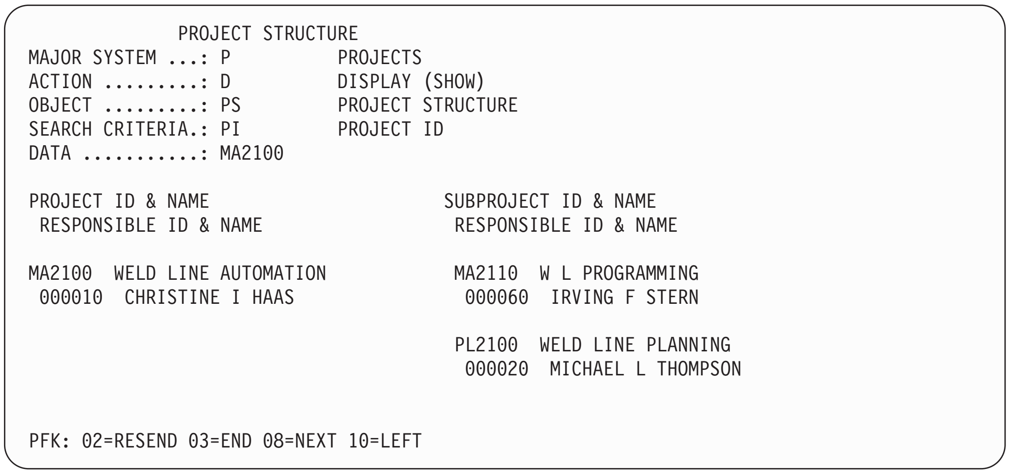
When an application executes, many areas on the display panel might be highlighted. The data you enter might not be highlighted, depending on the type of panel displayed.

The project application scenario

This scenario demonstrates the use of the project application. For example, you can find the person responsible for a project and list the activities assigned to one of its subprojects. Phase 4 (IMS) and Phase 5 (CICS) prepare the programs that you execute.

After you enter the appropriate transaction code, you see the first panel of the project application. Enter the following values:
- On the MAJOR SYSTEM, enter P for project.
- On the ACTION line, enter D for display.
- On the OBJECT line, enter PS for project structure.
- On the SEARCH CRITERIA line, enter PI for project ID.
- On the DATA line, enter MA2100 as the project ID.

The panel below shows the selected project along with its corresponding subprojects.



```
              PROJECT STRUCTURE
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........: D          DISPLAY (SHOW)
OBJECT .........: PS         PROJECT STRUCTURE
SEARCH CRITERIA.: PI         PROJECT ID
DATA ...........: MA2100

PROJECT ID & NAME                      SUBPROJECT ID & NAME
 RESPONSIBLE ID & NAME                  RESPONSIBLE ID & NAME

MA2100  WELD LINE AUTOMATION            MA2110  W L PROGRAMMING
 000010  CHRISTINE I HAAS                000060  IRVING F STERN

                                        PL2100  WELD LINE PLANNING
                                         000020  MICHAEL L THOMPSON


PFK: 02=RESEND 03=END 08=NEXT 10=LEFT
```

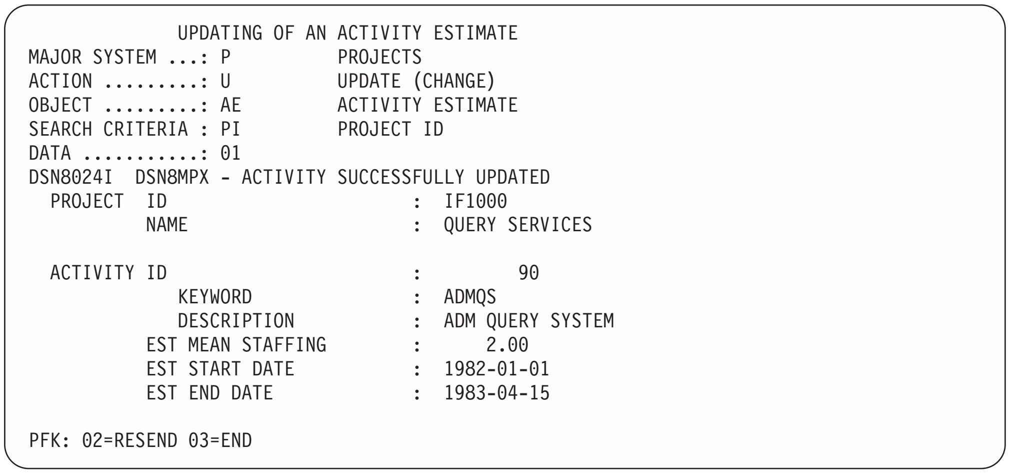
Figure 55. Project application—viewing a project structure

Updating an activity

Suppose you want to update activity information for a project with ID IF1000. Enter the following values:

- On the MAJOR SYSTEM, enter P for project.
- On the ACTION line, enter U for update.
- On the OBJECT line, enter AE for activity estimate.
- On the SEARCH CRITERIA line, enter PI for project ID.
- On the DATA line, enter IF1000 as the project ID.

Press the ENTER key, and a list of project IF1000 activities appears on the panel. Next, choose the activity to be updated. For instance, if you want to update the first activity listed, enter 1 as the DATA value and press the ENTER key. The next panel shows information about the estimated mean staffing requirements of this activity as well as the start and completion dates. To change information about the estimated end date, enter data over the existing information displayed on that input line. After you have verified the change, press ENTER. The next panel displays the updated information, as shown in Figure 56.



```
              UPDATING OF AN ACTIVITY ESTIMATE
MAJOR SYSTEM ...: P          PROJECTS
ACTION .........: U          UPDATE (CHANGE)
OBJECT .........: AE         ACTIVITY ESTIMATE
SEARCH CRITERIA : PI         PROJECT ID
DATA ...........: 01
DSN8024I  DSN8MPX - ACTIVITY SUCCESSFULLY UPDATED
  PROJECT  ID                      :  IF1000
           NAME                    :  QUERY SERVICES

  ACTIVITY ID                      :         90
              KEYWORD              :  ADMQS
              DESCRIPTION          :  ADM QUERY SYSTEM
           EST MEAN STAFFING       :      2.00
           EST START DATE          :  1982-01-01
           EST END DATE            :  1983-04-15

PFK: 02=RESEND 03=END
```

Figure 56. Project application—changes accepted

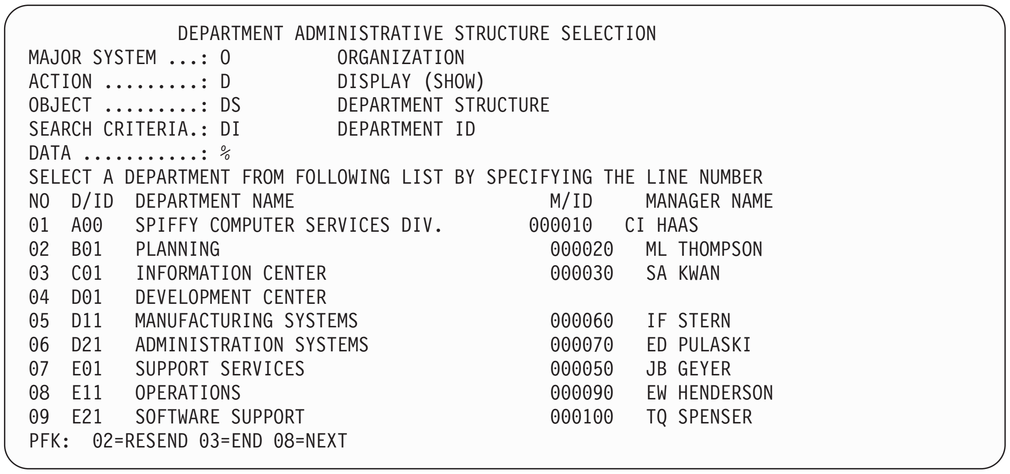
To terminate the project application, press the PF3 key. The APPLICATION TERMINATED message is displayed. If you are using CICS, clear the screen and enter a new transaction code. If you are using IMS, clear the screen and enter a new transaction code or a /FORMAT command.

The organization application scenario

This scenario shows how to use the organization application to display a list of departments within a department and the structure of one of these departments. This application is executed in Phase 4 for IMS and Phase 5 for CICS.

After you enter the appropriate transaction code, you see the first panel of the project application. Enter the following values:
- On the MAJOR SYSTEM, enter O for organization.
- On the ACTION line, enter D for display.
- On the OBJECT line, enter DS for department structure.
- On the SEARCH CRITERIA line, enter DI for department number.
- On the DATA line, enter %, which enables you to display a list of all the departments.

Each department entry is numbered on the far left side of the panel as shown in the Figure 57 on page 417.



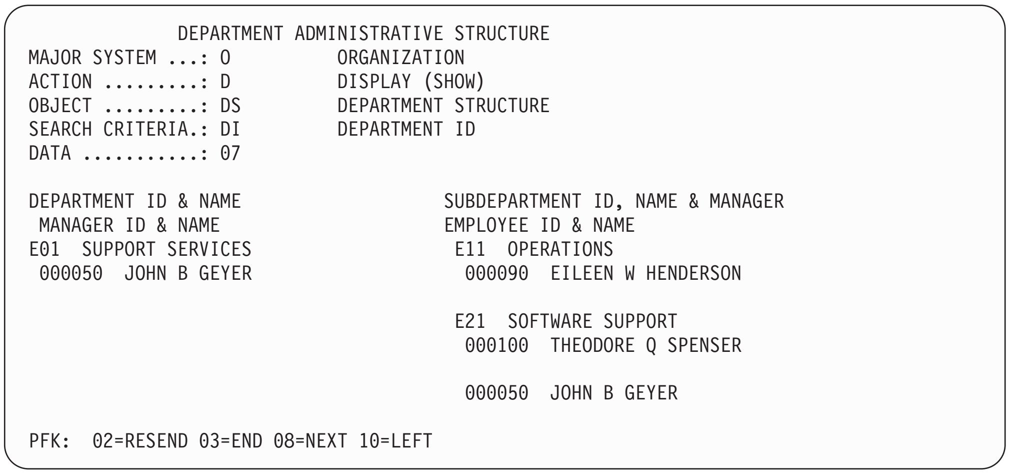
```
              DEPARTMENT ADMINISTRATIVE STRUCTURE SELECTION
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........: D         DISPLAY (SHOW)
OBJECT .........: DS        DEPARTMENT STRUCTURE
SEARCH CRITERIA.: DI        DEPARTMENT ID
DATA ...........: %
SELECT A DEPARTMENT FROM FOLLOWING LIST BY SPECIFYING THE LINE NUMBER
NO  D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME
01  A00   SPIFFY COMPUTER SERVICES DIV.        000010   CI HAAS
02  B01   PLANNING                               000020   ML THOMPSON
03  C01   INFORMATION CENTER                     000030   SA KWAN
04  D01   DEVELOPMENT CENTER
05  D11   MANUFACTURING SYSTEMS                  000060   IF STERN
06  D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
07  E01   SUPPORT SERVICES                       000050   JB GEYER
08  E11   OPERATIONS                             000090   EW HENDERSON
09  E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
PFK:  02=RESEND 03=END 08=NEXT
```

Figure 57. Organization application—viewing a list of departments

To retrieve further information, specify a line number as a data value. This method is called secondary selection. Secondary selection provides prompts to aid in finding the information to be displayed, added, erased, or updated. If only one entry possibility exists, secondary selection is not offered.

To view an individual department structure, specify the line entry number (secondary selection) of the department as a new DATA value. For example, to view the structure of Department E01, specify a data value of 7 on the DATA entry line (7 is the line number of the entry for Department E01).

The result of entering the data value of 7 is a display of Department E01 and its departments as shown in Figure 58. The department manager for E01 is listed on the left, and the departments of E01 are listed on the right. Employees of E01 are listed below the subdepartments of E01.



```
              DEPARTMENT ADMINISTRATIVE STRUCTURE
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........: D         DISPLAY (SHOW)
OBJECT .........: DS        DEPARTMENT STRUCTURE
SEARCH CRITERIA.: DI        DEPARTMENT ID
DATA ...........: 07

DEPARTMENT ID & NAME                   SUBDEPARTMENT ID, NAME & MANAGER
 MANAGER ID & NAME                     EMPLOYEE ID & NAME
E01  SUPPORT SERVICES                   E11  OPERATIONS
 000050  JOHN B GEYER                    000090  EILEEN W HENDERSON

                                        E21  SOFTWARE SUPPORT
                                         000100  THEODORE Q SPENSER

                                         000050  JOHN B GEYER

PFK:  02=RESEND 03=END 08=NEXT 10=LEFT
```

Figure 58. Organization application—viewing a department structure

Starting a new operation

You can start a new operation on the organization application by moving the cursor to the D on the ACTION line and retaining the D or changing it to a different action (add, erase, or update). Follow the displayed options to perform your selected action.

Alternatively, you can leave the organization application by pressing the PF3 key. If you are using CICS, enter the transaction code. If you are using IMS, clear the screen and enter the /FORMAT command to select the project application. In either case, to proceed with a different operation, select a different ACTION, OBJECT, and so forth.

Adding a new department

Adding a department falls under the organization major system. Start the organization application as described in "Starting a new operation" on page 417 and enter the following values:

- On the MAJOR SYSTEM, enter O for organization
- On the ACTION line, enter A for add (insert)
- On the OBJECT line, enter DE for the department that is to be added
- On the SEARCH CRITERIA line, enter DI for department ID
- On the DATA line, enter C11, the specific department number.

Next, you can enter the details of the new department. The four department fields are department number, department name, manager number, and administration department number. Enter:

- INFORMATION SERVICES for department name
- 000130 for manager number
- C01 for the administration department number.

Press ENTER to display the panel shown in Figure 59. The panel shows the successful addition of the new department.

```
              ADDING A NEW DEPARTMENT
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........: A         ADD (INSERT)
OBJECT .........: DE        DEPARTMENT
SEARCH CRITERIA.: DI        DEPARTMENT ID
DATA ...........: C11
DSN8012I  DSN8MPE - DEPARTMENT SUCCESSFULLY ADDED
  DEPARTMENT ID                    :  C11
             NAME                  :  INFORMATION SERVICES
             MANAGER ID            :  000130
             ADMIN DEP ID          :  C01
  MANAGER    ID                    :  000130
             FIRST NAME            :  DOLORES
             MIDDLE INITIAL        :  M
             LAST NAME             :  QUINTANA
             WORK DEPT ID          :  C01

PFK: 02=RESEND 03=END
```

Figure 59. Organization application—adding a department

Deleting an entry

Deleting an entry in the department table is also a function of the organization major system. Following the process outlined in "Starting a new operation" on page 417, replace the following values on the panel currently displayed on your screen:

- On the MAJOR SYSTEM, enter O for organization.
- On the ACTION line, enter E for erase.
- On the OBJECT line, enter DE for department.
- On the SEARCH CRITERIA line, enter DI for department ID.
- On the DATA line, enter C11 for department name.

Press ENTER to display the panel shown in Figure 60.

```
              ERASING A DEPARTMENT
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........: E         ERASE (REMOVE)
OBJECT .........: DE        DEPARTMENT
SEARCH CRITERIA.: DI        DEPARTMENT ID
DATA ...........: C11
PRESS ENTER TO ERASE A DEPARTMENT
  DEPARTMENT ID                   :  C11
             NAME                 :  INFORMATION SERVICES
             MANAGER ID           :  000130
             ADMIN DEP ID         :  C01
  MANAGER    ID                   :  000130
             FIRST NAME           :  DOLORES
             MIDDLE INITIAL       :  M
             LAST NAME            :  QUINTANA
             WORK DEPT ID         :  C01

PFK: 02=RESEND 03=END
```

Figure 60. Organization application—deletion successful

Press ENTER again to verify the erase action. The following message appears on the panel:

```
DSN8013I    csect DEPARTMENT SUCCESSFULLY ERASED
```

Transferring

The procedure for transferring one employee to another department and replacing that employee involves several steps. In this scenario, John B. Geyer (manager of the department for Support Services) is transferred to the staff of Spiffy Computer Service Division. Bruce Adamson is assigned as manager of Support Services.

To move Adamson into his new position as manager of Support Services, you must determine his employee number. Transferring an employee is a function of the organization major system. Start the organization application as described in "Starting a new operation" on page 417, and enter the following values:

- On the MAJOR SYSTEM, enter O for organization.
- On the ACTION line, enter D for display.
- On the OBJECT line, enter EM for employee.
- On the SEARCH CRITERIA line, enter EN for employee name.
- On the DATA line, enter ADAMSON as the specific employee name.

Press ENTER to display the panel showing that Adamson's employee number is 000150.

The next step is for you to change the manager number for the Support Services department to Adamson's number, 000150. But first you must find the Support Services department. To do this, change ACTION to U (update), OBJECT to DE (department), and SEARCH CRITERIA to DN (department name). Change DATA to %SUPPORT% to specify any department with the word SUPPORT in it.

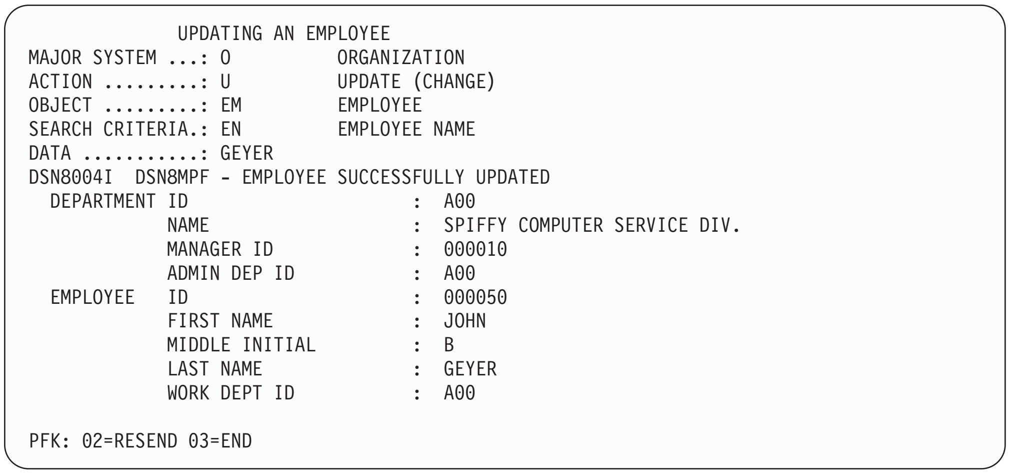
Press ENTER, and a list of departments with support in their name is displayed. Support Services has line number 01. Enter this number at DATA. (The leading zero is not needed.)

Press ENTER to display the next panel. The only values that can be changed are department name, manager ID, and administration department ID. Enter Adamson's employee number in the Support Services department after MANAGER ID. At this point, the data on the manager still pertains to Geyer.

Press ENTER to display the panel that shows Adamson as manager of Support Services. The work department ID shown (D11) is still Adamson's old number. To change Adamson's work department ID, enter EM for OBJECT, enter EI for SEARCH CRITERIA, and change the employee number to 000150 for DATA.

Press ENTER to display the employee information on Adamson. Now that information on Adamson can be updated. The fields that can be changed are employee first name, middle initial, last name, and work department ID. Enter the middle initial for Adamson, which was not in the database, and the department number E01. Press ENTER, and the information on Adamson is updated, including his new department number.

The final step is to move Geyer to the correct department. Change the SEARCH CRITERIA and DATA to EN and GEYER, respectively. Press ENTER to obtain the next panel. The employee ID, name, and work department ID can be changed on this panel. However, the only change necessary in this case is to change Geyer's work department ID to his new one, A00. The panel in Figure 61 shows the completed entry.



```
              UPDATING AN EMPLOYEE
MAJOR SYSTEM ...: O         ORGANIZATION
ACTION .........: U         UPDATE (CHANGE)
OBJECT .........: EM        EMPLOYEE
SEARCH CRITERIA.: EN        EMPLOYEE NAME
DATA ...........: GEYER
DSN8004I  DSN8MPF - EMPLOYEE SUCCESSFULLY UPDATED
  DEPARTMENT ID                     :  A00
             NAME                   :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID             :  000010
             ADMIN DEP ID           :  A00
  EMPLOYEE   ID                     :  000050
             FIRST NAME             :  JOHN
             MIDDLE INITIAL         :  B
             LAST NAME              :  GEYER
             WORK DEPT ID           :  A00

PFK: 02=RESEND 03=END
```

Figure 61. Organization application—employee data update completed

To terminate the application and return to the beginning of the operation, press the PF3 key.

The phone application scenario

The phone application is used in phase 2 (batch mode), phase 3 (CAF), and interactively in phase 4 (IMS) and phase 5 (CICS).

The phone application retrieves information from a phone directory and updates employee phone numbers. The phone directory consists of data from a combination (join) of the employee table (DSN8810.EMP) and the department table (DSN8810.DEPT). This joined view is called VPHONE. The program also uses a second view called VEMPLP to update the employee table, which does not affect a view that joins tables.

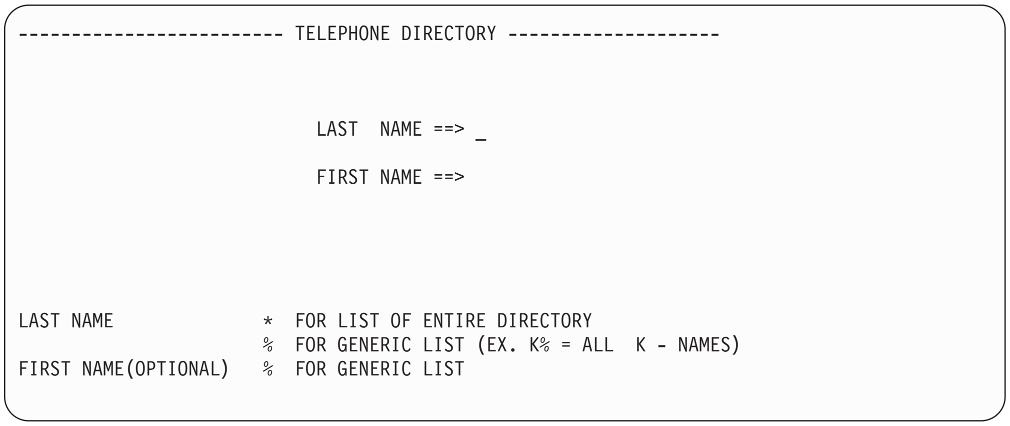
The phone application is designed to operate in batch and interactively in ISPF/TSO, IMS, and CICS. Table 107 describes the environments in which each phone application operates and the language in which each is written. For information on how to invoke the CAF application in an ISPF/TSO environment, see Figure 47 on page 386.

Table 107. Phone programs

Environment	Language	Name
ISPF/TSO	COBOL	DSN8SC3
ISPF/TSO	PL/I	DSN8SP3
IMS	PL/I	DSN8IP3
CICS	PL/I	DSN8CP3
batch	COBOL	DSN8BC3
batch	Fortran	DSN8BF3
batch	PL/I	DSN8BP3
batch	C	DSN8BD3

Phone application panels

The panels for the phone application are the same, whether IMS or CICS is used. In both cases, information is managed interactively beginning with the panel shown in Figure 62.



```
------------------------- TELEPHONE DIRECTORY --------------------



                            LAST  NAME ==> _

                            FIRST NAME ==>







LAST NAME               *  FOR LIST OF ENTIRE DIRECTORY
                        %  FOR GENERIC LIST (EX. K% = ALL  K - NAMES)
FIRST NAME(OPTIONAL)    %  FOR GENERIC LIST
```

Figure 62. Telephone application—first display

On this panel, enter the first and last name of the employee whose telephone number you want to view or change. To see an entire listing of employee numbers, put an * next to the LAST NAME input line. If only part of a first or last name is known, use the percent character (%) to qualify the list of names to appear in the directory. For example, entering K% on the LAST NAME input line calls a list of the telephone numbers of all employees whose last name begins with a K. Similarly, the first name can be qualified.

To keep this sample program as simple as possible and to allow updating, scrolling is not used with the IMS and CICS versions. Scrolling is used with the ISPF/CAF version. Only the first panel of selected names and phone numbers can be displayed. The second panel is the Telephone Directory itself. The employee telephone number is highlighted. To update an employee telephone number, type over the highlighted number and press ENTER. To update a phone number listed under the name Heather A Nicholls, specify NICHOL% when you are not sure if there are one or two LS in Nicholls.

Press ENTER to display the panel on which the phone number is highlighted. Suppose you want to change the phone number from 1793 to 1795. Just type over the number to be changed. You can type over as many numbers as appear listed in the current display. After you press ENTER, you get a message confirming the updated phone number. The panel in Figure 63 shows the updated panel.

```
-------------------------- TELEPHONE DIRECTORY --------------------

 FIRST NAME MID LAST NAME  PHONE  EMPL WORK WORKDEPT
           INIT             NO    NO  DEPT NAME

HEATHER  A      NICHOLLS     1795 000140 C01 INFORMATION CENTER
.
.
.
```

Figure 63. Telephone application—updated display

Using the phone application under batch

The sample batch phone applications are provided in Fortran (DSN8BF3), COBOL (DSN8BC3), C (DSN8BD3), and PL/I (DSN8BP3).

If you want to update an employee phone number, create a data set that contains information about the phone number to be updated. This data set works in combination with another data set that contains JCL for processing information. The first data set consists of card images in the format shown in Table 108.

Table 108. Format of phone application data set

Column	Description
1	ACTION—U for update, L for list
2	Employee last name
17	Employee first name
29	Employee number
35	New phone number

The ACTION code in this card image indicates whether an employee number is to be updated (U) or listed (L). When updating an employee phone number, only the employee number and the new phone number are specified in the data set. When listing phone numbers, the last name must be specified. Specifying the first name is optional. The * and % can be used with the ACTION code just as they are used with the panels.

Figure 64 on page 423 shows an example of an update data set and a list data set. Each time a number is listed or updated, a new data set is created containing a card image like the one in Figure 64. The first card in the data set shows the phone number of employee number 000140 being updated (U) to 6767. The second card shows a list (L) for Heather Nicholls. The last card shows a list (L) of all employees whose first names begin with the letters MAR. The example shows the letters MAR followed by a % in the first name column to indicate that only those employees whose first names begin with MAR are to be listed.

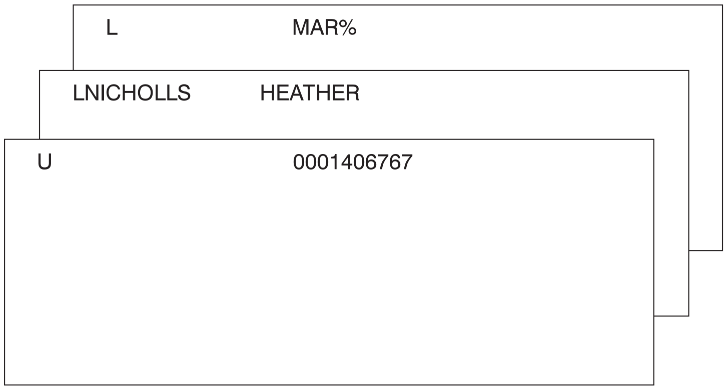


Figure 64. Example of a card image data set

The other data set that contains the JCL is supplied with DB2 and is contained in DSNTEJ2P, which is part of *prefix*.SDSNSAMP. Figure 65 shows the data set that contains the JCL with the card image data sets embedded.

```
//PH02PS05 EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//REPORT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//CARDIN DD *
U                           0001406767
LNICHOLLS       HEATHER
L               MAR%
//SYSTSIN DD *
 DSN SYSTEM(DSN)
 RUN  PROGRAM(DSN8BP3) PLAN(DSN8BP81) LIB('prefix.RUNLIB.LOAD')
 END
```

Figure 65. The job control language data set

The complete data set can be submitted to the system either through a card reader or from a terminal through TSO. Figure 66 on page 423 is an example of the batch output.

```
---------------------------- TELEPHONE DIRECTORY -----------------------

LAST NAME FIRST NAME INITIAL PHONE  EMPLOYEE WORK WORK
                             NUMBER NUMBER   DEPT DEPT NAME

QUINTANA  DOLORES       M    6767   000130   C01  INFORMATION CENTER
NICHOLLS  HEATHER       A    1793   000140   C01  INFORMATION CENTER
SCOUTTEN  MARILYN       S    1682   000180   D11  MANUFACTURING SYSTEMS
PEREZ     MARIA         L    9001   000270   D21  ADMINISTRATIVE SYSTEMS
```

Figure 66. Example of the phone application batch output

The distributed organization application scenario

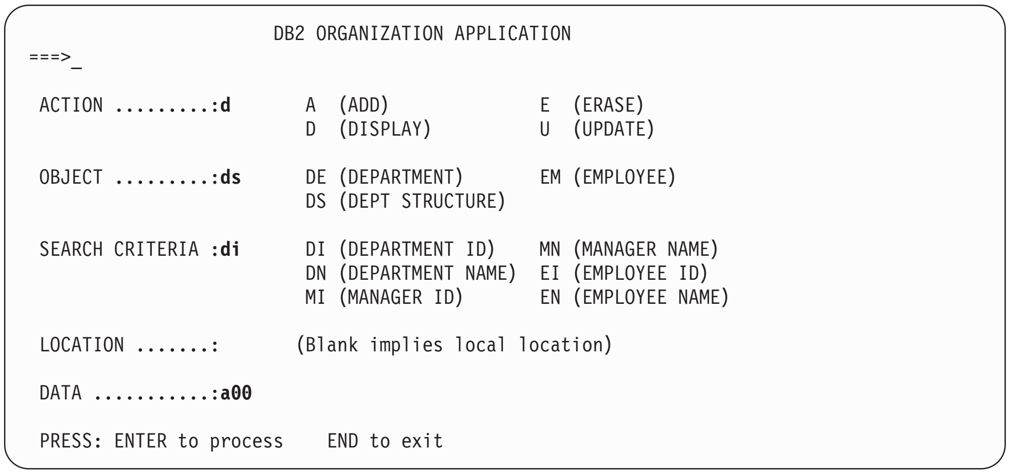
This scenario shows how to use the distributed organization application to display a department structure, display department information, and update a department at a local location. It also shows how to erase and add an employee at a remote location. This application is executed in Phase 6. The application accesses distributed data with DRDA access.

The department information (DEPT table) is shared by all locations. If you make changes to DEPT table at one location, the DEPT tables at the other locations are updated at the same time. The employee information (EMP table) is unique to each location, containing only the employees that work at that particular location.

After you enter the appropriate transaction code, you see the first panel of the organization application.

Displaying department structure at the local location

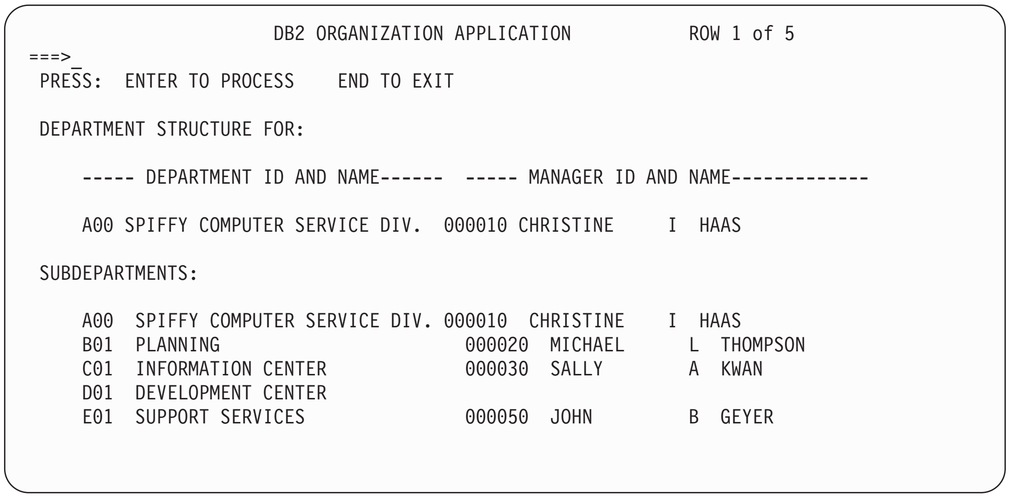
To display a department structure, enter the following values:
- On the ACTION line, enter D for display.
- On the OBJECT line, enter DS for department structure.
- On the SEARCH CRITERIA line, enter DI for department number.
- On the LOCATION line, leave blank, indicating local location.
- On the DATA line, enter A00 for department number.



```
                       DB2 ORGANIZATION APPLICATION
===>_

 ACTION .........:d       A  (ADD)             E  (ERASE)
                          D  (DISPLAY)         U  (UPDATE)

 OBJECT .........:ds      DE (DEPARTMENT)      EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:       (Blank implies local location)

 DATA ...........:a00

 PRESS: ENTER to process    END to exit
```

Figure 67. Starting the distributed organization application

Press the ENTER key. The panel below shows the structure of the department requested.



```
                       DB2 ORGANIZATION APPLICATION          ROW 1 of 5
===>_
 PRESS:  ENTER TO PROCESS    END TO EXIT

 DEPARTMENT STRUCTURE FOR:

     ----- DEPARTMENT ID AND NAME------  ----- MANAGER ID AND NAME-------------

     A00 SPIFFY COMPUTER SERVICE DIV.  000010 CHRISTINE     I  HAAS

 SUBDEPARTMENTS:

     A00  SPIFFY COMPUTER SERVICE DIV. 000010  CHRISTINE    I  HAAS
     B01  PLANNING                       000020  MICHAEL      L  THOMPSON
     C01  INFORMATION CENTER             000030  SALLY        A  KWAN
     D01  DEVELOPMENT CENTER
     E01  SUPPORT SERVICES               000050  JOHN         B  GEYER
```

Figure 68. Displaying department structure

Press ENTER or END to exit.

Displaying department information at the local location

To display department information, enter the following values:
- On the ACTION line, enter D for display.
- On the OBJECT line, enter DE for department.
- On the SEARCH CRITERIA line, enter DI for department number.
- On the LOCATION line, leave blank, indicating local location.
- On the DATA line, enter A00 for department number.

```
                       DB2 ORGANIZATION APPLICATION
===>_

 ACTION .........:d       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:de      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:        (Blank implies local location)

 DATA ...........:a00

 PRESS: ENTER to process    END to exit
```

Figure 69. Starting the distributed organization application

Press the ENTER key. The panel below shows the department information requested.

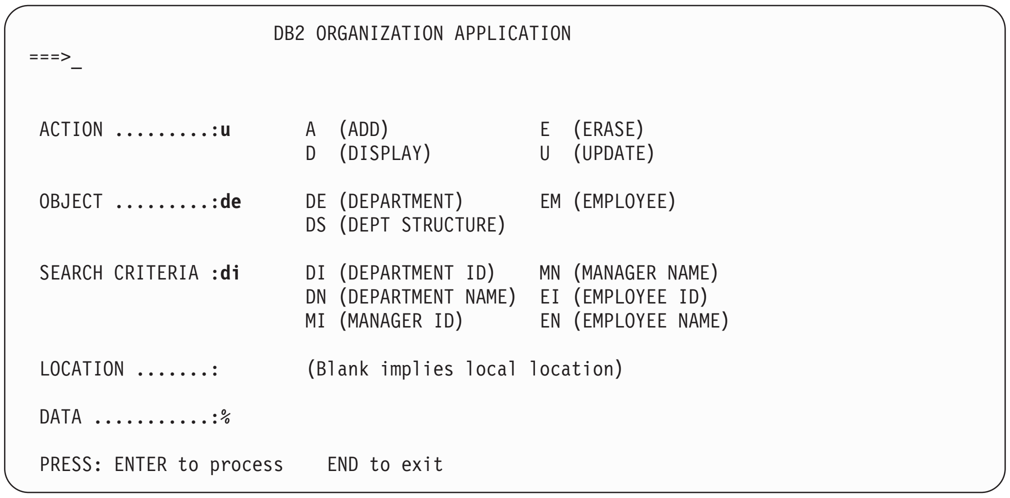
```
                       DB2 ORGANIZATION APPLICATION
===>_

                            DISPLAY A DEPARTMENT

  DEPARTMENT ID                    :  A00
             NAME                  :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID            :  000010
             ADMIN DEP ID          :  A00
             LOCATION              :

  MANAGER    ID                    :  000010
             FIRST NAME            :  CHRISTINE
             MIDDLE INITIAL        :  I
             LAST NAME             :  HAAS
             WORK DEPT ID          :  A00


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 70. Displaying department information

Press ENTER or END to exit.

Updating a department at the local location

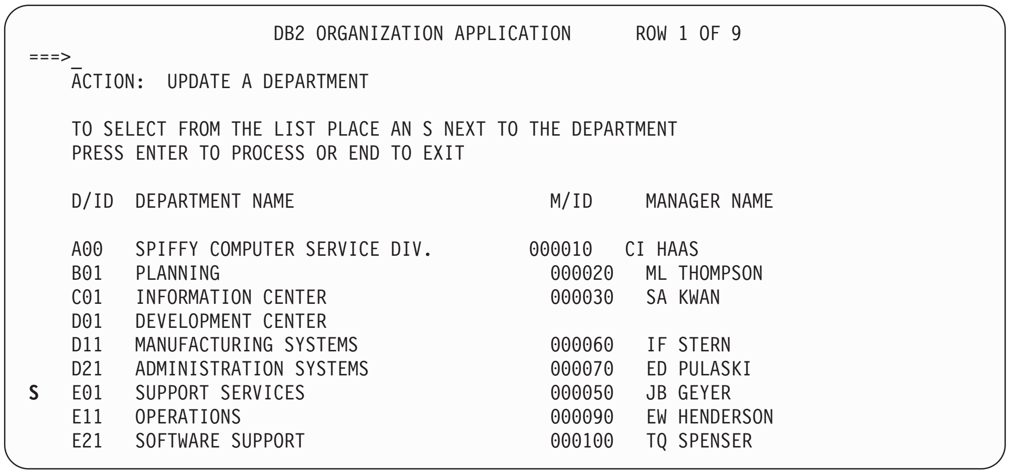
To update department information, enter the following values:
- On the ACTION line, enter U for update.
- On the OBJECT line, enter DE for department.
- On the SEARCH CRITERIA line, enter DI for department number.
- On the LOCATION line, leave blank, indicating local location.
- On the DATA line, enter % which enables you to display a list of all the departments.



```
                       DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:u       A  (ADD)             E  (ERASE)
                          D  (DISPLAY)         U  (UPDATE)

 OBJECT .........:de      DE (DEPARTMENT)      EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :di      DI (DEPARTMENT ID)   MN (MANAGER NAME)
                          DN (DEPARTMENT NAME) EI (EMPLOYEE ID)
                          MI (MANAGER ID)      EN (EMPLOYEE NAME)

 LOCATION .......:        (Blank implies local location)

 DATA ...........:%

 PRESS: ENTER to process    END to exit
```

Figure 71. Starting the distributed organization application

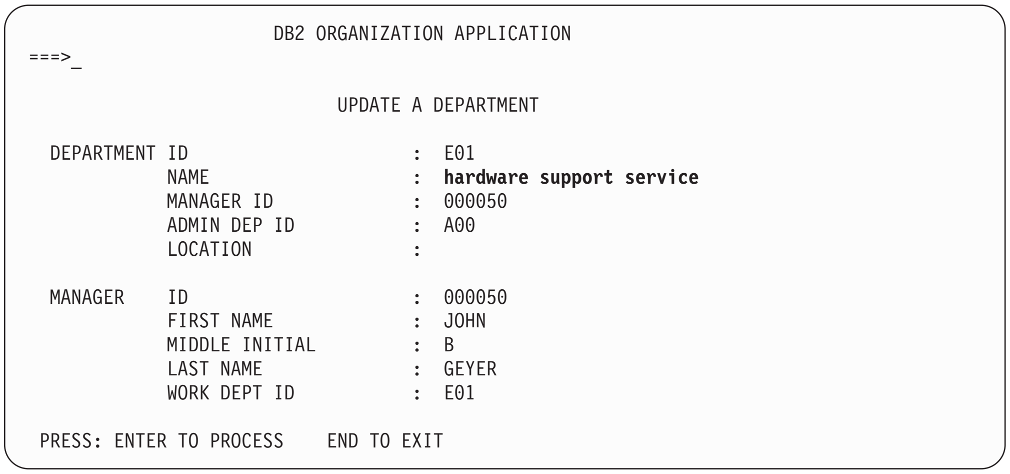
Press the ENTER key. The panel below lists the departments that can be updated. Select the department to be updated by putting an S in the left margin by the department number.



```
                       DB2 ORGANIZATION APPLICATION      ROW 1 OF 9
===>_
    ACTION:  UPDATE A DEPARTMENT

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE DEPARTMENT
    PRESS ENTER TO PROCESS OR END TO EXIT

    D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME

    A00   SPIFFY COMPUTER SERVICE DIV.         000010   CI HAAS
    B01   PLANNING                               000020   ML THOMPSON
    C01   INFORMATION CENTER                     000030   SA KWAN
    D01   DEVELOPMENT CENTER
    D11   MANUFACTURING SYSTEMS                  000060   IF STERN
    D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
S   E01   SUPPORT SERVICES                       000050   JB GEYER
    E11   OPERATIONS                             000090   EW HENDERSON
    E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
```

Figure 72. Selecting a department to be updated

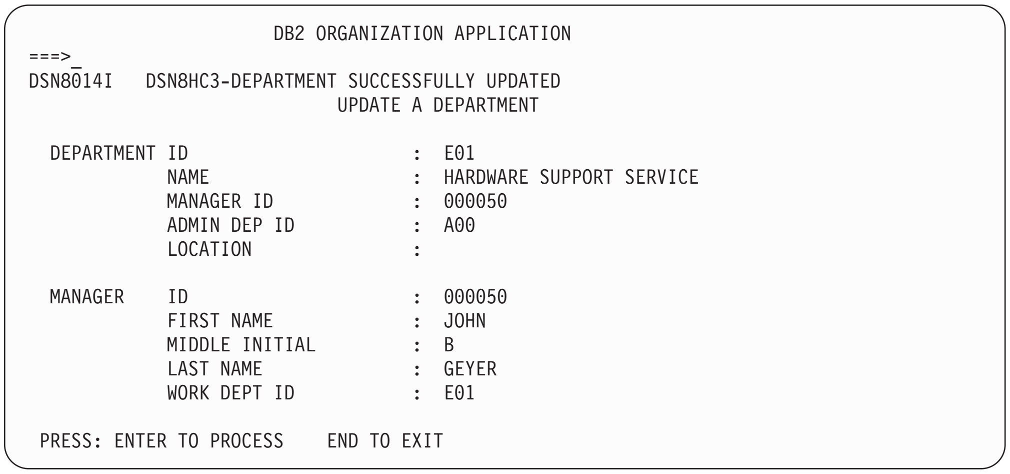
Press the ENTER key. The panel below displays the information relevant to the selected department. Enter the information you want to update on this panel; in this case, enter the name of the department.



```
                       DB2 ORGANIZATION APPLICATION
===>_

                             UPDATE A DEPARTMENT

  DEPARTMENT ID                     :  E01
             NAME                   :  hardware support service
             MANAGER ID             :  000050
             ADMIN DEP ID           :  A00
             LOCATION               :

  MANAGER    ID                     :  000050
             FIRST NAME             :  JOHN
             MIDDLE INITIAL         :  B
             LAST NAME              :  GEYER
             WORK DEPT ID           :  E01

 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 73. Updating a department

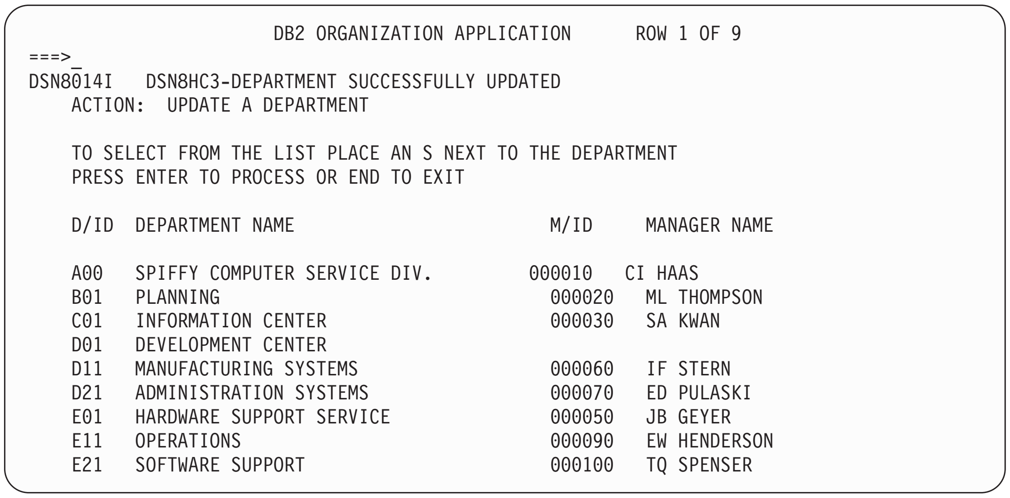
Press the ENTER key to process the updated information. A message appears on this panel that states the update was successful.



```
                       DB2 ORGANIZATION APPLICATION
===>_
DSN8014I   DSN8HC3-DEPARTMENT SUCCESSFULLY UPDATED
                             UPDATE A DEPARTMENT

  DEPARTMENT ID                     :  E01
             NAME                   :  HARDWARE SUPPORT SERVICE
             MANAGER ID             :  000050
             ADMIN DEP ID           :  A00
             LOCATION               :

  MANAGER    ID                     :  000050
             FIRST NAME             :  JOHN
             MIDDLE INITIAL         :  B
             LAST NAME              :  GEYER
             WORK DEPT ID           :  E01

 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 74. Update successfully processed

Press ENTER to return to the previous panel or END to exit. If you return to the previous panel, you can now select another department to update, or press ENTER or END to exit. The same message appears on this panel, indicating the update was successful.

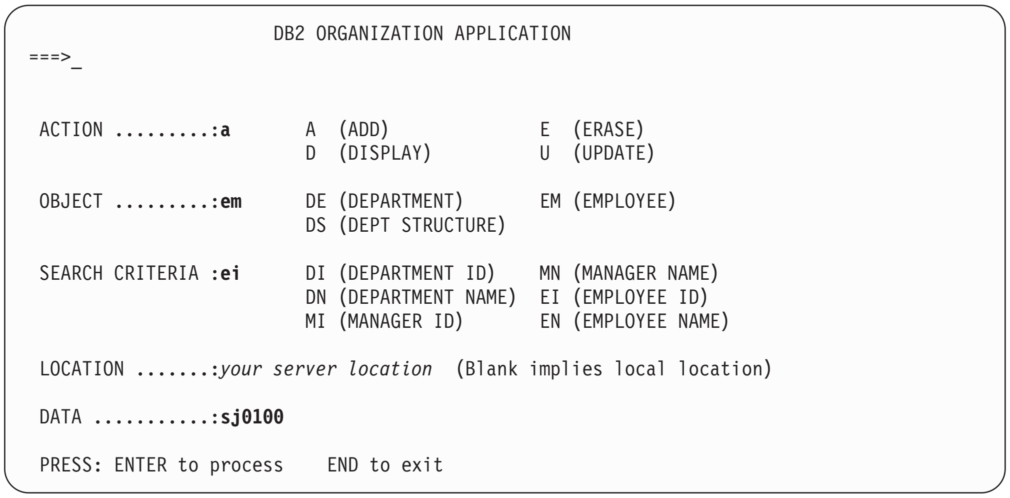


```
                     DB2 ORGANIZATION APPLICATION      ROW 1 OF 9
===>_
DSN8014I   DSN8HC3-DEPARTMENT SUCCESSFULLY UPDATED
    ACTION:  UPDATE A DEPARTMENT

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE DEPARTMENT
    PRESS ENTER TO PROCESS OR END TO EXIT

    D/ID  DEPARTMENT NAME                        M/ID     MANAGER NAME

    A00   SPIFFY COMPUTER SERVICE DIV.         000010   CI HAAS
    B01   PLANNING                               000020   ML THOMPSON
    C01   INFORMATION CENTER                     000030   SA KWAN
    D01   DEVELOPMENT CENTER
    D11   MANUFACTURING SYSTEMS                  000060   IF STERN
    D21   ADMINISTRATION SYSTEMS                 000070   ED PULASKI
    E01   HARDWARE SUPPORT SERVICE               000050   JB GEYER
    E11   OPERATIONS                             000090   EW HENDERSON
    E21   SOFTWARE SUPPORT                       000100   TQ SPENSER
```

Figure 75. Department successfully updated

Adding an employee at a remote location

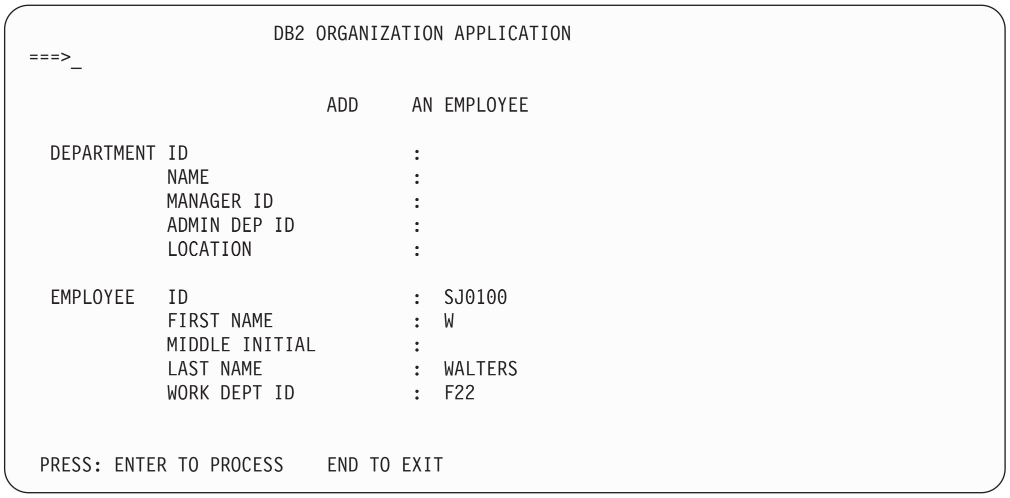
To add an employee at a remote location, enter the following values:
- On the ACTION line, enter A for add.
- On the OBJECT line, enter EM for employee.
- On the SEARCH CRITERIA line, enter EI for employee number.
- On the LOCATION line, enter *your server location* for the remote location.
- On the DATA line, enter SJ0100 which indicate the employee ID of the employee to be added.



```
                     DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:a       A  (ADD)            E  (ERASE)
                          D  (DISPLAY)        U  (UPDATE)

 OBJECT .........:em      DE (DEPARTMENT)      EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :ei      DI (DEPARTMENT ID)   MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:your server location  (Blank implies local location)

 DATA ...........:sj0100

 PRESS: ENTER to process    END to exit
```

Figure 76. Starting the distributed organization application

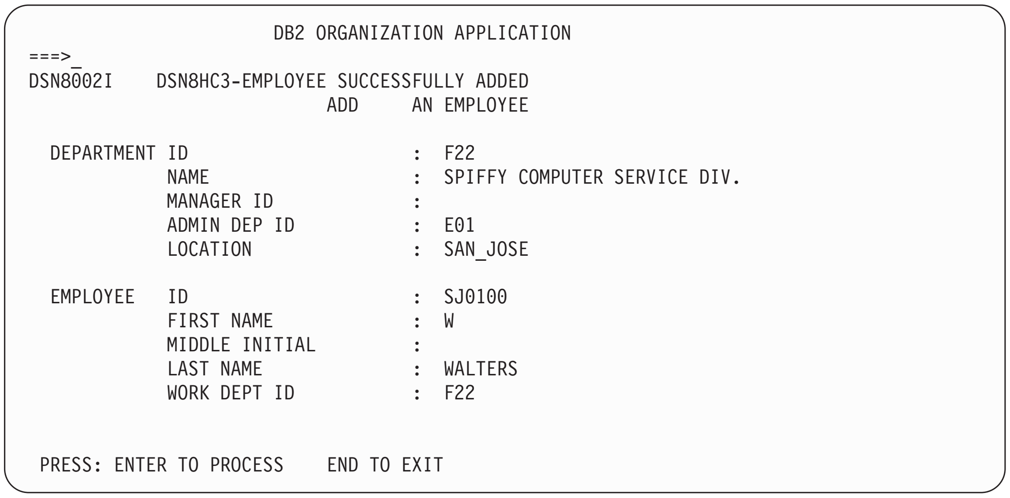
Press the ENTER key. The panel below allows you to only enter information about the employee. The department information on the panel is protected. Enter the necessary information about the employee.



```
                       DB2 ORGANIZATION APPLICATION
===>_

                            ADD     AN EMPLOYEE

  DEPARTMENT ID                    :
             NAME                  :
             MANAGER ID            :
             ADMIN DEP ID          :
             LOCATION              :

  EMPLOYEE   ID                    :  SJ0100
             FIRST NAME            :  W
             MIDDLE INITIAL        :
             LAST NAME             :  WALTERS
             WORK DEPT ID          :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 77. Employee to be added

Press the ENTER key to process or END to exit. If you press the ENTER key, a message appears on the panel indicating that the employee has been added.



```
                       DB2 ORGANIZATION APPLICATION
===>_
DSN8002I    DSN8HC3-EMPLOYEE SUCCESSFULLY ADDED
                            ADD     AN EMPLOYEE

  DEPARTMENT ID                    :  F22
             NAME                  :  SPIFFY COMPUTER SERVICE DIV.
             MANAGER ID            :
             ADMIN DEP ID          :  E01
             LOCATION              :  SAN_JOSE

  EMPLOYEE   ID                    :  SJ0100
             FIRST NAME            :  W
             MIDDLE INITIAL        :
             LAST NAME             :  WALTERS
             WORK DEPT ID          :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

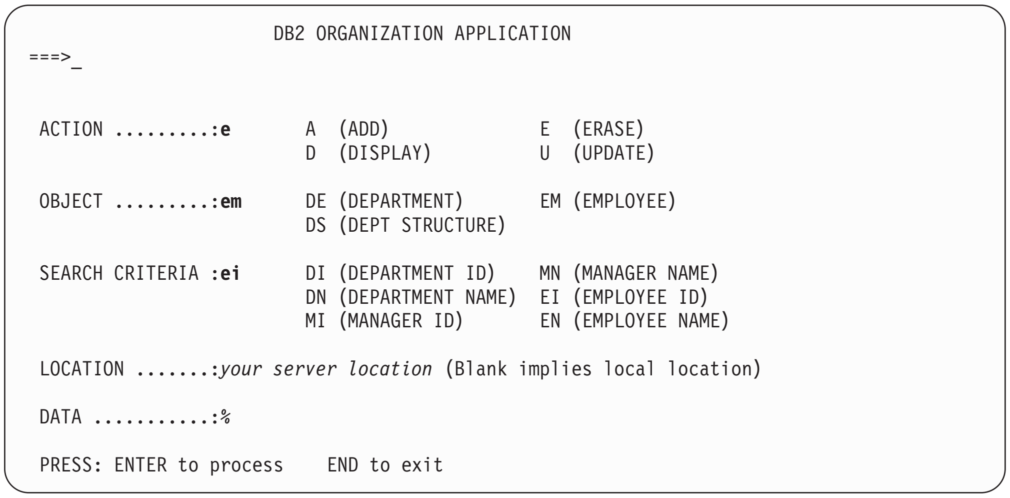
Figure 78. Employee successfully added

Press the ENTER key to return to the selection panel or END to exit.

Erasing an employee at a remote location

To erase an employee at a remote location, enter the following values:

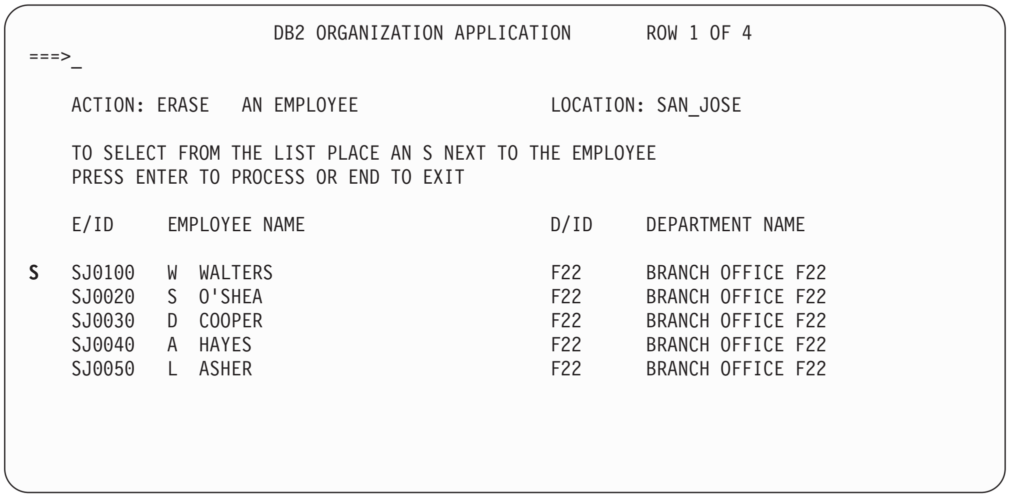
- On the ACTION line, enter E for erase.
- On the OBJECT line, enter EM for employee.
- On the SEARCH CRITERIA line, enter EI for employee number.
- On the LOCATION line, enter *your server location* for the remote location.
- On the DATA line, enter % which enables you to display a list of all the employees.



```
                       DB2 ORGANIZATION APPLICATION
===>_


 ACTION .........:e       A  (ADD)              E  (ERASE)
                          D  (DISPLAY)          U  (UPDATE)

 OBJECT .........:em      DE (DEPARTMENT)       EM (EMPLOYEE)
                          DS (DEPT STRUCTURE)

 SEARCH CRITERIA :ei      DI (DEPARTMENT ID)    MN (MANAGER NAME)
                          DN (DEPARTMENT NAME)  EI (EMPLOYEE ID)
                          MI (MANAGER ID)       EN (EMPLOYEE NAME)

 LOCATION .......:your server location (Blank implies local location)

 DATA ...........:%

 PRESS: ENTER to process    END to exit
```

Figure 79. Starting the distributed organization application

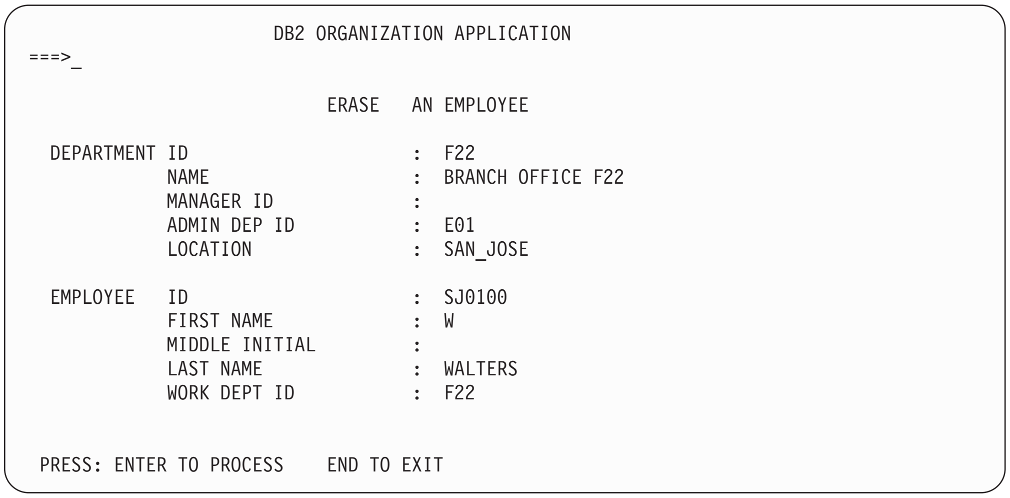
Press the ENTER key. The panel below lists the employees that can be erased. Select the employee to be erased by putting an S in the left margin by the employee ID.



```
                       DB2 ORGANIZATION APPLICATION       ROW 1 OF 4
===>_

    ACTION: ERASE   AN EMPLOYEE                  LOCATION: SAN_JOSE

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE EMPLOYEE
    PRESS ENTER TO PROCESS OR END TO EXIT

    E/ID     EMPLOYEE NAME                       D/ID     DEPARTMENT NAME

S   SJ0100   W  WALTERS                          F22      BRANCH OFFICE F22
    SJ0020   S  O'SHEA                           F22      BRANCH OFFICE F22
    SJ0030   D  COOPER                           F22      BRANCH OFFICE F22
    SJ0040   A  HAYES                            F22      BRANCH OFFICE F22
    SJ0050   L  ASHER                            F22      BRANCH OFFICE F22
```

Figure 80. Selecting an employee at a remote location

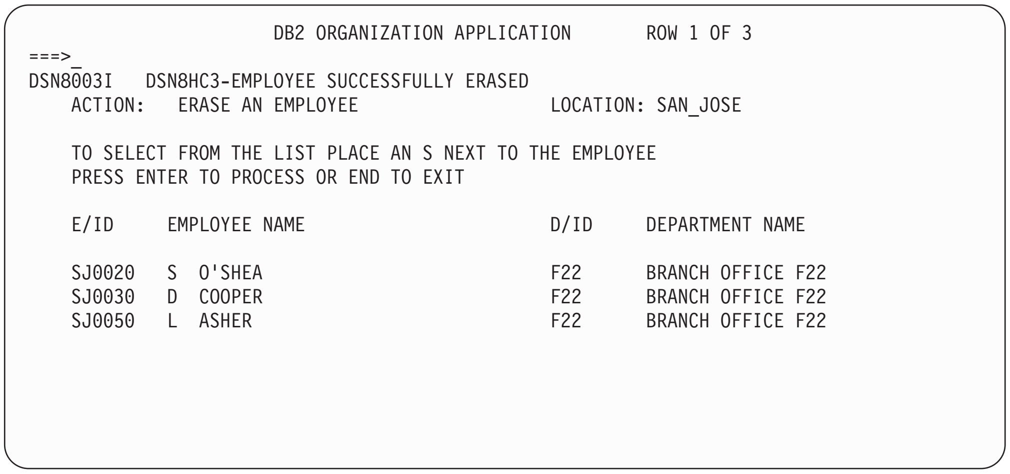
Press the ENTER key. The panel below displays the information relevant to the selected employee that is to be erased.



```
                       DB2 ORGANIZATION APPLICATION
===>_

                            ERASE   AN EMPLOYEE

  DEPARTMENT ID                     :  F22
             NAME                   :  BRANCH OFFICE F22
             MANAGER ID             :
             ADMIN DEP ID           :  E01
             LOCATION               :  SAN_JOSE

  EMPLOYEE   ID                     :  SJ0100
             FIRST NAME             :  W
             MIDDLE INITIAL         :
             LAST NAME              :  WALTERS
             WORK DEPT ID           :  F22


 PRESS: ENTER TO PROCESS    END TO EXIT
```

Figure 81. Employee to be erased

Press the ENTER key to erase the employee information. A message appears on the panel stating the employee has been successfully erased. You can now erase another employee or press ENTER or END to exit.



```
                       DB2 ORGANIZATION APPLICATION       ROW 1 OF 3
===>_
DSN8003I   DSN8HC3-EMPLOYEE SUCCESSFULLY ERASED
    ACTION:   ERASE AN EMPLOYEE                  LOCATION: SAN_JOSE

    TO SELECT FROM THE LIST PLACE AN S NEXT TO THE EMPLOYEE
    PRESS ENTER TO PROCESS OR END TO EXIT

    E/ID     EMPLOYEE NAME                    D/ID     DEPARTMENT NAME

    SJ0020   S  O'SHEA                        F22      BRANCH OFFICE F22
    SJ0030   D  COOPER                        F22      BRANCH OFFICE F22
    SJ0050   L  ASHER                         F22      BRANCH OFFICE F22
```

Figure 82. Employee at a remote location erased

Employee resume and photo scenarios

The LOB sample application extends the existing DB2 sample employee database by adding a new table for storing employee resumes and photographs as CLOB and BLOB entries. The supporting JCL, application objects, and input data for this table are provided.

The purpose of the LOB sample application is to:

- Provide an IVP for LOB functions
- Demonstrate how to:
  - Use DDL to create LOB objects
  - Use the DB2 LOAD utility to populate LOB columns of 32K bytes or less
  - Create an application program to populate LOB columns of greater than 32K bytes
  - Use LOB locators and related functions to manipulate LOBs without materializing the data.

The LOB sample application consists of a batch portion and an optional online portion. The batch portion verifies that LOB objects can be created, populated, and read successfully using locators and supporting functions. The online portion demonstrates further techniques for using and manipulating LOB data.

The batch portion consists of the following jobs: DSNTEJ7, DSNTEJ71, DSNTEJ73, DSNTEJ75, DSNTEJ76, DSNTEJ77, and DSNTEJ78. For more information about these batch jobs see "Phase 7: Accessing LOB data" on page 407.

The optional online portion of the LOB sample application adds two additional scenarios to the existing three scenarios provided in IVP phase 3. These scenarios include the following activities by the user:

1. Invoking the DB2 sample connection manager to display the DB2 ISPF sample application menu, DSN8SSM
2. Selecting and viewing sample employee resumes by using option 4 of DSN8SSM
3. Selecting and viewing sample employee photo images by using option 5 of DSN8SSM

Application programs for the LOB sample are written in the C and COBOL languages. The resume viewer requires ISPF. The photo viewer requires ISPF and GDDM.

Sample LOB table

The LOB sample application uses the EMP_PHOTO_RESUME table. The sample jobs create, load and manipulate the table.

Table 109. EMP_PHOTO_RESUME table

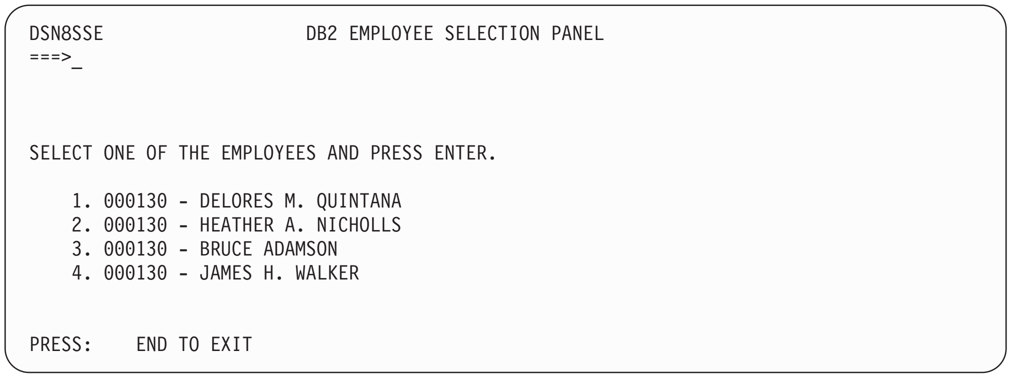
Column Name:	**EMPNO**	**EMP_ROWID**	**PSEG_PHOTO**	**BMP_PHOTO**	**RESUME**
Type:	CHAR(6) NOT NULL	ROWID	BLOB(500K)	BLOB(100K)	CLOB(5K)
Description:	Employee number	Row identifier	Employee photo (PSEG)	Employee photo (BMP)	Employee resume
Values:	000130	#####	Delores M. Quintana	Delores M. Quintana	Delores M. Quintana
Values:	000140	#####	Heather A Nicholls	Heather A Nicholls	Heather A Nicholls
Values:	000150	#####	Bruce Adamson	Bruce Adamson	Bruce Adamson
Values:	000190	#####	James H. Walker	James H. Walker	James H. Walker

LOB application panels for Resume scenario

The LOB application begins with the panel shown in Figure 47 on page 386. Select option 4 from the DB2 Sample Application Menu, DSN8SSM, to look at the resumes under ISPF. DSN8DLRV and DSN8CLRV show the following panels when selecting and viewing sample employee resumes until the user signals ISPF to exit by pressing the END key.

If you ran DSNTEJ73, you can select option 4 on the sample applications panel. If you ran DSNTEJ77, you can select option 6 on the sample applications panel.

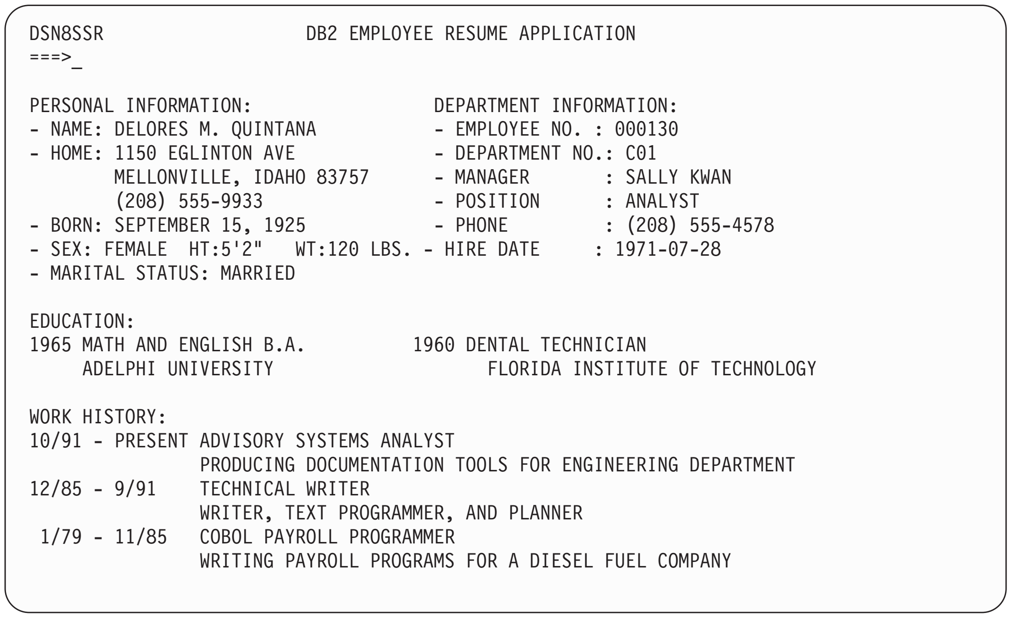
Prompt the user to select from the list of available resumes by displaying ISPF panel DSN8SSE:



```
DSN8SSE                  DB2 EMPLOYEE SELECTION PANEL
===>_



SELECT ONE OF THE EMPLOYEES AND PRESS ENTER.

    1. 000130 - DELORES M. QUINTANA
    2. 000130 - HEATHER A. NICHOLLS
    3. 000130 - BRUCE ADAMSON
    4. 000130 - JAMES H. WALKER


PRESS:    END TO EXIT
```

Figure 83. DB2 Employee Selection Panel

The employee serial numbers and names are hard-coded into the panel because the data for this sample is predetermined. Display the formatted resume on ISPF panel DSN8SSR:
This panel is designed around the sample data. It is assumed that all resume



```
DSN8SSR                  DB2 EMPLOYEE RESUME APPLICATION
===>_

PERSONAL INFORMATION:                 DEPARTMENT INFORMATION:
- NAME: DELORES M. QUINTANA           - EMPLOYEE NO. : 000130
- HOME: 1150 EGLINTON AVE             - DEPARTMENT NO.: C01
        MELLONVILLE, IDAHO 83757      - MANAGER       : SALLY KWAN
        (208) 555-9933                - POSITION      : ANALYST
- BORN: SEPTEMBER 15, 1925            - PHONE         : (208) 555-4578
- SEX: FEMALE  HT:5'2"   WT:120 LBS. - HIRE DATE     : 1971-07-28
- MARITAL STATUS: MARRIED

EDUCATION:
1965 MATH AND ENGLISH B.A.          1960 DENTAL TECHNICIAN
     ADELPHI UNIVERSITY                   FLORIDA INSTITUTE OF TECHNOLOGY

WORK HISTORY:
10/91 - PRESENT ADVISORY SYSTEMS ANALYST
                PRODUCING DOCUMENTATION TOOLS FOR ENGINEERING DEPARTMENT
12/85 - 9/91    TECHNICAL WRITER
                WRITER, TEXT PROGRAMMER, AND PLANNER
 1/79 - 11/85   COBOL PAYROLL PROGRAMMER
                WRITING PAYROLL PROGRAMS FOR A DIESEL FUEL COMPANY
```

Figure 84. DB2 Employee Resume Application

information for an employee will fit predictably and no handling for special cases is provided. The ″Interests″ section of the resume is not presented due to space constraints on the panel.

DSN8DLRV is written in C language and linked with ISPF and DB2 Call Attach Facility. The package name and plan name are both DSN8LR*vr*, where *vr* is the DB2 version and release.

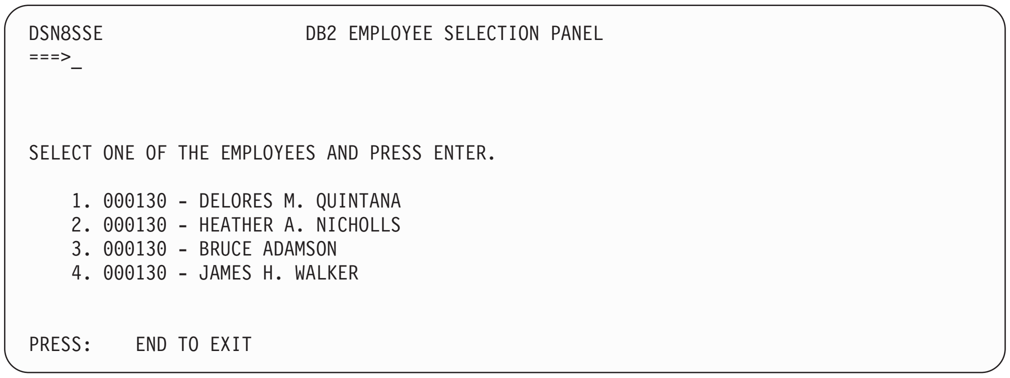
DSN8CLVR is written in the COBOL language and linked with ISPF and DB2 Call Attach Facility. The package name and plan name are both DSN8CR*vr*, where *vr* is the DB2 version and release.

LOB application panels for Photo scenario

The LOB application begins with the panel shown in Figure 47 on page 386. Select option 5 from the DB2 Sample Application Menu, DSN8SSM, to look at employee photos using GDDM. This application requires that you include the GDDM load module library (SADMMOD) in your logon procedure or in the ISPLLIB concatenation.

| If you ran job DSNTEJ75, you can select option 5 on the sample applications panel.
| If you ran job DSNTEJ78, you can select option 8 on the sample applications panel.

DSN8DLRV and DSN8CLRV show the following panels when selecting and viewing sample employee photos. Prompt the user to select from the list of available photos by displaying ISPF panel DSN8SSE:
Select END to exit.



```
DSN8SSE                  DB2 EMPLOYEE SELECTION PANEL
===>_



SELECT ONE OF THE EMPLOYEES AND PRESS ENTER.

    1. 000130 - DELORES M. QUINTANA
    2. 000130 - HEATHER A. NICHOLLS
    3. 000130 - BRUCE ADAMSON
    4. 000130 - JAMES H. WALKER


PRESS:    END TO EXIT
```

Figure 85. DB2 Employee Selection Panel

DSN8DLPV is a C language program linked with ISPF, GDDM and the DB2 Call Attach Facility. The package name and plan name are both DSN8LP*vr*, where *vr* is the DB2 version and release. To run DSN8DLPV you must include the GDDM load module library (SADMMOD) in the logon procedure or in the ISPLLIB concatenation.

| DSN8CLPV is a COBOL language program linked with ISPF, GDDM and the DB2
| Call Attach Facility. The package name and plan name are both DSN8CP*vr*, where
| *vr* is the DB2 version and release. To run DSN8CLPV you must include the GDDM
| load module library (SADMMOD) in the logon procedure or in the ISPLLIB
| concatenation.

Edit exit routine

The edit exit routine is prepared in Phase 1. It works with the employee table (DSN8810.EMP) and is written in assembler language.

The name of the edit exit routine is DSN8EAE1. When the employee table (DSN8810.EMP) is changed by either an update or an add, the edit exit routine encodes the salary amount that goes into the SALARY column. When the SALARY column is read from the employee table, the amount is decoded. The encoding and decoding of the salary column protects the confidentiality of the employee's salary.

Huffman compression exit routine

IBM supplies a sample edit routine that compresses data using the Huffman algorithm (first described in *Proceedings of the IRE* September, 1952). Before using any data compression routine, understand its limitations and consider tailoring it to your particular table. For the restrictions and concerns that apply to the IBM sample, see the comments provided with the code. The routine is called DSN8HUFF and resides in library *prefix*.SDSNSAMP.

Sample field procedure

A sample field procedure is prepared in Phase 1. This procedure causes values in a CHAR(6) column to be ordered in the ASCII sorting sequence.

Dynamic SQL statements (DSNTESA, DSNTESQ)

prefix.SDSNSAMP library members DSNTESA and DSNTESQ contain dynamic SQL statements to help verify the success of an installation or migration.

DSNTESA

The SQL statements in DSNTESA are run dynamically by SPUFI. DSNTESA is used in Phase 3 of the verification process.

The first group of statements in DSNTESA create a temporary work file table space and defines a created temporary table. The INSERT statements fill the table with names, midterm scores, and final examination results, and the SELECT statement then does a check of the averages. The UPDATE statements assign a grade according to the formula in the first UPDATE statement: 60% for the final and 40% for the midterm. The next SELECT statement produces the entire table. The ROLLBACK statement removes the table space and the table within it.

General-use Programming Interface

The following statements make some administrative queries on the system tables:

- The following SELECT statements find all the plans and packages that are owned by the current user, and the date they were bound.

```
SELECT NAME, BINDDATE
    FROM SYSIBM.SYSPLAN
    WHERE CREATOR = USER;
```

```
SELECT COLLID, NAME, VERSION, BINDTIME
    FROM SYSIBM.SYSPACKAGE
    WHERE OWNER = USER;
```

- The following SELECT statements find the plans and packages that require a bind or rebind before they can be run, and the plans and packages that are automatically rebound the next time they are run.

```
SELECT NAME, CREATOR, BINDDATE, VALID, OPERATIVE
    FROM SYSIBM.SYSPLAN
    WHERE OPERATIVE = 'N' OR VALID = 'N';
```

```
SELECT COLLID, NAME, VERSION, BINDTIME, VALID
    FROM SYSIBM.SYSPACKAGE
    WHERE OPERATIVE = 'N' OR VALID = 'N';
```

- The following SELECT statements find all objects required for the current user's programs.

```
SELECT DNAME, BTYPE, BCREATOR, BNAME
    FROM SYSIBM.SYSPLANDEP
    WHERE BCREATOR = USER
    ORDER BY DNAME, BTYPE, BCREATOR, BNAME;
```

```
SELECT DCOLLID, DNAME, BTYPE, BQUALIFIER, BNAME
    FROM SYSIBM.SYSPACKDEP
    WHERE BQUALIFIER = USER
    ORDER BY DCOLLID, DNAME, BTYPE, BQUALIFIER, BNAME;
```

- The second SELECT from SYSTABLES provides information about all the DEPT tables regardless of the owner.

```
SELECT *
   FROM SYSIBM.SYSTABLES
   WHERE NAME = 'DEPT';
```

- The SELECT from SYSCOLUMNS supplies a description of the fields of the DSN8810.DEPT table. This information can also be provided by DCLGEN, and, within a program, the DESCRIBE statement gives this same information.

```
SELECT NAME, COLTYPE, LENGTH, SCALE, NULLS, REMARKS, COLNO
   FROM SYSIBM.SYSCOLUMNS
   WHERE TBNAME= 'DEPT' AND TBCREATOR = 'DSN8610'
   ORDER BY COLNO;
```

- The following SELECT statements find the kinds of authority a user can have. Determining which tables a specific user can access is relatively complicated because of the various authorities. If the user has SYSADM authority, any table can be accessed.

```
SELECT * FROM SYSIBM.SYSPLANAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSPACKAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSUSERAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSDBAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSTABAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSCOLAUTH WHERE GRANTEE = USER;
SELECT * FROM SYSIBM.SYSRESAUTH WHERE GRANTEE = USER;
```

- The final four SELECT statements show the tables and views that can be accessed directly by the current user, those that can be accessed using a plan, and those that are accessed using the database authority.

```
SELECT TCREATOR, TTNAME, STNAME, GRANTOR
   FROM SYSIBM.SYSTABAUTH
   WHERE GRANTEE = USER;
SELECT BNAME, BTYPE, GRANTOR, NAME
   FROM SYSIBM.SYSPLANAUTH, SYSIBM.SYSPLANDEP
   WHERE GRANTEE = USER
      AND NAME = DNAME
      AND EXECUTEAUTH ¬= ' '
      AND (BTYPE = 'T' OR BTYPE = 'V');
SELECT DCOLLID, BNAME, BTYPE, BQUALIFIER, BNAME
   FROM SYSIBM.SYSPACKAUTH, SYSIBM.SYSPACKDEP
      WHERE GRANTEE = USER
         AND COLLID = DCOLLID
         AND NAME = DNAME
         AND EXECUTEAUTH ¬= ' '
         AND (BTYPE = 'T' OR BTYPE = 'V');
SELECT NAME, CREATOR, TYPE, DBNAME, TSNAME
   FROM SYSIBM.SYSTABLES
      WHERE DBNAME IN
        (SELECT NAME FROM SYSIBM.SYSDBAUTH
           WHERE GRANTEE = USER
           AND DBADMAUTH ¬= ' ');
```

End of General-use Programming Interface

DSNTESQ

DSNTESQ contains a set of queries to check consistency between catalog tables. The SQL statements are in a format available for input to SPUFI and DSNTEP2. If SPUFI is not bound when you want to execute these queries, you can use the Version 7 DSNTEP2.

Before running these queries, you should run the DSN1CHKR utility to make sure the physical structure of the catalog is correct. You should also run the CHECK INDEX utility.

DSNTESQ contains SQL that creates copies of the catalog using segmented table spaces. In some cases, the queries in DSNTESQ run faster when run on copies of the catalog instead of the actual catalog because the copies have additional indexes. If you plan to use the copies of the catalog, use the comment lines in DSNTESQ for guidance.

Dynamic SQL programs (DSNTIAD, DSNTEP2, DSNTIAUL)

SPUFI is a part of the distributed product. An installation job is used to bind it. It can be used only with ISPF. DSNTIAD, DSNTEP2, and DSNTIAUL are sample programs and must be compiled, link-edited, and bound as usual. These programs are documented in an appendix of *DB2 Utility Guide and Reference* and *DB2 Application Programming and SQL Guide*

Chapter 11. Connecting the IMS attachment facility

This information explains the requirements for connecting the IMS attachment facility from a DB2 perspective and refers you to IMS topics for specific IMS information. Connecting DB2 to IMS requires coordination with your IMS support group.

To connect the IMS attachment facility, you must:
- Make DB2 load modules available to IMS.
- Define DB2 to IMS.
- Define new programs and transactions to IMS.

Depending on your site, you might also need to:
- Define DB2 plans for IMS applications.
- Generate a user language interface.

The required and optional tasks are described in this topic. An IMS system definition might be required to perform these steps. If RACF is installed, you also need to define the IMS-to-DB2 connection to RACF. For information about how to do this, see Part 3 (Volume 1) of *DB2 Administration Guide*.

In this topic:
- "Making DB2 load modules available to IMS."
- "Defining DB2 to IMS" on page 440.
- "IMS attachment facility macro (DSNMAPN)" on page 445.

Making DB2 load modules available to IMS

If you have already included the *prefix*.SDSNLOAD library in your LNKLST*xx*, you can skip this step. Version 8 modules will be available through normal z/OS module search.

Connecting to more than one release of DB2: If any IMS region connects to more than one release of DB2, then you must ensure that the DB2 load library used for that region is compatible with each release. The IMS attachment facility is upward compatible, but not downward compatible. This means you should use the oldest release of the DB2 load library for the IMS region.

If you have not included the DB2 load libraries in your LNKLST*xx*, you must add STEPLIB statements to your startup procedures and add *prefix*.SDSNLOAD to the DFSESL DD statement.
- If all the data sets referred to in the JOBLIB or STEPLIB statement for an IMS region are APF-authorized, then add the DD statement for *prefix*.SDSNLOAD to the JOBLIB or STEPLIB statement. If the DYNAM option of COBOL is being used, the IMS RESLIB DD statement must precede the reference to *prefix*.SDSNLOAD in the JOBLIB or STEPLIB statement.
- Add the ddname DFSESL DD statement for *prefix*.SDSNLOAD. All libraries specified on the DFSESL DD statement must be APF-authorized. The DFSESL

DD statement is not required by DB2 DL/I batch support. IMS requires that an IMS RESLIB DD statement also be referenced by the DFSESL DD statement, as in the following:

```
//DFSESL     DD      DSN=ims_reslib,DISP=SHR
//           DD      DSN=prefix.SDSNLOAD,DISP=SHR
```

Defining DB2 to IMS

The DB2 identification must be defined to the control region, the DL/I batch region, and, optionally, to each dependent region accessing that DB2 system. To make this identification, you must create a subsystem member (SSM) in the IMS.PROCLIB library, and identify the SSM to the applicable IMS regions.

The DB2 identification for DL/I batch has more parameters than the control and dependent regions. For information about DL/I batch, see Part 4 of *DB2 Application Programming and SQL Guide* .

Placing the subsystem member entry in IMS.PROCLIB: Each SSM entry in IMS.PROCLIB defines at least one connection from an IMS region to at least one different z/OS subsystem.

To name an SSM member, concatenate the value (one to four alphanumeric characters) of the IMSID field of the IMS IMSCTRL macro with any name (one to four alphanumeric characters) defined by your site.

One SSM member can be shared by all of the IMS regions, or a specific member can be defined for each region. This record contains as many entries as there are connections to external subsystems. Each entry is an 80-character blocked or deblocked record. The following examples show how to define fields for IMS. Fields are keyword or positional and are delimited by commas. For more information, see *IMS Customization Guide* from the appropriate release. The fields in this record are:

SST=,SSN=,LIT=,ESMT=,RTT=,REO=,CRC=

where:

SST=DB2
: is a required one-to eight-character name which defines the external subsystem type. It must be set to DB2 for IMS to connect to DB2.

SSN=
: is a required one-to four-character DB2 subsystem name. This name must be the name you specified for SUBSYSTEM NAME on installation panel DSNTIPM. The default is DSN1.

LIT=
: is a required four-character alphanumeric option, specifying the language interface token (LIT) supplied to IMS. The IMS-supplied language interface module (DFSLI000) requires a value of **SYS1** for this option.

  If you need to define connections to different DB2 subsystems, you can follow the procedure described in "Defining DB2 plans for IMS applications (optional)" on page 444.

ESMT=
: is a required one-to eight-character alphanumeric option specifying the external subsystem module table. This module specifies which attachment modules must be loaded by IMS. **DSNMIN10** is the required value for this field.

RTT= is an optional one to eight character alphanumeric name of the user-generated resource translation table (RTT). This table maps the IMS application names into DB2 plan names. If this entry is omitted, the DB2 plan name is the IMS application load module name.

See "Defining DB2 plans for IMS applications (optional)" on page 444 for details on how to generate a resource translation table.

REO= is the optional one-character region error option to be used if an IMS application attempts to reference a non-operational external subsystem or if resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

R passes a SQL return code to the application, indicating that the request for DB2 services failed (default). The most commonly returned SQL codes are -922, -923, and -924. However, there might be other SQL codes returned to the application.

When the first connection to DB2 cannot be established, a SQL return code is not returned. Instead, the application is terminated with an abend code U3047.

Q abends the application with an abend code U3051, backs out activity to the last commit point, does a PSTOP of the transaction, and re-queues the input message. This option only applies when an IMS application attempts to reference a non-operational external subsystem or if the resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

A abends the application with an abend code of U3047 and discards the input message. This option only applies when an IMS application attempts to reference a non-operational external subsystem or if the resources are unavailable at create thread time. If DB2 detects the unavailable resource condition during normal SQL processing, a -904 SQLCODE is returned to the application.

If DB2 is not active or the connection cannot be established when the first SQL call is made from the application program (such as DB2 unavailable, DB2 quiescing, or DB2 terminating), the action you take depends on the region error option specified. SQL codes of -922, -923, or -924 might be returned to the application if option R is specified.

You can change the default for an application if a resource translation table entry is generated for that application. See "Defining DB2 plans for IMS applications (optional)" on page 444.

CRC= is a command recognition character used by IMS to identify DB2 commands entered from an IMS terminal with the /SSR command. Any character is valid for the CRC except the period (.), slash (/), or comma (,). The default CRC is the hyphen (-).

These options apply to DL/I batch only:

CONNECTION_NAME=
The connection name is optional. It represents the name of the job step that is the coordinator for DB2 activity. The connection name defaults are:

Table 110. Default connection names for DL/I batch

Type of Application	Default Connection Name
Batch job	Job name
Started task	Started task name
TSO user	TSO authorization ID

If a batch job fails, you must use a separate job to restart the batch job. The connection name used in the restart job must be the same as the name used in the batch job that failed. Or, if the default connection name is used, the restart job must have the same job name as the batch update job that failed.

DB2 requires unique connection names for DB2 DL/I batch support. If two applications try to connect with the same connection name, then the second application is not allowed to connect to DB2. CONNECTION_NAME can be 1-8 characters long.

PLAN=
You can specify a DB2 plan name. If you do not specify a plan name, the application program module name is checked against the optional resource translation table. If a match is found, the translated name is used as the DB2 plan name. If no match is found, the application program module name is used as the plan name. PLAN can be 1-8 characters long.

PROG=
You must specify the name of the application program to be loaded and to receive control. PROG can be 1-8 characters long.

Providing IMS support for DB2 commands: You can enter DB2 commands through the /SSR command of IMS. The /SSR command format is:

```
/SSR crc DB2 command
```

as in

```
/SSR -DISPLAY THREAD (*)
```

IMS supports this command; you must define the CRC in the SSM member of the IMS control region. If the /SSR command is entered through the z/OS console, the AUTHID WTOR needs to be granted the appropriate authority. If the /SSR command is entered through an IMS terminal, the IMS LTERM name or the signon ID (if active) needs to be granted the appropriate authority.

Specifying the SSM exec parameter: Specify the SSM EXEC parameter in the startup procedure of the IMS control, MPP, BMP, or DL/I batch region. The SSM is concatenated with the IMSID to form a member name in IMS.PROCLIB. The IMSID comes from the IMSID option of the IMSCTRL generation macro or the IMSID option in the control region startup procedure.

For DL/I batch regions, you can specify the DB2 connection parameters in the DDITV02 data set instead of an SSM member. The DDITV02 data set and an SSM member have the same format. See Part 4 of *DB2 Application Programming and SQL Guide* for more information about the DDITV02 data set.

If you specify the SSM for the IMS control region, any dependent region running under the control region can attach to the DB2 subsystem named in the IMS.PROCLIB member specified by the SSM parameter. The IMS.PROCLIB member name is the IMS ID (IMSID=*xxxx*) concatenated with the one to four characters specified in the SSM EXEC parameter. The IMS ID is the IMSID parameter of the IMSCTRL generation macro.

IMS allows you to define as many external subsystem connections as are required. More than one connection can be defined for different DB2 subsystems. All DB2 connections must be within the same z/OS system. For a dependent region, you can specify a dependent region SSM or use the one specified for the control region. You can specify different region error options (REOs) in the dependent region SSM member and the control region SSM member. Table 111 shows the different possibilities of SSM specifications.

Table 111. SSM specifications options

SSM for Control Region	SSM for Dependent Region	Action	Comments
No	No	None	No external subsystem can be connected
No	Yes	None	No external subsystem can be connected.
Yes	No	Use the control region SSM	Applications scheduled in the region can access external subsystems identified in the control region SSM. Exits and control blocks for each attachment are loaded into the control region address space.
Yes	Yes (NULL entry)	No SSM is used for the dependent region	Applications scheduled in this region can access DL/I databases only. Exits and control blocks for each attachment are loaded into the control region address space and each dependent region address space.
Yes	Yes	Check the dependent region SSM with the control region SSM.	Applications scheduled in this region can access only external subsystems identified in both SSMs. Exits and control blocks for each attachment are loaded into the control region and the dependent region address space.

No specific parameter exists to control the maximum number of SSM specification possibilities.

Defining new programs and transactions to IMS

Programs and transactions already defined to IMS can use SQL without any additional definition to IMS.

You can define new programs and transactions that access DB2 resources to your IMS system. Coordinate with your IMS support group to install the programs and transactions for Phase 4 of the verification process. For more information, see Chapter 10, “Verifying installation with the sample applications,” on page 363.

Defining DB2 plans for IMS applications (optional)

The application plan defines the DB2 resources being accessed from an application. The application plan is identified by its plan name. Each IMS application is associated with a plan name.

The default is to have the DB2 plan name the same as the IMS application program load module name. The recommendation is that you use the default.

If you assigned a different name to the plan, you need a resource translation table (RTT). If you chose an error option different from the REO default, you also need an RTT. DB2 provides the DSNMAPN macro in *prefix*.SDSNMACS to generate an RTT. After it is assembled, the table must be link-edited as REENTRANT with RMODE=24 into any authorized library that is concatenated with the library from which IMS loads the DB2 IMS attach modules.

The format of DSNMAPN macro is shown in Table 112.

Table 112. DSNMAPN macro format

Macro	Option	Meaning
DSNMAPN	APN=	IMS application name
	,PLAN=	Associated DB2 plan name
	[,OPTION=]	Specific entry error option R, Q, or A. See REO in the SSM entry.
	[,END=]	Indicates last entry (YES/NO). NO is the default.

This macro is described in more detail in “IMS attachment facility macro (DSNMAPN)” on page 445.

Generating a user language interface (optional)

This step is required only if you intend to access two DB2 subsystems from the same dependent region.

To provide this access, the SSM must contain one entry for each subsystem. Each entry contains a different subsystem ID and its associated language interface token (LIT). IMS provides the DFSLI macro to generate additional language interface modules with unique LITs. The general format of the macro is shown in Table 113.

Table 113. DFSLI macro format and meaning

Macro	Option	Meaning
DFSLI	TYPE	Specifies the type of subsystem that can be accessed through this language interface module. DB2 is the only value supported by this option
	LIT	Defines a name (called LIT) to relate a language interface module with an entry in the SSM for the dependent region

When an IMS application issues a DB2 request, IMS knows the target subsystem by the LIT used in the request. For example, consider the case of a dependent region accessing two DB2 subsystems (DSN1 and DSN2):

- You generate a language interface with LIT=SYS2 (DFSLI001).
- You define two entries in the SSM member. The first entry points to DSN1 with LIT=SYS1; the second points to DSN2 with LIT=SYS2.
- You link-edit applications accessing the DSN1 subsystem with the IMS-provided language interface (DFSLI000).
- You link-edit applications accessing the DSN2 subsystem with the user-generated language interface (DFSLI001).

Even though a region can communicate with two or more DB2 subsystems, an IMS application can access only one—the DB2 subsystem referred to in the language interface that is link-edited. You can alter the SSM to route application requests to a different DB2 subsystem.

IMS attachment facility macro (DSNMAPN)

This macro is required only when an IMS application load module name is different from the name of its related IBM DATABASE 2 application plan, or if the error option is different from the ERR value specified on the IMS SSM entry.

Macro statements are assembled in *prefix*.SDSNMACS and must be link-edited as REENTRANT with RMODE=24 into the DB2 library *prefix*.SDSNLOAD. The module name must be specified on the IMS SSM entry for the DB2 subsystem. The name must be specified as in the RTT entry for the SSM member defining the connection of this region. IMS loads the RTT module into the dependent region address space.

For more information on the RTT, refer to "Defining DB2 plans for IMS applications (optional)" on page 444.

Notes:

1. The macro name must be followed by one or more blanks before options are coded.
2. Multiple options must be separated by commas (with no blanks).

***label* DSNMAPN**
DSNMAPN is the name of the macro. It must be coded exactly as it appears here, and it must be separated from any optional options by one or more blanks.

For *label*, substitute the CSECT name of your module. This name must match the name of the module specified to the linkage editor. *Label* is optional except for the first invocation of the DSNMAPN macro. The last invocation requires END=YES.

APN=*program-name*
Specifies the name of an application load module scheduled by IMS. For *program-name*, substitute an application name of up to eight characters.

PLAN=*plan-name*
Specifies an application plan name that is used (instead of the default application name) when a thread is created. For *plan-name*, substitute an application plan name of up to eight characters.

OPTION=R | Q | A
Specifies the action taken when an application program call cannot be performed because there is some problem in communication between the application program and the DB2 subsystem or if resources are unavailable.

If OPTION is not specified, the region error option (REO) is used.

R Specifies that a return code is returned to the application to indicate that the request for DB2 services failed.

Q Specifies that the transaction is abnormally terminated with an abend code U3051, activity is backed out to the last commit point, and the input message is re-queued.

A Specifies that the transaction is abended with an abend code of U3047, and the input message is deleted.

Default: R

END=NO|YES
Specifies whether this is the last DSNMAPN macro invocation.

NO
Specifies that this is not the last DSNMAPN macro invocation.

YES
Specifies that this is the last DSNMAPN macro invocation.

Default: NO

The last DSNMAPN macro invocation must be followed by the specification END=YES.

Usage notes

- To enter more than one application name (with its corresponding plan name and OPTION specification), you must use multiple invocations of the DSNMAPN macro. The first invocation requires the label; the last invocation requires END=YES.
- Invocations must be in ascending order by application name. If they are not, an MNOTE macro error is generated.

Part 3. Communicating with other systems

Chapter 12. Connecting distributed database systems

You can use the distributed data facility (DDF) of DB2 to access data held by other data management systems, or to make your DB2 data accessible to other systems. DB2 does not place any upper limit on the number of systems it can connect to; available storage is the limiting factor.

The following topics provide additional information:

- "The database protocols (DRDA vs private)"
- "The communications protocols"
- "The role of the communications database (CDB)" on page 450
- "DRDA enhancements" on page 451
- "DB2 installation considerations" on page 452

See *DB2 Data Sharing: Planning and Administration* for information about connecting data base management systems (DBMSs) with a data sharing group.

The database protocols (DRDA vs private)

Applications have two access methods to control remote access:

- The recommended method is to use **DRDA**.

  With Distributed Relational Database Architecture™ (DRDA), the application connects to a server at another location and executes packages that have been previously bound at that server. The application uses a CONNECT statement, a three-part name, or an alias (if bound with DBPROTOCOL (DRDA)) to access the server.

  Queries can originate from any system or application that issues SQL statements as a *requester* in the formats required by DRDA.

  Although use of DRDA is not visible to you, information about it is available on the Open Group web site at www.opengroup.org. For two-phase commit using Systems Network Architecture (SNA) connections, DB2 supports both presumed abort and presumed nothing protocols that are defined by DRDA. If you are using TCP/IP, DB2 uses the sync point manager defined in the documentation for DRDA Level 3. Again, this is not visible to you, but information about presume nothing protocols is contained in *SNA LU 6.2 Peer Protocols Reference*.
- The other method, **which is not recommended**, is to use **DB2 private protocol**.

  With private protocol, the application must use an alias or three-part name to direct the SQL statement to a given location. Private protocol only works between requesters and servers that are both DB2 subsystems. Private protocol does not support many distributed functions, such as TCP/IP or stored procedures. The newer data types, such as LOB or user-defined types, are also not supported by private protocol. You cannot use new Version 8 functionality if you use private protocol.

The communications protocols

DDF uses TCP/IP or SNA to communicate with other systems. Figure 86 on page 450 shows the connectivity options you have with DB2's DDF.

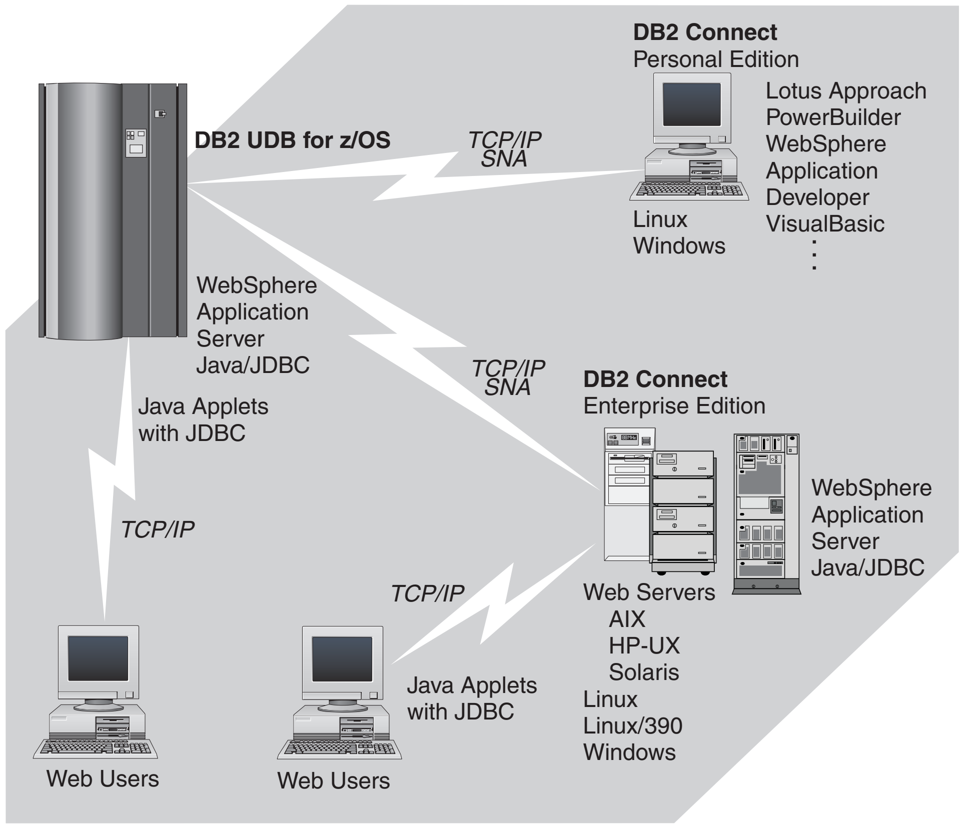


Figure 86. Connectivity options

Setting up a network for use by database management systems requires knowledge of both database management and communications. Thus, you must put together a team of people with those skills to plan and implement the network.

The role of the communications database (CDB)

When sending a request, DB2 uses the LINKNAME column of the SYSIBM.LOCATIONS catalog table to determine which protocol to use, as shown in Figure 87 on page 451. To receive VTAM requests, you must select an LUNAME in installation panel DSNTIPR. To receive TCP/IP requests, you must select a DRDA port and a resynchronization port in installation panel DSNTIP5. TCP/IP uses the server's port number to pass network requests to the correct DB2 subsystem.

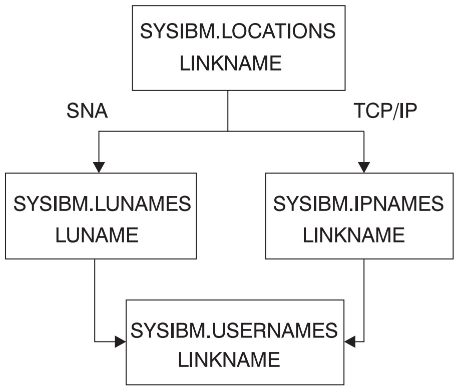


Figure 87. The linkname column of SYSIBM.LOCATIONS determines protocol

If the value in the LINKNAME column is found in the SYSIBM.IPNAMES table, TCP/IP is used for DRDA connections. If the value is found in SYSIBM.LUNAMES table, SNA is used. If the same name is in both SYSIBM.LUNAMES and SYSIBM.IPNAMES, TCP/IP is used to connect to the location.

Attention: A requester cannot connect to a given location using both SNA and TCP/IP protocols. For example, if your SYSIBM.LOCATIONS specifies a LINKNAME of LU1, and if LU1 is defined in both the SYSIBM.IPNAMES and SYSIBM.LUNAMES table, TCP/IP is the only protocol used to connect to LU1 from this requester for DRDA connections. For private protocol connections, the SNA protocols are used. If you are using private protocol connections, the SYSIBM.LUNAMES table must be defined for the remote location's LUNAME

DRDA enhancements

Some of the DRDA enhancements in Version 8 include:

- **Long 255 IDs:** Allows the flowing of long SQL identifier, package and procedure names.
- **Enhanced describe:** Allows requesters to request additional SQLDA descriptive information about an SQL statement than the information returned in a standard SQLDA. JDBC and CLI clients exploit this enhancement for improved JDBC and CLI functionality.
- **Extended diagnostics:** Adds and extends the existing SQLCA structure that flows between requesters and servers. The new SQLCA structure provides additional diagnostics for multi-row SQL operations, server-provided warning and error message texts, and additional serviceability information to help diagnose distributed data problems.
- **Multi-row fetch and insert:** Supports the new multi-row SQL statements introduced in Version 8.
- **SQL cancel:** Remote interrupt support allows a CLI or JDBC application the ability to cancel a long-running request running on a DB2 Server.
- **Scrollable cursors:** Support for scrollable cursors has been enhanced in Version 8. See "Migration considerations" on page 304 for more information about scrollable cursors.
- **Cursor attributes:** Allows additional cursor attributes to be provided at prepare time.
- **Monitoring:** Provides the ability for a server to return how long it takes to process a request by DB2. This function is used by the DB2 Connect system monitoring feature.

- **SELECT with INSERT:** Allows a query to update resources on a server.
- **Data encryption:** Provides the ability to encrypt security sensitive user data when flowing across a TCP/IP network.
- **Package path:** Supports the new CURRENT PACKAGE PATH special register that can be used to specify a list of qualifiers for packages which is used by the server during package resolution. This results in reduced network traffic and an improvement in CPU elapsed time for applications because it requires crossing the network only once to resolve the package name at the server, instead of crossing once per collection to perform resolution at the requester.
- **Cursor instances:** Allows a requester to have more than one instance of an open cursor or a stored procedure result set concurrently. A stored procedure can be invoked again without losing the results of the previous invocation. JDBC and CLI clients use this enhancement for improved JDBC and CLI functionality.
- **DRDA XA protocol support:** Supports distributed transactions that implement Java 2 Enterprise Edition (J2EE), Java Transaction Service (JTS), and Java Transaction API (JTA) specifications. This support is only available for TCP/IP connections.

DB2 installation considerations

The installation options for DDF are described in "Distributed data facility panel 1: DSNTIPR" on page 219 and "Distributed data facility panel 2: DSNTIP5" on page 225. Use these option to define, among other things:

- Whether you want DDF to start automatically when DB2 starts
- Important names for this DB2 subsystem, including a LU name and NETID

  Even if you do not plan to use VTAM, you still must define an LU name and NETID, because DB2 as a requester using TCP/IP protocols generates the unit of work using NETID and LUNAME.
- Thread management options
- Security options
- Default database protocol (DRDA or private)
- TCP/IP port numbers
- Control of the number of DRDA query blocks that can flow on a network request that was specified with OPTIMIZED FOR n ROWS where *n* exceeds the number of rows that fit in a single query block.

Support for extended dynamic SQL: If this DB2 subsystem services requesters that support extended dynamic SQL, such as SQL/DS™, enter YES in field DESCRIBE FOR STATIC on installation panel DSNTIPF. This option lets applications from the requesting system execute SQL DESCRIBE statements that appear as extended dynamic SQL statements in the requesting system, but appear as static SQL in the DB2 package. For the option to take effect, you must bind the package with DESCRIBE FOR STATIC enabled.

Test your connections You should test systems with each other to ensure that their communications setups are correct. If you are testing with another DB2 UDB for z/OS, enter the location name of that other site in field REMOTE LOCATION of installation panel DSNTIPY. The remote location must also have DDF installed and active and must have run the first sample job, DSNTEJ1.

Chapter 13. Connecting systems with VTAM

This chapter tells you how to set up DB2 and VTAM for remote communication. For information about enabling communication with non-DB2 database management systems, see *Distributed Relational Database Architecture: Connectivity Guide* and the appropriate product publications.

Terminology: The following communications terms are used in this chapter:

Logical unit (LU)	A source of requests entering the network and a receptor of replies from the network. For example, a particular DB2 is an LU.
Session	A logical connection between two LUs. Multiple sessions can run on a single physical connection.
Conversation	A dialog that uses a session to transfer information between transaction programs, such as DB2 to DB2. A single session can support multiple conversations, but only one at a time.

To prepare DB2 for communication using the distributed data facility (DDF), we suggest the following steps. You can do steps 1, 2, and 3 after installing DB2. Steps 6 through 8 are optional.

"Step 1: Customize VTAM for DB2" on page 455

To make monitoring of the network easier, consider installing NetView. For information about NetView, see *Tivoli NetView for z/OS Installation: Getting Started*. For information about planning your network, see *Planning for NetView, NCP, and VTAM*. For information about installing VTAM, see *VTAM for MVS/ESA Network Implementation Guide*.

"Step 2: Choose names and a password" on page 455

You need to choose two names for the local DB2 subsystem: a location name and a logical unit name (LU name). A *location name* distinguishes a specific database management system in a network, so applications use this name to direct requests to your local DB2 subsystem. Other systems use different terms for a location name. For example, DB2 Connect calls this the *target database name*. We use the DRDA term, *RDBNAM*, to refer to non-DB2 systems' relational database names.

An *LU name* is the name by which VTAM recognizes this subsystem in the network. You might need to know the LU names of other systems that can request data from the local DB2 subsystem, or you can use a default LU name of eight blanks.

If you plan to request data from other systems, you need the LU names *and* location names for those serving systems. Most of the time, system administrators and operators need to know both names, because they can use both names in various commands, and DB2 uses both names in messages.

In addition to the names mentioned above, you can choose an optional password to validate your local DB2 subsystem to VTAM. If the z/OS system on which DB2 is running is part of an z/OS Parallel Sysplex, you can choose a generic LU name to define a DB2 group to remote locations. For information about using generic resources, see *VTAM for MVS/ESA Network Implementation Guide*.

"Step 3: Define the DB2 subsystem to VTAM" on page 457

In this topic, we tell how to use the VTAM APPL statement to make the DB2 subsystem known to VTAM. You must include the APPL definitions in the VTAM SYS1.VTAMLST library at VTAM startup.

We also tell how to use the VTAM MODEENT statement to define default session modes. DB2 uses one default mode for DRDA access conversations and another for DB2 private protocol access conversations. You must include mode tables in the VTAM SYS1.VTAMLIB library at VTAM startup.

Sample VTAM definitions are provided in "Sample VTAM definitions to connect two DB2s" on page 482, in the data set DSN8VTAM in SDSNSAMP, and in examples throughout this chapter.

"Step 4: Populate the communications database" on page 464

The DB2 catalog includes the communications database (CDB), which contains several tables that hold information about your connections with remote systems. You must populate some of these tables before you can request data from those remote systems. If this DB2 system only services data requests, you do not have to populate the CDB; you can use the default values.

"Step 5: Start VTAM to use DB2" on page 468

When you start VTAM to use DB2, you must be sure that the proper definitions are in the VTAM libraries VTAMLST and VTAMLIB.

"Step 6: Tune the system" on page 468

This is an optional step, which you can do after you have established communications between two or more systems. The procedure outlined up to this point gives you default values for your DB2 modes and your class of service. Although the defaults are probably adequate for your preliminary testing, you can change them to improve performance in the network, or to assign different modes to different application plans. VTAM publications, such as *VTAM for MVS/ESA Network Implementation Guide* , contain more detailed information about tuning the network.

In this topic, we discuss session and mode options you can modify. When VTAM links two nodes, it establishes a session. The number of sessions available can have a significant impact on performance; therefore, you might need to modify your session limit values. "Calculating session limits" on page 476 contains more detailed information about calculating session limits. Also, large amounts of DB2 data travelling through the network can severely affect VTAM storage, and you might need to tune buffer storage. "Calculating VTAM I/O buffer pool (IOBUF) storage" on page 479 contains more detailed information about calculating VTAM buffer pool storage.

You can also tune the system by changing mode options. A *mode* describes various characteristics of a session, such as the maximum number of bytes sent at one time. Modes can point to a *class of service* table, which ranks the available virtual routes for this mode with respect to preference of use and paths through the network. Essentially, the class of service table allows you to assign different network priorities to your modes.

″**Step 7: Create Aliases**″

This is an optional step. Each DB2 location can create aliases for the tables it wants to access, using DB2 private protocol access or DRDA, at the other DB2 locations. For more information about creating aliases, see Part 2 (Volume 1) of *DB2 Administration Guide*.

″**Step 8: Provide Authorization for an Appropriate Level of Security**″

See Part 3 (Volume 1) of *DB2 Administration Guide* for information about security considerations for distributed data processing.

The following topics provide additional information:

- “Step 1: Customize VTAM for DB2”
- “Step 2: Choose names and a password”
- “Step 3: Define the DB2 subsystem to VTAM” on page 457
- “Step 4: Populate the communications database” on page 464
- “Step 5: Start VTAM to use DB2” on page 468
- “Step 6: Tune the system” on page 468
- “Sample VTAM definitions to connect two DB2s” on page 482
- “Using the change log inventory utility to update the BSDS” on page 490

Step 1: Customize VTAM for DB2

For DB2 to provide the best performance for distributed, you probably need to customize VTAM. Before you customize VTAM, consider the communication needs of your DB2 connections. Because you could allow your DB2 subsystem to send large amounts of data through the network, reexamine the capacity of your existing network. In some cases, portions of your existing network might need additional communication hardware to provide the required capacity. VTAM publications, including *VTAM for MVS/ESA Network Implementation Guide* and others, contain more information about these considerations.

Step 2: Choose names and a password

In this step, you choose names for your local DB2 subsystem, and, possibly, a VTAM password for it. We also describe the conditions under which you need to know the names of remote systems in the network.

Names for the local subsystem

You define the names for the local subsystem and its VTAM password to DB2 by using the installation panels, or by using the change log inventory utility as described in “Using the change log inventory utility to update the BSDS” on page 490. Choose the following names for the local DB2 subsystem:

- A unique name by which the other systems in the network can recognize your subsystem. The name can have from 1 to 16 characters and is called the *location name*. (DB2 Connect refers to this as the *target database name*.) Make sure that the local location name is different from the name of every other system in the network, no matter where it is physically located.

  You must share the location name with the other systems that need to send SQL requests to this one.

  The location name should not change even if the network changes. Therefore, tightly control the allocation of location names. To ensure uniqueness, we recommend that you use an IBM-registered SNA NETID as the first six bytes of your location name. If location names are not unique, you have to change many programs and tables if your network is later joined with another network using the same location name.

  The IBM recommendation for the NETID is the following format:

  – The first two bytes are the country code as defined in ISO standard ISO 3166. These codes include the uppercase letters A through Z.
  – The next four bytes are the enterprise code of the registering enterprise. This might already be registered with IBM as your SNA NETID. The enterprise code can include the uppercase letters A through Z, the numbers 0 through 9, and the underscore character (_).

To register your SNA NETID, see your IBM representative.

- A name by which VTAM can recognize the local subsystem. It must be either a unique name or, in some cases, a generic name.
  - The *unique* name must be unique within the network of connected systems, can have from 1 to 8 characters, and is called the *LU name*. The LU name and the location name of a subsystem can be identical, but we do not recommend this; LU names are unique only within a network, and networks can change. You must share the LU name with any system that requests data from your local subsystem. Later, you enter this name in the VTAM APPL statement described in "Step 3: Define the DB2 subsystem to VTAM" on page 457.
  - If the z/OS system on which DB2 is running is part of an z/OS sysplex, you can use a *generic* 8-character name to represent a group of VTAM LU names. The generic name might be useful if your network is in a transitional period, and you want to use generic names to reference network nodes.

    Specify the generic LU name in the field DB2 GENERIC LUNAME on installation panel DSNTIPR. Use column GENERIC of SYSIBM.LUNAMES to indicate that you want to use the generic LU name for CNOS processing and SQL requests to a particular server.

    See *DB2 Data Sharing: Planning and Administration* for instructions on setting up a generic LU name for a data sharing group.
- Server location aliases. You can use the Change Log Inventory utility to define up to eight aliases for a location. Applications can use these alias names to refer to the local DB2 subsystem or data sharing group.

  Aliases are most useful in a data sharing environment in which two or more DB2 subsystems are migrated to a single data sharing group. In this case, you can define the old location for each subsystem as an alias for the location name of the group. Remote applications that refer to the old location names do not need to change.

  Use the Print Log Map utility to print all location alias names that are defined for a DB2 subsystem.

  You can set up multiple sever location aliases as locations on the z/OS requester to restrict which members an application will use when accessing a data sharing group. The z/OS requester must be using the IPLIST or LULIST table to route connections to the data sharing group.

A password for the local subsystem

Choosing a VTAM password is optional but recommended. It can have from one to eight EBCDIC characters. If you decide to use a password, you must enter it on the PRTCT option of the VTAM APPL statement. This password is not transmitted through the network, so there is no need to share the password with the other systems. For more information about this password, see Part 3 (Volume 1) of *DB2 Administration Guide*.

DB2 does not require you to use a password as long as you have not included one in the VTAM APPL statement.

Names you need from the remote systems

Location names and LU names: When you populate the communications database (CDB) in the local DB2, you must know the location names (or DRDA RDBNAMs) and LU names of remote servers (that is, systems from which this DB2 will request data). The local DB2 does not need location names of requesters; however, you need to know the requesters' LU names if you intend to change default communication options.

DB2 does not receive DRDA RDBNAM from requesters other than DB2 UDB for z/OS. If DB2 does not have an RDBNAM, it displays LU names in messages, display output, and trace output. To help you distinguish between location names and LU names in those cases, the LU name is enclosed in less-than (<) and greater-than (>) brackets.

When your systems begin communicating, you and others involved in working with distributed systems need to be aware of the LU name to DRDA RDBNAM mappings. When you have obtained the necessary names, enter them in the CDB as described in "Step 4: Populate the communications database" on page 464.

Transaction program names (TPNs): If a server is not a DB2 UDB for z/OS, it might have an additional name that uniquely identifies it. In LU 6.2, this is known as a *transaction program name* (TPN), and can be from 1 to 64 characters long. When a DB2 UDB for z/OS subsystem communicates with other DB2 UDB for z/OS subsystems, you do not need to supply TPN values. The DB2 subsystems automatically choose the correct TPN values for both DRDA access and DB2 private protocol access.

When a TPN is necessary: You might need to supply TPN values when a DB2 subsystem requests data from a server that is not a DB2 UDB for z/OS subsystem. For cases where the server does not accept the default TPN for DRDA access, enter into your CDB the TPN chosen by that server. For DB2 for VM, for example, the TPN is the SQL database machine ID.

TPN values accepted by DB2 UDB for z/OS: A requester that is not DB2 UDB for z/OS must use either the TPN name X'07F6C4C2' or DB2DRDA, which are the only values DB2 recognizes when it accepts a request from another system. Some requesters enter the TPN as two separate fields: a 1-byte prefix (X'07') and a 3-byte suffix ('6DB').

Names chosen by Spiffy Computer Company

Spiffy has chosen the location names and LU names shown in Table 114, some of which we use in later examples.

Table 114. Spiffy's location names, lu names, and transaction program names (TPNS)

Location Name	LU Name	TPN	Comments
USIBMSTODB21	LUDB21		DB2 *
USIBMSTODB22	LUDB22		DB2
USIBMSTOSQL1	LUSQLDS	TPNSQLDS1	DB2 for VM production system
USIBMSTOSQL2	LUSQLDS	TPNSQLDS2	DB2 for VM test system

Note: USIBMSTODB21 plans to accept requests from many OS/2® and Windows NT® requesters.

Step 3: Define the DB2 subsystem to VTAM

You need to use the following VTAM objects in this step:

- An APPL definition statement, described in "The APPL statement" on page 458
- A MODEENT macro, described in "The MODEENT macro" on page 462.

Samples of both the APPL and MODEENT macros are in the DSN8VTAM sample data set.

The APPL statement

A VTAM APPL definition statement defines the VTAM options for the DB2 subsystem and includes it in a major node. With VTAM, you can use a model application program definition for DB2. With a model definition, you use wild card characters for the application name (LU name).

Spiffy uses the statement in Figure 88 for the USIBMSTODB21 DB2 subsystem:

```
LUDB21 APPL    APPC=YES,                                               X
               ATNLOSS=ALL,                                            X
               AUTH=(ACQ),                                             X
               AUTOSES=1,                                              X
               DMINWNL=25,                                             X
               DMINWNR=25,                                             X
               DSESLIM=50,                                             X
               MODETAB=DB2MODES,                                       X
               PRTCT=D02DN,                                            X
               SECACPT=ALREADYV,                                       X
               SRBEXIT=YES,                                            X
               SYNCLVL=SYNCPT,                                         X
               VERIFY=NONE,                                            X
               VPACING=2
```

Figure 88. Example of a VTAM APPL definition statement

For your convenience, the APPL statement example is provided in data set DSN8VTAM, in the sample library, SDSNSAMP.

The topics that follow describe the APPL options that Spiffy uses and a few more in which you might be interested. There are others you can use; for information about those, see *VTAM for MVS/ESA Resource Definition Reference.*

Options for which you must choose values

For some options, you must supply a specific value; for others, DB2 suggests values that are not the VTAM defaults. In your APPL statement, you must code values for the following:

name
The 1 to 8 character LU name you chose in "Step 2: Choose names and a password" on page 455. For their USIBMSTODB21 DB2 system, Spiffy uses LUDB21.

AUTOSES
The number of contention winner sessions that VTAM is to activate automatically between this DB2 and another system on a given mode before DB2 requests a conversation to be created.

Contention occurs when two LUs want to allocate a conversation at the same time in the same session. In order to resolve contention situations, VTAM denotes one LU as the contention winner and one as the contention loser. The winner automatically prevails and is allowed to allocate its conversation. The loser must wait to allocate its conversation.

The default is 0. The suggested value is 1 or greater to ensure that VTAM informs DB2 if a session is inactivated.

Too large a number can take up storage and create resources that are not used. A small number can result in a one-time delay to bring up additional sessions when they are needed by an application.

DMINWNL The minimum number of parallel sessions in which, if there is contention for a conversation, this local DB2 subsystem is the winner.

The suggested value is one-half the value of DSESLIM, described below.

DMINWNR For the same situation as described for DMINWNL, the number of sessions in which the remote system is the winner. The suggested value is one-half the value of DSESLIM, described below.

For more information about session negotiations, see "Interpreting CNOS messages" on page 480.

DSESLIM The default maximum number of sessions allowed for this DB2 subsystem as it communicates with any other system on a given mode. For performance reasons, the DB2 suggested value for DSESLIM is the maximum number of sessions that can possibly be in use on any mode. For example, assume you have 5 modes for which the following maximum numbers of sessions could be active: 10, 12, 20, 30, 40. In this case, DSESLIM should be 40.

Because calculating a precise value for this number can be rather difficult if you do not know exactly how many applications run on a specific mode, Spiffy chooses 50. They can modify this option later if they have problems obtaining enough sessions, or if they find they are requesting sessions that they never need. For information about how to calculate this number, see "Calculating session limits" on page 476.

You can use DSESLIM to control the number of sessions that this subsystem can issue or receive. For example, to avoid overloading this subsystem with requests from remote application processes, you can assign a low number to DSESLIM to limit the number of simultaneous remote requests issued by a given partner and mode.

Use the CONVLIMIT column of the LUMODES table in the CDB to override this value for specific cases. See "Update SYSIBM.LUMODES with conversation limits" on page 474 for information about how to do that.

MODETAB The name of the VTAM logon mode table you use to define DB2 session modes. Only modes defined in this table are eligible for conversations created by the local DB2. If you leave this blank, DB2 uses the default mode table shipped with VTAM (ISTINCLM). Spiffy decides to set up a separate mode table and chooses the name DB2MODES. DB2 cannot use either the default mode table or the one you set up yourself until you make entries into the table as described in "The MODEENT macro" on page 462.

PRTCT If you decided to use a password, as described in "Step 2: Choose names and a password" on page 455, this is that password. Later, you must store the same password in the bootstrap data set (BSDS), entering it through installation panels or the change log inventory utility.

If you prefer not to use a password, omit this option. The installation panels and the change log inventory utility do not require you to enter a password. For more information about using the VTAM password, see Part 3 (Volume 1) of *DB2 Administration Guide*.

SECACPT The level of conversation-level security allowed.

Recommendation: Use ALREADYV, which gives you the most flexibility in determining your security. You can use the CDB to determine levels of security on a more granular basis as described in Part 3 (Volume 1) of *DB2 Administration Guide*

We do not recommend SECACPT=CONV because in many cases, it does not allow already verified conversations for DRDA access. It works for conversations that use only DB2 private protocol access.

VERIFY Whether you want SNA partner LU verification. The default, VERIFY=NONE, means that any system can connect with yours. Because Spiffy is setting up a small, restricted network, it chooses the default for now.

Use VERIFY=REQUIRED to activate partner LU verification. This means that you let RACF and VTAM check the identity of an LU that is attempting to connect with yours. For more information # about partner LU verification, see Part 3 (Volume 1) of *DB2 Administration Guide* and *VTAM for MVS/ESA Network Implementation Guide*.

DB2 has no dependency on the value you choose.

VPACING The maximum number of messages that another system can send to this local DB2 subsystem during a conversation before waiting to receive a pacing response. The suggested value is 2.

These message sizes are determined by the RUSIZES option of the MODEENT macro. VPACING and RUSIZES, together with some overhead, determine the amount of storage required for the pacing window. For more information about pacing, see “Controlling pacing” on page 470.

Options you must code exactly as given

In some cases, DB2 requires particular values of APPL options. For the following, you must code the options exactly as shown; they are *not* the VTAM defaults:

APPC=YES Tells VTAM that DB2 uses APPC conversation verbs.

AUTH=(ACQ)
Determines the DB2 system authority to use certain VTAM functions.

SRBEXIT=YES
Tells VTAM that DB2 uses service request block (SRB) processing in its exit routines.

SYNCLVL=SYNCPT
Tells VTAM that DB2 supports two-phase commit. Other systems communicating with this DB2 use this indication to determine if DB2 supports the updating of many locations in one unit of work.

Coding SYNCLVL=SYNCPT does not preclude the support of partner LUs that do not support two-phase commit. DB2 still supports the non-two-phase process. See Part 4 of *DB2 Application Programming and SQL Guide* for information about writing applications that access partners that do not have two-phase commit processing.

Options that must use VTAM defaults

For the following options, DB2 *must* use the VTAM defaults; you do not need to code the options:

HAVAIL=NO	Indicates whether an XRF session can be supported. DB2 requires the default, NO.
PARSESS=YES	Specifies that parallel sessions are allowed. This defaults to YES when APPC=YES.
ENCR=NONE	Specifies information about specific cryptographic requirements. There is no support for encryption in this release of VTAM for LU 6.2 applications; therefore, this must be NONE.
SONSCIP=NO	Specifies information about SCIP exit routines. DB2 does not have SCIP exit routines; this must be NO.
VTAMFRR=NO	Specifies whether a VTAM functional recovery routine is in effect when control is returned to DB2. DB2 uses its own recovery routines; this must be NO.

Other options of interest

In most cases, you can reasonably use the VTAM defaults at first, as Spiffy does. You can change them later. We list them here in case you have some reason not to use the default values.

ACBNAME	The LU name for the DB2 subsystem. If the ACBNAME is different from the APPL name and both the originating and destination LUs are in the same VTAM domain, do not refer to the ACBNAME in a CDB definition. If the ACBNAME is not the same as the APPL name, VTAM may encounter name conflicts.
DDRAINL	Whether the local DB2 subsystem wants to accept permission to drain its allocation requests if a change-number-of-sessions (CNOS) request is received that specifies that draining is allowed. The suggested value is the default, NALLOW (do not allow draining).
DRESPL	Whether the local DB2 is responsible for deactivating sessions when it receives a CNOS request specifying the local DB2 as the responsible system. The suggested value is the default, NALLOW (do not be responsible).
EAS	The approximate number of concurrent sessions for this DB2 subsystem. For performance reasons, it is better to estimate slightly high. The VTAM default is 509.
LMDENT	The number of entries to be used for a hash table of other systems. The suggested value is the approximate number of other systems in the network. Spiffy decides to use the default value of 19.
MAXPVT	The maximum additional amount of private area storage that can be used by VTAM within the DDF address space for the session-related control blocks and messages for DB2. Specifying 0 indicates an unbounded amount; this is the VTAM default.
OPERCNOS	The ability to have a VTAM operator display and set VTAM session limits for a given LUNAME and MODENAME.

- Use ALLOW to enable a VTAM operator to change session limits dynamically without stopping DDF or changing the CONVLIMIT column of the SYSIBM.LUMODES table.
- Use NALLOW, the default, to make sure VTAM operators are not able to change DB2's session limits dynamically.

Options ignored by DB2

The following options are not applicable to DB2 as a VTAM application; do not code them in your APPL statement:

ASLENT	ATNLOSS	MDLTAB	SSCPFM
ASLTAB	MDLENT	POAQNAM	USSTAB

The MODEENT macro

A VTAM link between two systems is a *session*. For every session, there must be a defined set of characteristics called a *mode* existing in a VTAM table called a *log mode table*. This is the table you named in the MODETAB option of the APPL statement, described on "The APPL statement" on page 458.

You can create your own log mode table, or add mode names to the default mode table, called ISTINCLM, that is shipped with VTAM. If you decide to add your modes to the default mode table (ISTINCLM), you can find that table in SYS1.SAMPLIB.

Spiffy decides to use the DB2 default modes at first, but also to go ahead and set up a separate mode table for modes used by DB2 for distributed data processing. They can then populate this table with additional modes as they are needed.

Default modes

There are the following default modes:

- SNASVCMG is an optional mode. It is reserved for use by VTAM for CNOS processing (described in "Interpreting CNOS messages" on page 480) and exists in the VTAM default log mode table. Because SNASVCMG is reserved for use by VTAM, do not enter it as a mode name in the CDB. If you have decided to set up a separate mode table for DB2, you can, if you choose, copy the SNASVCMG mode entry into your DB2 mode table, or just use it as it exists in the ISTINCLM mode table. See *VTAM for MVS/ESA Resource Definition Reference* for a description of this mode.
- IBMRDB is a recommended mode entry because it is used as a default for DRDA access whenever you do not explicitly assign a mode to a session. It does *not* exist in the default table; to use it as a default you must add it to your mode table.
- IBMDB2LM is a recommended mode entry because it is used as a default for DB2 private protocol access whenever you do not explicitly assign a mode to a session. It does *not* exist in the default table; to use it as a default you must add it to your mode table.

Use the MODEENT macro to enter each mode into your mode table. When this table is complete, you must assemble and link-edit it into SYS1.VTAMLIB. See *VTAM for MVS/ESA Resource Definition Reference* for more information about creating mode tables.

Sample mode entries

The sample mode entries for IBMDB2LM and IBMRDB contain the following options that are necessary for dependent LUs to request VTAM sessions.

COMPROT	PRIPROT	TSPROF	SECPROT
FMFPROF	PSERVIC	TYPE	

The samples in Figure 89 work for both dependent and independent LUs; however, if you have no dependent LUs, it is not necessary to re-assemble your existing mode table with the above options. For your convenience, a sample MODEENT is included in data set DSN8VTAM, in SDSNSAMP. See *Distributed Relational Database Architecture: Connectivity Guide* for more information about dependent LUs.

The ENCR option is ignored by LU 6.2 and is thus not included in our samples.

```
DB2MODES MODETAB
IBMDB2LM MODEENT LOGMODE=IBMDB2LM,   DB2 DEFAULT MODE FOR SYS-DIR ACC  X
               TYPE=0,             NEGOTIABLE BIND                     X
               SSNDPAC=X'02',     SECONDARY SEND PACING COUNT          X
               SRCVPAC=X'00',     SECONDARY RECEIVE PACING COUNT       X
               RUSIZES=X'8989',   RUSIZES IN-4096 OUT-4096             X
               FMPROF=X'13',       LU6.2 FM PROFILE                    X
               TSPROF=X'07',       LU6.2 TS PROFILE                    X
               PRIPROT=X'B0',      LU6.2 PRIMARY PROTOCOLS             X
               SECPROT=X'B0',      LU6.2 SECONDARY PROTOCOLS           X
               COMPROT=X'50A5',    LU6.2 COMMON PROTOCOLS              X
               PSERVIC=X'060200000000000000122F00'    LU6.2 LU TYPE
IBMRDB   MODEENT LOGMODE=IBMRDB,     DB2 DEFAULT MODE FOR APP-DIR ACC  X
               TYPE=0,             NEGOTIABLE BIND                     X
               SSNDPAC=X'02',      SECONDARY SEND PACING COUNT         X
               SRCVPAC=X'00',      SECONDARY RECEIVE PACING COUNT      X
               RUSIZES=X'8989',    RUSIZES IN-4096 OUT-4096            X
               FMPROF=X'13',       LU6.2 FM PROFILE                    X
               TSPROF=X'07',       LU6.2 TS PROFILE                    X
               PRIPROT=X'B0',      LU6.2 PRIMARY PROTOCOLS             X
               SECPROT=X'B0',      LU6.2 SECONDARY PROTOCOLS           X
               COMPROT=X'50A5',    LU6.2 COMMON PROTOCOLS              X
               PSERVIC=X'060200000000000000122F00'    LU6.2 LU TYPE
         MODEEND
         END
```

Figure 89. Sample mode entries

Modeent options

When considering values for modes, realize that the partner system can choose different values. If the partner has different values, VTAM negotiates the values to limits acceptable to both systems when the session is established for the mode.

The options used in the MODEENT macro have the following meanings.

name
: The *name* option (IBMDB2LM and IBMRDB in the examples) is optional and has no function in the specification of a logon mode table.

LOGMODE
: Specifies the logon mode name to be used as a key for the session options in this table entry. This logon mode name corresponds to mode name columns in the CDB.

TYPE
: TYPE=0 indicates that DB2 is using a negotiable BIND, which is required for communicating with dependent LUs.

SRCVPAC	Specifies the secondary receive pacing count. The DB2 suggested value is X'00'.
SSNDPAC	Specifies the secondary send pacing count. The DB2 suggested value is any nonzero number. Do not use 0; this turns off pacing, which can result in problems with IOBUF storage.
RUSIZES	Specifies the maximum length of data in bytes that can be sent and received in one *request/response unit* (RU). It is read as two numbers, each having two hexadecimal digits: the first number for the send amount, the second for the receive amount. The suggested value of X'8989' means that VTAM sends a maximum of 4096 bytes (8×2^9) across *at one time*, but there is no limit on how much total information can be sent.
FMPROF	This constant specifies the function management profile required for LU 6.2.
TSPROF	This constant specifies the transmission services profile required for LU 6.2.
PRIPROT	This constant specifies the primary LU protocols used in LU 6.2.
SECPROT	This constant specifies the secondary LU protocols used in LU 6.2.
COMPROT	This constant specifies the common LU protocols used in LU 6.2.
PSERVIC	This constant specifies this as an LU type 6.2.

Some of the above options can have a profound effect on performance because of their impact on pacing. For more information about these pacing options, see "Controlling pacing" on page 470.

The ENCR option is ignored by LU 6.2; thus it is not included in the sample above.

Step 4: Populate the communications database

If you plan to use DB2 only as a server, you do not need to populate the CDB; default values are used. For example, Spiffy's USIBMSTODB21 subsystem works as a server for many OS/2 and Windows NT requesters. It is not necessary for Spiffy to register all those requesters in DB2's CDB.

However, if you intend to request data, you need to insert one row for each remote system into SYSIBM.LOCATIONS and SYSIBM.LUNAMES. You do not need to populate table SYSIBM.LULIST unless DB2 is acting as a requester of data that resides in a data sharing group. See *DB2 Data Sharing: Planning and Administration* for more information. Part 3 (Volume 1) of *DB2 Administration Guide* discusses the requirements for the other tables.

After you populate these tables, you can write queries that access data at a remote system. For instructions on sending SQL statements to other systems, see *DB2 Application Programming and SQL Guide*. For instructions on granting privileges to users on remote DB2 subsystems, see Part 3 (Volume 1) of *DB2 Administration Guide*.

SYSIBM.LOCATIONS table

The table LOCATIONS has the following purposes:

- When you do an SQL CONNECT, the LOCATION column maps the location name (or DRDA RDBNAM) to the VTAM LU name and, if necessary, the transaction program names (TPNs).
- When your DB2 receives a request from another DB2 site (using DB2 private protocol access), it uses the LOCATION column to validate the requesting site's location name. (Only DB2 sites exchange location names in both directions.) You do not need to populate this table for systems that use only DRDA access and make requests only of your local DB2.

LOCATIONS has the following columns relating to VTAM:

DBALIAS VARCHAR(128) NOT NULL	The name that is associated with the server. This name is used to access a remote database server. If DBALIAS is blank, the location name is used to access the remote database server. This column does not change database object names that are sent to the remote site using a location qualifier. Use the DBALIAS column to access data at two or more different remote locations when those remote locations have the same name. The LOCATION specifies where the database is in the network, and the DBALIAS is used to access the database server. This column does not change database object names that are executed in the application using the LOCATION. All fully qualified table names must reference the server's LOCATION name or one of its server LOCATION alias names, otherwise the SQL statement will fail because the table does not exist.
LOCATION CHAR(16)	The unique network location name, or DRDA RDBNAM, assigned to a system, remote or local. You must provide location names for any systems that you request data from. This column is the primary key for this table. If the remote LU exists in the same VTAM domain, specify the APPL name, not the ACBNAME. DBALIAS can override this name.
LINKNAME CHAR(8)	Identifies the VTAM attributes associated with this location. For each LINKNAME specified, you must have a row in SYSIBM.LUNAMES whose LUNAME matches the value specified in this column. Because this table is used for outbound requests, you must provide an LUNAME or your requests fail. Do not enter blanks in this column.
TPN VARCHAR(64)	This column is used to enter a transaction program name (TPN) for SNA conversations with non-DB2 systems. You only need to use this column if you are sending or receiving SQL requests from systems using non-default TPNs as described in "Step 2: Choose names and a password" on page 455.

Spiffy's USIBMSTODB21 location wants a LOCATIONS table that looks like Table 115 on page 466.

Table 115. Spiffy's LOCATIONS table

LOCATION	LINKNAME	TPN
USIBMSTODB21	LUDB21	
USIBMSTODB22	LUDB22	
USIBMSTOSQL1	LUSQLDS	TPNSQLDS2
USIBMSTOSQL2	LUSQLDS	TPNSQLDS1

For example, add the second row with this statement:

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
  VALUES ('USIBMSTODB22','LUDB22');
```

A row for the local location: You do not need a row for the local DB2 in the LUNAMES and LOCATIONS tables. For example, Spiffy's USIBMSTODB21 subsystem does not require a row that shows its own LU name and location name. However, for convenience, Spiffy decides to populate one LUNAMES table and one LOCATIONS table and to duplicate them entirely at each location. As a result, each table contains a row for its own LU name or location name.

SYSIBM.LUNAMES table

LUNAMES defines the security and mode requirements for conversations with other systems. Decisions about how to populate this table depend on how you intend to use DB2:

- If you use this system only as a server, DB2 can use a blank in the LUNAME column as a default. DB2 uses the values in the default row as defaults for LUs that are not explicitly defined in LUNAMES. If you do not have a row with a blank in the LUNAME column, DB2 rejects client connections that do not explicitly state a valid LUNAME. The DSNTIJSG installation job creates the default row in table SYSIBM.LUNAMES.
- If this DB2 requests data from other systems, you need to provide LU names for those systems.If the remote LU exists in the same VTAM domain, specify the APPL name, not the ACBNAME.

LUNAMES has the following columns:

LUNAME CHAR(8)
: The LU name of the remote system. The default of 8 blanks indicates that this row is used for serving the requests of any system that is not specifically listed in the LUNAMES table. For example, because USIBMSTODB21 acts strictly as a server for many OS/2 requesters, Spiffy leaves the LUNAME column blank for those requesters and uses default values for the entire row.

  However, you must provide LU names for any remote system that uses different values from the defaults.

SYSMODENAME CHAR(8)
: The mode used to establish system-to-system conversations for DB2 private protocol access. This column is ignored for DRDA access conversations. For now, Spiffy leaves it blank to use the default mode, IBMDB2LM, which they entered in step 3.

SECURITY_IN CHAR(1)
: Defines the security options that are accepted by this DB2 subsystem when an SNA client connects to DB2. The default, A, means that an incoming connection request is accepted if it includes any of these:

- A user ID
- A user ID and password
- A user ID and RACF PassTicket
- A Kerberos security ticket.

SECURITY_OUT CHAR(1)
: Defines the security option that is used when local DB2 SQL applications connect to any remote server associated with this LUNAME. The default, A, means that outgoing connection requests contain an authorization ID without a password.

ENCRYPTPSWDS CHAR(1)
: For now, Spiffy uses a blank to indicate no encryption of passwords. For more information about using this column for security, see Part 3 (Volume 1) of *DB2 Administration Guide*.

MODESELECT CHAR(1)
: Determines whether to use the default mode or to choose a mode from the MODESELECT table. Spiffy uses a blank to use the default modes: IBMDB2LM for conversations using DB2 private protocol access and IBMRDB for conversations using DRDA access. For more information about this column, and other tables in the CDB, see “Associating applications with modes” on page 473.

USERNAMES CHAR(1)
: This column is used for inbound and outbound requests to control authorization ID translation.

  Spiffy uses a blank to indicate that no authorization IDs are translated, and also that no passwords are sent to the server. For more information, see Part 3 (Volume 1) of *DB2 Administration Guide*.

GENERIC CHAR(1)
: A Y in this column indicates that a generic LU name is to be used for CNOS processing and SQL requests sent to the partner LU. A value of N or a blank indicates that the name specified in the LUNAME column is to be used. See Chapter 4 of *DB2 Data Sharing: Planning and Administration* for instructions about setting up a generic LU name.

Spiffy’s USIBMSTODB21 location wants a LUNAMES table that looks like Table 116.

Table 116. Spiffy’s SYSIBM.LUNAMES table. The row of blanks is a default row that Spiffy intends to use for Windows NT and OS/2 requesters in its initial testing.

LUNAME	SYSMODENAME	USERSECURITY[1]	ENCRYPTPSWDS	MODESELECT	USERNAMES
LUDB21					
LUDB22					
LUSQLDS					
(blanks)					

Note: [1] USERSECURITY refers to SECURITY_IN AND SECURITY_OUT

Spiffy can use an SQL INSERT statement to add the appropriate rows. For example, they add the LU name for USIBMSTODB22 with this statement:

```
INSERT INTO SYSIBM.LUNAMES (LUNAME)
  VALUES ('LUDB22');
```

SYSIBM.USERNAMES table

SYSIBM.USERNAMES contains information needed for outbound and inbound ID translation and also for *come from* checking. See Part 3 (Volume 1) of *DB2 Administration Guide* for information about populating this table.

Step 5: Start VTAM to use DB2

You do not need to code any special VTAM start options to use DB2, but you can tailor start option values for DB2 communications. For more information on start options, see *VTAM for MVS/ESA Resource Definition Reference*.

You must start VTAM before starting DDF. For information on how to start VTAM, see *VTAM for MVS/ESA Network Implementation Guide*.

There are two VTAM libraries that must contain definitions for DB2:

- SYS1.VTAMLST contains the definitions that define DB2 as a VTAM application. "Step 3: Define the DB2 subsystem to VTAM" on page 457 contains more information about these definitions.

  You can use the following VTAM command to enable DB2, assuming that the member DB2APPLS contains definitions for DB2:

  ```
  V NET,ACT,ID=DB2APPLS
  ```
- SYS1.VTAMLIB contains mode table definitions used by DDF. This must be an APF-authorized library, or in a concatenation of APF-authorized libraries. For more information about modes and mode tables, see "The MODEENT macro" on page 462.

Step 6: Tune the system

As you begin testing with DB2's distributed data facility, you probably need to modify VTAM options and CDB values to handle certain problems. We highly recommend that you consult a VTAM communications expert to tune your network. This topic describes, at a fairly high level, some of the things to consider when tuning VTAM for DDF. See also Part 5 (Volume 2) of *DB2 Administration Guide* for more information about monitoring and tuning DB2 distributed applications.

In this topic we describe:

- "Controlling buffer storage" on page 469.

  By sending large amounts of data through the network, DB2 can cause problems with your VTAM I/O buffer pool.
- "Controlling pacing" on page 470.

  You probably need to tune your pacing options if your VTAM buffers become overloaded with data that is sent to this local DB2.
- "Modifying default session limits" on page 472

  Consider modifying session limits if you have problems obtaining enough sessions to handle your distributed workload efficiently.
- "Modifying class of service" on page 472.

  Specifying a class of service can help you assign priorities to your network applications.
- "Associating applications with modes" on page 473.

  Tuning the system can require that you add new modes to your log mode table so that there is a greater variety of classes of service available for your sessions.

This variety allows you to have more flexibility in tuning the system for specific uses. This topic tells you how to associate specific sessions with modes.

Before you begin tuning the network, you must understand the relationship between VTAM options and associated values in DB2's CDB. Table 117 summarizes the relationship.

Table 117. Relationship between DB2's CDB and VTAM macros

Macro Name	Option	CDB table.column	Relationship
APPL	*name*	LOCATIONS.LINKNAME LUNAMES.LUNAME LUMODES.LUNAME MODESELECT.LUNAME USERNAMES.LINKNAME LULIST.LINKNAME	The LU name used in VTAM communication. This name maps 1:1 to the system's location name in LOCATIONS.
APPL	DSESLIM	LUMODES.CONVLIMIT	CONVLIMIT overrides session limits specified with DSESLIM. Session limit values are used in CNOS processing.
MODEENT	LOGMODE	LUNAMES.SYSMODENAME	MODENAME chooses the mode for the system conversation in DB2 private protocol access.
MODEENT	LOGMODE	LUMODES.MODENAME	LUMODES creates session limits for *specific* LU name and mode name combinations.
MODEENT	LOGMODE	MODESELECT.MODENAME	MODESELECT maps authorization IDs and plans to specific modes.

Controlling buffer storage

VTAM uses buffer pools for control blocks, network traffic data, and channel programs. A shortage of buffer pools, can have an adverse effect on VTAM CPU time, storage consumption, and the ability to serve DB2 requests.

You can monitor VTAM buffer pools using one of the following:

- The VTAM command DISPLAY NET,BFRUSE
- A VTAM trace, obtained by entering the following z/OS MODIFY command:
  ```
  F procname,TRACE,TYPE=SMS,ID=VTAMBUF
  ```
  Procname in the command is the VTAM start procedure name. The data is collected by the generalized trace facility (GTF).

DB2 applications can consume a large number of VTAM IOBUF pool buffers, depending on the VTAM options you choose and the volume of data being transmitted between distributed systems. You can estimate the number of IOBUF pool buffers, and the storage they use, by using the formula described in "Calculating VTAM I/O buffer pool (IOBUF) storage" on page 479.

The VTAM IOBUF pool is an area of storage that VTAM uses to store messages that are exchanged between network resources. The IOBUF pool is shared among all VTAM resources. When you calculate the IOBUF storage required to satisfy DB2 requirements, keep in mind that the IOBUF pool must have enough space to satisfy requests from other VTAM applications as well, such as TSO, CICS, and IMS.

To prevent shortages of these VTAM buffers (IOBUFs), you can do any of the following:

Increase the number of IOBUF buffers

The IOBUF pool definition is one of the VTAM start options. You can enter the IOBUF option from the z/OS console, or you can include it at VTAM startup in SYS1.VTAMLST in member ATCSTRxx.

Tuning the IOBUF pool encompasses both base allocation and dynamic expansion values. At installation, you can specify a base allocation for the IOBUF pool (in number of buffers) and a dynamic expansion (in number of buffers). When storage runs short in the buffer pool, VTAM temporarily expands the IOBUF pool by the dynamic expansion value, based on a trigger which you can also specify in VTAM definitions. Recommendation: Set a maximum size for the IOBUF pool size using the *xpanlim* start option for the buffer pool. If you turn off pacing accidentally, *xpanlim* prevents DB2 from causing VTAM to grab unlimited amounts of storage. For more information about allocating buffer pools, see *VTAM for MVS/ESA Network Implementation Guide*.

Decrease the session level pacing count

Pacing is vital for controlling the potentially large amounts of data that are transferred around the network. See "Controlling pacing" for more information about modifying pacing values.

Decrease the number of concurrent conversations

You can reduce the number of concurrent conversations by reducing the number of sessions. See "Modifying default session limits" on page 472 for more information about how to do this.

Decrease the request unit (RU) size

The RUSIZES option is part of the mode entry statement described in "The MODEENT macro" on page 462.

Because reducing the number of sessions and the RUSIZES value can adversely affect performance, you should first consider increasing IOBUF buffers and decreasing the session pacing count.

Controlling pacing

Session level pacing is the mechanism by which the receiver of data (DB2 in this case) can control the pace at which the sender sends data (in the form of RUs). The pacing size is the number of RUs VTAM sends across the line at one time, and you set that value using the VPACING option of the VTAM APPL definition statement described in "The APPL statement" on page 458. You set the RU size in the MODEENT macro described in "The MODEENT macro" on page 462. The receiving VTAM stores these RUs in its IOBUF pool; it uses pacing so that its buffers do not become flooded with data.

The pacing process works as shown in Figure 90 on page 471. The system at the sending side (let's assume it is USIBMSTODB22) passes data to its VTAM system. VTAM formats the data into RUs and sends those RUs across the network. If, for example, the pacing size is 2, then it sends two RUs. A 29-byte network header is sent with each RU.

After the USIBMSTODB22 VTAM system sends the specified number of RUs, it does not send any more data on this session until it receives a pacing response from the VTAM system at USIBMSTODB21. The USIBMSTODB21 VTAM system does not send a response until VTAM transfers the data into the DB2 buffers.

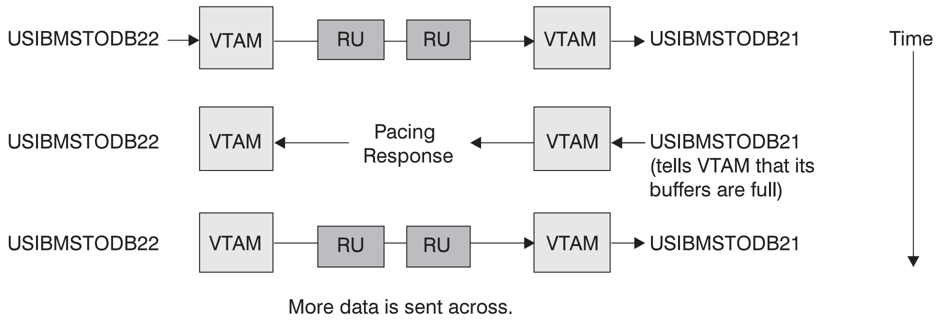


Figure 90. How pacing works

Although it is generally true that the receiving system can control inbound pacing, both communicating systems negotiate final pacing values. For more information about pacing negotiation, see *VTAM for MVS/ESA Network Implementation Guide*.

Recommendation for APPL pacing option

The VPACING option of the APPL statement is the maximum number of RUs that another LU can send, on a session, to this LU before waiting to receive a pacing response. This should always be a nonzero value, or else you turn off all pacing for all sessions affected by this option.

Recommendation: Start with a value of 2 for both communicating systems. This pacing size is the same in both directions for all modes.

The VPACING value is used with the RUSIZES option of the MODEENT macro to control the pacing window size. Thus, if VPACING is 2 and RUSIZES is 4–KB (X'8989'), then about 2 × 4KB = 8KB are sent before waiting to receive a pacing response. You should verify that your VTAM buffer pools are large enough to accommodate the chosen pacing and RU sizes. See “Calculating VTAM I/O buffer pool (IOBUF) storage” on page 479 for information about calculating buffer pool sizes.

Recommendation for MODEENT pacing options

The MODEENT macro described on 462, contains several pacing options:

PSNDPAC This does not apply to DB2; therefore, you can ignore this option.

SSNDPAC This option is really a flag that you set to either 0 (off) or nonzero (on). When 0, outbound pacing for sessions is disabled, which can lead to severe problems with IOBUF storage. **Recommendation:** Specify a nonzero value for this option.

SRCVPAC If 0, which is the recommended value, the VPACING option of the VTAM APPL statement controls both the send and receive pacing for all sessions in all modes. A value of 0 makes it easier for you to predict pacing results and makes it easier to maintain your pacing definitions.

If nonzero, VPACING controls pacing in one direction, and SRCVPAC controls it in another. LU 6.2 protocols make it difficult to predict which option is in control at any given time.

Modifying default session limits

An understanding of session limits helps you control resource use. DRDA access uses one session per partner. A given application can connect to many partners at any time.

DB2 private protocol access can take advantage of more sessions. For best performance, every read-only cursor in an application can use its own conversation. However, there can be resource constraints that disallow so many sessions. When conversation limits have been reached, DB2 begins sharing available conversations, if the application already owns one or more VTAM conversations. This means that, if an application has acquired its first conversation, it is not rejected because of a shortage of conversations.

As you begin increasing the number of applications that use the distributed data facility, you might find that your applications are waiting for conversations to become available, thus increasing the network delay associated with the application. Therefore, this topic also tells how to increase your default maximum session limit to ensure enough resources for best performance.

Increasing session limits for specific modes and LUs

To fine tune session limits for specific LU name and mode name combinations, you can modify the CONVLIMIT column of the SYSIBM.LUMODES table. To calculate CONVLIMIT, follow up to step 6 in "Calculating session limits for DB2 private protocol access" on page 477.

If you have specified OPERCNOS=ALLOW in the VTAM APPL statement, you can change the session limits dynamically with the VTAM command MODIFY VTAM,DEFINE. See *VTAM for MVS/ESA Operation* for more information about VTAM commands.

Increasing session limits for all modes and LUs

You can modify the overall session limit default by modifying the DSESLIM option of the VTAM APPL statement. (Remember: a value in CONVLIMIT for a given LU name and mode name combination still overrides DSESLIM.) The suggested value for DSESLIM is the maximum number of sessions that could possibly be in use on any single mode. The procedure for determining session limits is in "Calculating session limits" on page 476.

Modifying class of service

You can define the transmission priority and paths between systems with entries into a class of service (COS) table. Each entry in the table is associated with a list of routes to be used with a particular class. For example, you might want to place interactive sessions on a faster route than a batch job.

To specify a class of service for a specific mode, use the COS (class of service name) option of the MODEENT macro. When you specify a name of a COS entry in the mode description, you select the list of routes you want to be used for the session. When VTAM establishes the session, it chooses the first available route in the list of routes you tell it to use for that class.

If you do not specify a COS name, the mode gets the default list of routes from VTAM.

Associating applications with modes

The information under this heading, up to “Updating CDB values” on page 476 is General-use Programming Interface, as defined in “Notices” on page 545.

As you tune your system, you can assign certain applications, such as a high priority job, to the mode that is best suited for that job. You can also use a specific mode assignment for an application that uses many conversations. You can assign such an application to a mode that allows more conversations than the VTAM DSESLIM value you entered in the APPL statement.

To associate a specific mode with a particular session, you need to update or insert rows into three tables in the CDB: LUNAMES, LUMODES, and MODESELECT.

For information about when updates to the CDB are activated, see “Updating CDB values” on page 476.

Update LUNAMES to associate modes with LU names

This table is described in “Step 4: Populate the communications database” on page 464. It associates a mode with each remote system that the local subsystem can send a query to. There are two types of connections that you can specify in this table:
- System conversation (used only for DB2 private protocol access); not the recommended connection type
- SQL processing conversations (can be many for DB2 private protocol access; only one for DRDA access)

System conversation: For DB2 private protocol access, the *system conversation* must be established before any processing can begin. To choose a system mode, DB2 looks in the SYSMODENAME column of the row in the LUNAMES table that corresponds to the target DB2.

If SYSMODENAME is blank, then IBMDB2LM is used. If SYSMODENAME contains a mode name, then that mode is used when creating the first conversation between the two DB2 subsystems.

Recommendation: You should define a separate mode name for the system conversation. Assign a mode name to the SYSMODENAME column rather than leaving it blank. At the same time, ensure that the SYSIBM.MODESELECT table is defined to prevent other applications from using this mode. This defines a mode exclusively for DB2 system processing. The specified system conversation mode should be defined with a minimum of 3 sessions, allowing for two system conversations that always exist once a remote system is accessed via private protocols, and a third for obtaining Parallel Sysplex balancing information. The advantage of doing this is that system conversations and SQL conversations will not contend with each other for sessions.

SQL processing conversations: For DRDA access, an SQL processing conversation is established. The mode name for SQL processing conversations is determined by the MODESELECT table of the CDB. If the MODESELECT column of LUNAMES table is blank or contains N, then the default mode (IBMRDB) is used. If it contains a Y, then MODESELECT is searched.

Spiffy wants to use the mode LOC2MODE for system conversations with USIBMSTODB22’s DB2, using DB2 private protocol access. They also want to begin setting up different modes for specific applications to use in conversations with USIBMSTODB22. They enter into their DB2 mode table a mode named

LOC2MODE. In a later step, they define LOC2MODE in MODESELECT; for now, they update the LUDB22 row of LUNAMES as shown in Table 118.

Table 118. Spiffy's LUNAMES table, after update

LUNAME	SYSMODENAME	USERSECURITY[1]	...	MODESELECT	...
LUDB22	LOC2MODE			Y	

Note: [1]USERSECURITY represents SECURITY_IN and SECURITY_OUT

Spiffy can use the UPDATE statement below to make the change, which takes effect the next time DDF is started. (It takes effect immediately if DDF is started but USIBMSTODB22 is not yet accessed.)

```
UPDATE SYSIBM.LUNAMES
SET MODENAME='LOC2MODE', MODESELECT='Y'
WHERE LUNAME='LUDB22';
```

Update SYSIBM.LUMODES with conversation limits

Populating this table is optional; if you do not specify mode names in this table, the VTAM defaults are used. Use this table to provide VTAM with conversation limits for *specific* LU name and mode name combinations. (The table is unlike the DSESLIM option of the VTAM APPL definition statement, which provides the default session limits for *all* LU name and mode name combinations.) The primary key for this table is formed by the LU name and mode name combination. Only one entry with the same LU name and mode name is allowed.

LUMODES is accessed for negotiation of session limits with a remote DB2 for a specific mode. This negotiation is called *change number of sessions* (CNOS), which is discussed in "Interpreting CNOS messages" on page 480.

For example, suppose Spiffy wants to allocate 75 sessions instead of 50 (the value in DSESLIM) for conversations to USIBMSTODB22, using the mode named LOC2MODE. They can use the INSERT statement below to update the value in the CONVLIMIT column to 75. The new session limit takes effect the next time DDF is started, or in the initial connection to this LU for this mode.

```
INSERT INTO SYSIBM.LUMODES VALUES ('LUDB22','LOC2MODE',75,'N');
```

CNOS processing negotiates a value that is the lesser of the number of sessions available at either system for that mode. Therefore, USIBMSTODB22 must also increase its CONVLIMIT value to derive any benefit from the added sessions.

Columns of the LUMODES table:

LUNAME CHAR(8)
: Again, this is the LU name of the other system. This column is a foreign key of the LUNAMES table; thus, all LU names defined in this table must be defined in LUNAMES. When you delete an LU name from the LUNAMES table, all associated rows in LUMODES are deleted.

MODENAME CHAR(8)
: The name of the logon mode description in the VTAM logon mode table that VTAM uses when creating a conversation to support the local DB2's request for data from another system. The mode named here must exist in the mode table used by DB2 before a conversation can be created between USIBMSTODB21 and USIBMSTODB22.

CONVLIMIT SMALLINT
: The maximum number of conversations to be concurrently active between

this DB2 subsystem and the other system for this mode. This number is overrides the number in the DSESLIM option of the VTAM APPL definition statement during CNOS processing, as described in “Interpreting CNOS messages” on page 480.

Update SYSIBM.MODESELECT to associate plans with modes

This table maps authorization IDs and plan names to mode names. The primary key for this table is the combination of AUTHID, LUNAME, and PLANNAME. Only one entry with the same AUTHID, LUNAME, and PLANNAME is allowed.

Use this table to make sure that certain authorization IDs using certain plans always have a predefined class of service suited for that operation. For example, the USIBMSTODB21 location might want to work with USIBMSTODB22 to set up a high performance mode for DBADM to run queries to USIBMSTODB22. After the following statement is committed, all subsequent threads to USIBMSTODB22 use mode DB2MODE1 to process SQL processing conversations:

```
INSERT INTO SYSIBM.MODESELECT VALUES ('DBADM',' ','LUDB22','DB2MODE1');
```

Populating this table is optional. If the remaining columns are blank for any given LU name, then the mode name applies to all authorization IDs for all PLANNAMEs accessing the given LU name.

Columns of the MODESELECT table:

AUTHID CHAR(8)
: The authorization ID of the request for data from another system. A blank AUTHID indicates that the specified mode name applies to all authorization IDs. Blank is the default.

PLANNAME CHAR(8)
: The plan name associated with the request for data from another system. A blank plan name indicates that the specified mode name applies to all plan names. Blank is the default.

LUNAME CHAR(8)
: The LU name to which the specific mode name applies. This column is a foreign key of the LUNAMES table; therefore, all LU names defined in this table must be defined in LUNAMES.

MODENAME CHAR(8)
: The name of the logon mode description in the VTAM logon mode table that is used when creating a conversation to support the request for data from another system. If this column is blank, the default mode (IBMDB2LM or IBMRDB) is used.

How an SQL processing conversation mode is chosen:

The MODESELECT table of the CDB is used to choose a mode for an SQL processing conversation (if the MODESELECT column of the LUNAMES table contains Y for this LU name). Table 119 shows the search order of the MODESELECT table.

Table 119. Precedence search order for MODESELECT table of CDB

AUTHID	PLANNAME	Result
Name	Name	The MODENAME applies to the named AUTHID for the named PLANNAME accessing the named LU.

Table 119. Precedence search order for MODESELECT table of CDB (continued)

AUTHID	PLANNAME	Result
Name	Blank	The MODENAME applies to the named AUTHID for all PLANNAMEs accessing the named LU.
Blank	Name	The MODENAME applies to all AUTHIDs for the named PLANNAME accessing the named LU.
Blank	Blank	The MODENAME applies to all AUTHIDs for all PLANNAMEs accessing the named LU.

If the MODESELECT column of the LUNAMES table contains Y for a particular LU name and no row is found for that LU name in the MODESELECT table, then you receive a negative SQL return code when trying to access the system at that LU.

Plan name for remote bind operations: If you want to specify a particular mode for remote bind operations, use the plan name DSNBIND in MODESELECT.

Updating CDB values

Any table in the CDB can be updated while DDF is active. The changes take effect as follows:

- Changes to LUMODES take effect the next time DDF is started, or on the initial session to a given LUMODE combination.
- Changes to LUNAMES, LOCATIONS, and LULIST take effect as follows:
  - If DDF has not yet tried to communicate with a particular remote location, rows added to LUNAMES and LOCATIONS take effect when DDF attempts to communicate with that location.
  - If DDF has already attempted communication with a particular location, rows added to LUNAMES and LOCATIONS take effect the next time DDF is started.
- Changes to USERNAMES and MODESELECT take effect at the next thread access.

In all cases, existing conversations continue to operate as the table specified before the update.

The process of modifying the CDB, particularly MODESELECT and USERNAMES, can interfere with DDF's access to the tables. This could potentially cause deadlocks and timeouts, which cause the attempted access to the remote system to fail.

Calculating session limits

You might have to derive a precise figure for your session limits (DSESLIM). For example, if you specify a very large number for your session limits and you are running short of space, it might help to calculate a number closer to what you actually need. This topic tells you how to calculate session limit values based on whether the applications are using DRDA access or DB2 private protocol access.

If you have applications that use both DRDA access and DB2 private protocol access, first read this topic, then see "Considerations for mixed applications" on page 478.

The recommended session limit for a specific mode is the following:

- For DRDA access: the maximum number of concurrently active applications using that mode to and from the remote system.

  For example, suppose location A has a maximum of three DRDA access applications that can run concurrently on Mode1 to other locations. Also suppose that a maximum of 10 DRDA access applications can be incoming to location A on Mode1. Thus, Mode1 should support 13 sessions for DRDA access applications.
- For DB2 private protocol access applications: the maximum number of system conversations using that mode name to and from the remote DB2 subsystem, plus the maximum number of conversations needed for each concurrently active application using that mode to and from the remote DB2. The formula for calculating session limits for DB2 private protocol access is outlined in "Calculating session limits for DB2 private protocol access."

Calculating session limits for DB2 private protocol access

To calculate the session limits for location A, which uses DB2 private protocol access:

1. Determine all the applications that run concurrently on location A and use Mode1 to access location B. Call these A1, A2, ... AN.
2. For *each* of these applications, determine the total number of read-only cursors that can be concurrently active to location B and add 1. This additional conversation represents the cursor for SQL statements that modify data (UPDATE, DELETE, and INSERT) and for all SQL statements that do not need cursors. Call these numbers C1, C2, ... Cn, respectively. For example, if App_1 has three cursors opened concurrently, the value of C1 is 4.
3. Determine the total number of Mode1 sessions needed by location A to access location B by adding C1+C2+ ... CN=LIMIT_A.
4. If location A also uses Mode1 for the system conversation to location B, then add 1 to LIMIT_A.
5. Repeat steps 1 through 4 for location B applications accessing location A using Mode1. Call this result LIMIT_B.
6. The limit for Mode1 between location A and location B is LIMIT_A+LIMIT_B. This value can be entered as location A's CONVLIMIT for all conversations using Mode1 to access location B.
7. Repeat steps 1 through 6 for every possible mode between location A and location B. Get the maximum of all those numbers, and call it AB_MAX. This result is the maximum session limit between location A and location B.
8. Repeat all the above steps for every possible location that A can connect to.
9. Get the maximum of AB_MAX, AC_MAX, AD_MAX, and so on. This is DSESLIM.

Example: Assume that Mode2 could possibly be running the three applications described below concurrently.

App_1	App_2	App_3
Has 3 cursors open concurrently referencing location B.	Has 4 cursors open concurrently to location B. Has 6 cursors open concurrently to location C. Mode2 is used for system conversations to location B.	Has no open cursors. References location B only.

You can determine the mode limit for Mode2 as follows:

1. Use the directions in step 2 on page 477 to calculate the number of SQL conversations needed for location A to access location B. This number, LIMIT_A, is 10 (4 + 5 + 1).
2. Because Mode2 is used for the system conversation, add 1 to LIMIT_A, which results in a value of 11 for LIMIT_A.
3. Assume that the value of LIMIT_B is 16. Then, the limit for Mode2 as it accesses location B is 27. This value can be entered as the CONVLIMIT value in location A's LUMODES table.
4. The above steps must be repeated for A's connection to location C through App_2. This gives you the maximum number of sessions for Mode2.

This procedure has to be repeated for every mode to determine which mode uses the greatest number of sessions. This, then, is the value of DSESLIM.

Considerations for mixed applications

Applications can take advantage of both DB2 private protocol access and DRDA access to data. The following are possible ways to use both DRDA access and DB2 private protocol access in a single application:

- An application at the requesting site accesses a DB2 subsystem using DRDA access, then from there, "hops" to another DB2 subsystem or multiple DB2 subsystems using DB2 private protocol access. This "double hop" application is shown in Figure 91.

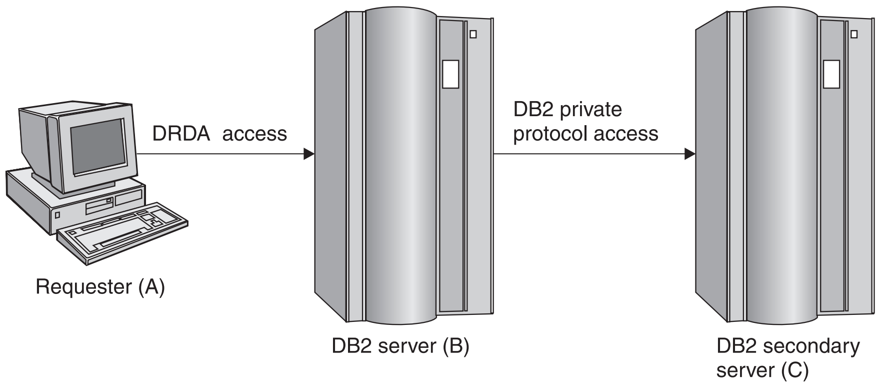


Figure 91. Example of a mixed application

The session requirements for such an application at each location are:

(A)	For the requester, the requirement is 1, which is the only number of sessions that can be used by an DRDA access application.
(B)	The requirement is the value for DB2 private protocol access (connection from B to C) plus 1 for the DRDA access connection from B to A.
(C)	The requirement is the value for DB2 private protocol access only. No DRDA access is possible at this site for this application.

- An application uses DRDA access to connect to one location, then drops that connection using SQL RELEASE and COMMIT statements, then uses DB2 private protocol access to access another DB2. (This same method could be used to access the same location; you would still have to drop your DRDA access connection to access DB2 using DB2 private protocol access.)

  This type of application is shown in Figure 92 on page 479.

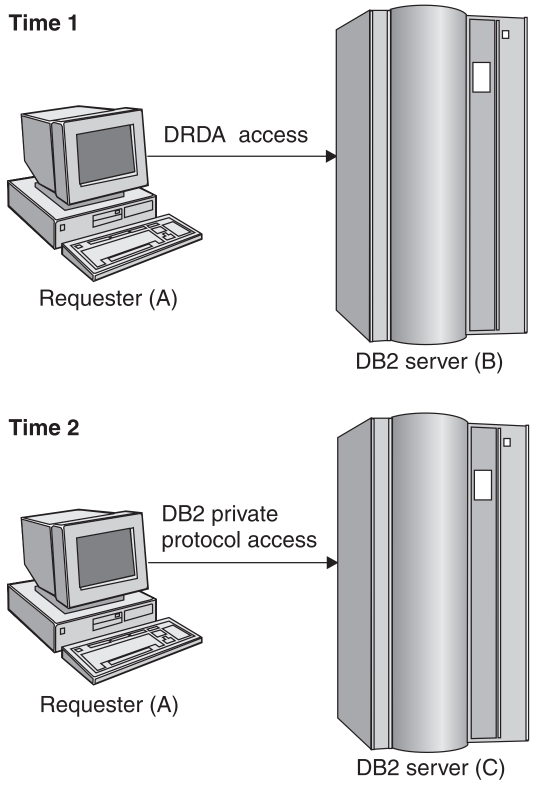


Figure 92. Another example of a mixed application

The session requirements for such an application at each location are:

(A)	The session requirement is the same as for DB2 private protocol access because only one type of access can be active at a time.
(B)	The requirement is 1, for the DRDA access connection.
(C)	The requirement is the value for DB2 private protocol access only.

Calculating VTAM I/O buffer pool (IOBUF) storage

This topic includes formulas for estimating VTAM buffer pool storage when using DB2's distributed data facility. Every *path information unit* (PIU) that enters or leaves VTAM resides in one or more IOBUF buffers.

A PIU is composed of a 26-byte transmission header, a 3-byte request/response header, and the request/response unit (RU) that contains VTAM application data. You define the length of the RU in a mode entry, using the RUSIZES option.

Do the following to calculate the maximum number of buffers required for the local DB2 subsystem:

1. Calculate the number of buffers that each PIU occupies, and call it PIUBUF.

   ```
   PIUBUF = CEILING(( 29 + RUSIZE ) / BUFSIZE)
   ```

   RUSIZE is the length of the RU in bytes. It is assumed to be the same for both session directions. BUFSIZE is the value you specified in the IOBUF pool definition.

   Assume you have a buffer size of 441 bytes, and an RUSIZE of 4096. With these values, PIUBUF would be 10 ((29+4096) / 441, rounded up).

For channel-to-channel (CTC) and NCP connections, you need to be concerned with the VTAM MAXBFRU value (as shown in “Channel-connected DB2s” on page 485 and “NCP-connected DB2s” on page 486). For CTC connections, MAXBFRU is the number of 4KB buffers allocated to hold the PIUs sent over the channel. If your RU size is 4096 and you allow 29 bytes for the header, then you need to allocate at least 2 4KB buffers. Thus, you need a MAXBFRU value of at least 2.

When you route data through NCP, MAXBFRU is the number of VTAM IOBUF buffers allocated to hold the PIUs sent to the NCP, which means MAXBFRU must be at least as large as PIUBUF.

2. Calculate the maximum number of IOBUF buffers used by a session, and call it SESSBUF.

   ```
   SESSBUF = PACECNT × PIUBUF
   ```

   PACECNT stands for pacing count. Pacing is discussed in greater detail in “Controlling pacing” on page 470, but for this example we assume that pacing is the same in both directions, and it is the same for all modes.

   If pacing is set to 2, then SESSBUF is 20.
3. Calculate the maximum number of sessions that can be active for all modes to all systems and call it SESCNT. Use the formula outlined in “Calculating session limits” on page 476 to calculate the maximum for each mode, then add those results to get SESCNT.
4. Calculate the maximum number of VTAM buffers used by DB2, and call this DB2BUF. The formula for DB2BUF is based on a worst case scenario, because it assumes that all sessions are used by concurrent conversations.

   ```
   DB2BUF = SESCNT × SESSBUF
   ```

   If we assume that the maximum number of sessions that can be active is 50 (SESCNT), then 1000 is the number of IOBUF entries required by DB2 in a worst case scenario.
5. Calculate actual VTAM buffer storage consumption used by DB2, and call it STORAGE.

   ```
   STORAGE = DB2BUF × (BUFSIZE + 71)
   ```

   Each buffer includes 71 bytes for VTAM internal headers.

   So, to continue the above example, we can estimate an upper value of real storage as follows:

   ```
   1000 × (441 + 71) = 500KB
   ```

Interpreting CNOS messages

DB2’s distributed data facility can request to alter the number of sessions with another system for a specific VTAM logon mode. This automatic process is called “change number of sessions” (CNOS). This topic contains a brief overview of the process as it relates to DB2; it should help you understand the messages that CNOS processing generates. For more information about CNOS processing in general, see *z/OS Communicatons Server SNA Programmer’s LU 6.2 Reference*.

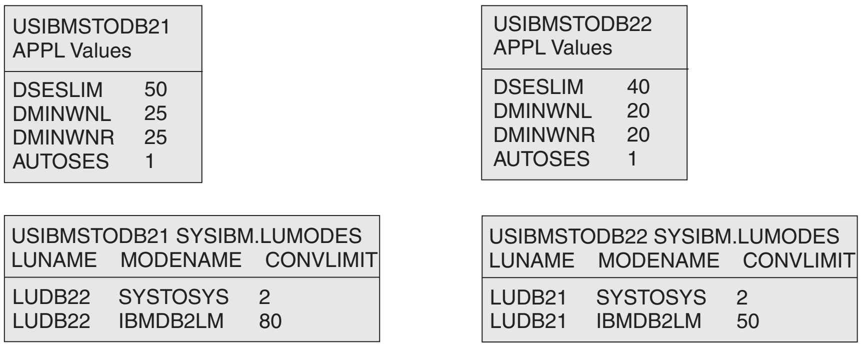
When sessions are started: The AUTOSES option of the VTAM APPL determines whether, and how many, sessions are started at the time CNOS is negotiated. If AUTOSES is 0, then the sessions are not started at CNOS negotiation time; they are started as they are needed. We do not recommend an AUTOSES of 0, because then DB2 is not informed if CNOS fails, and you receive a “resource unavailable” SQL code with the first SQL request to the remote system.

If AUTOSES is not 0, then sessions are started as follows:

- If AUTOSES is equal to or less than the number of contention winner sessions for a specific DB2 subsystem, then the number of sessions that are automatically started at CNOS negotiation is equal to AUTOSES.
- If AUTOSES is greater than the number of contention winner sessions for a specific DB2 subsystem, only the contention winner sessions are automatically started at CNOS negotiation.

Each LU has its own value for the number of contention winner sessions to start. The total number of sessions started on behalf of a CNOS negotiation request is the sum of the sessions started at each site.

Example: Suppose the DB2 subsystems at USIBMSTODB21 and USIBMSTODB22 have the following values in their VTAM APPL statements and LUMODES tables:



USIBMSTODB21 APPL Values	
DSESLIM	50
DMINWNL	25
DMINWNR	25
AUTOSES	1

USIBMSTODB22 APPL Values	
DSESLIM	40
DMINWNL	20
DMINWNR	20
AUTOSES	1

USIBMSTODB21 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB22	SYSTOSYS	2
LUDB22	IBMDB2LM	80

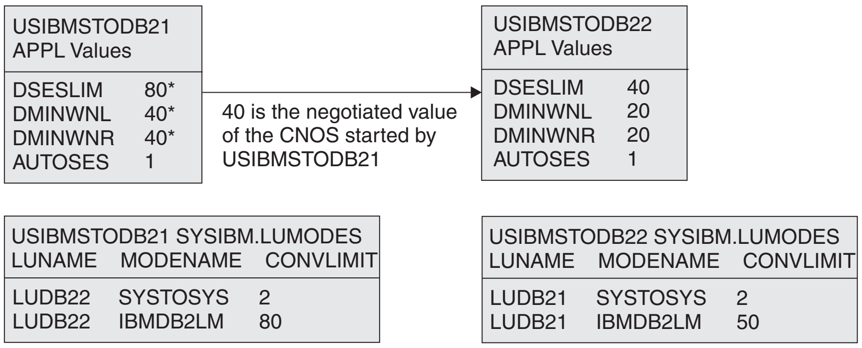
USIBMSTODB22 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB21	SYSTOSYS	2
LUDB21	IBMDB2LM	50

Figure 93. CNOS negotiation example: VTAM and DB2 definitions

Assume that USIBMSTODB21's DDF is started first. CNOS processing fails because USIBMSTODB22's DDF has not yet started, and you get a message at the console. When USIBMSTODB22's DDF is started, CNOS processing can begin.

USIBMSTODB21 sends to USIBMSTODB22 a CNOS value of 80, which is its CONVLIMIT value. However, USIBMSTODB22 replies with a value of 40 (its DSESLIM value), and, as shown in Figure 94, that becomes the negotiated value for the CNOS started by USIBMSTODB21. Both systems begin starting the number of sessions that are specified in their respective AUTOSES options.


40 is the negotiated value of the CNOS started by USIBMSTODB21

USIBMSTODB21 APPL Values	
DSESLIM	80*
DMINWNL	40*
DMINWNR	40*
AUTOSES	1

USIBMSTODB22 APPL Values	
DSESLIM	40
DMINWNL	20
DMINWNR	20
AUTOSES	1

USIBMSTODB21 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB22	SYSTOSYS	2
LUDB22	IBMDB2LM	80

USIBMSTODB22 SYSIBM.LUMODES

LUNAME	MODENAME	CONVLIMIT
LUDB21	SYSTOSYS	2
LUDB21	IBMDB2LM	50

Figure 94. Result of CNOS negotiation started by USIBMSTODB21. Overridden values are noted with asterisks (*).

VTAM does not start all 40 sessions unless the two AUTOSES values total up to 40 or greater. Instead, VTAM delays starting the other sessions until they are needed.

Everything up to this point occurred because USIBMSTODB21 issued CNOS. Now, as shown in Figure 95, USIBMSTODB22 starts CNOS processing back to USIBMSTODB21 because its CDB has CNOS limits specified (50 in CONVLIMIT). USIBMSTODB21's VTAM sees that DB2 allows up to 80, so VTAM sends the CNOS reply message back to USIBMSTODB22 unchanged (50). USIBMSTODB22's CONVLIMIT value of 50 is compared with USIBMSTODB21's overridden value of 80 from the previous CNOS, and 50 is chosen as the value.

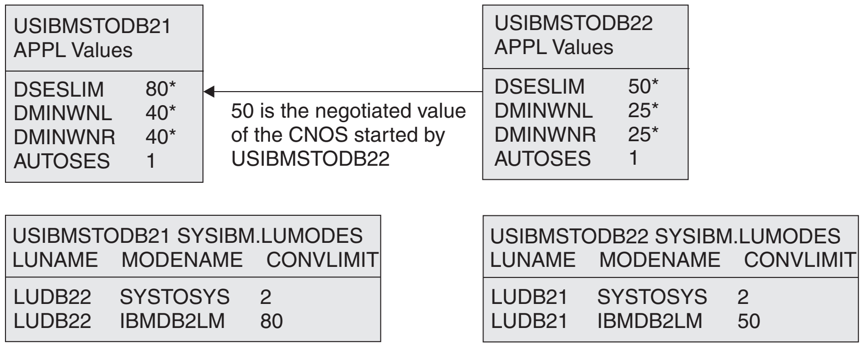


Figure 95. CNOS negotiation from USIBMSTODB22 to USIBMSTODB21. Overridden values are noted with asterisks (*).

If the new negotiated value is smaller than the number already started by USIBMSTODB21, then VTAM terminates the number of sessions that makes up the difference. If the CONVLIMIT value at USIBMSTODB22 is 20, for example, VTAM terminates 20 sessions on behalf of the request from USIBMSTODB22 because the lowest negotiated value always wins. If a session is currently being used by a conversation, the session is terminated as soon as the conversation is deallocated.

Sample VTAM definitions to connect two DB2s

These definitions are included to give you some guidance on setting up your network to connect two DB2s. It is not intended to give you information about all the options; the ones most relevant have been discussed previously in this chapter. We suggest you see VTAM publications for more information about VTAM options.

This set of definitions includes the basic definitions you need to connect two DB2s. Additional options for channel-to-channel and Network Control Program (NCP) connections are covered as well.

Basic definitions

The basic definitions here are required for all VTAM connections. For more information about these definitions, see *VTAM for MVS/ESA Resource Definition Reference*.

```
*********************************************************************
**      APPL STATEMENT FOR SYSTEM 1                                 *
*********************************************************************
DB1APPL  APPL APPC=YES,                                             X
         ATNLOSS=ALL,                                               X
         AUTH=(ACQ),                                                X
         AUTOSES=1,                                                 X
         DSESLIM=20,                                                X
         DMINWNL=10,                                                X
         DMINWNR=10,                                                X
         PRTCT=DB1PWD,                                              X
         SECACPT=ALREADYV,                                          X
         EAS=509,                                                   X
         MODETAB=DB2MODES,                                          X
         PARSESS=YES,                                               X
         SRBEXIT=YES,                                               X
         SYNCLVL=SYNCPT,                                            X
         VPACING=2
*********************************************************************
**      APPL STATEMENT FOR SYSTEM 2                                 *
*********************************************************************
DB2APPL  APPL APPC=YES,                                             X
         ATNLOSS=ALL,                                               X
         AUTH=(ACQ),                                                X
         AUTOSES=1,                                                 X
         DSESLIM=20,                                                X
         DMINWNL=10,                                                X
         DMINWNR=10,                                                X
         PRTCT=DB2PWD,                                              X
         SECACPT=ALREADYV,                                          X
         EAS=509,                                                   X
         MODETAB=DB2MODES,                                          X
         PARSESS=YES,                                               X
         SRBEXIT=YES,                                               X
         SYNCLVL=SYNCPT,                                            X
         VPACING=2
**********************************************************************
**    LOGMODE TABLE FOR SYSTEM 1 TO SYSTEM 2 CONNECTIONS
**********************************************************************
DB2MODES MODETAB
*********************************************************************
*     LU6.2 SERVICES MANAGER BASE MODE FOR APPC/VTAM
*********************************************************************
SNASVCMG MODEENT LOGMODE=SNASVCMG,                                  X
                 FMPROF=X'13',                                      X
                 TSPROF=X'07',                                      X
                 PRIPROT=X'B0',                                     X
                 SECPROT=X'B0',                                     X
                 PSERVIC=X'060200000000000000000300',               X
                 COMPROT=X'D0B1',                                   X
                 RUSIZES=X'8585',                                   X
                 ENCR=B'0000'
*********************************************************************
```

Figure 96. Basic VTAM definitions (Part 1 of 2)

```
** DB2 DEFAULT MODE FOR SYSTEM 1 AND SYSTEM 2 PRIVATE PROTOCOL      **
**********************************************************************
IBMDB2LM MODEENT LOGMODE=IBMDB2LM,                                   X
                 TYPE=0,                                             X
                 PSNDPAC=X'00',                                      X
                 SSNDPAC=X'02',                                      X
                 SRCVPAC=X'00',                                      X
                 RUSIZES=X'8989',                                    X
                 FMPROF=X'13', LU6.2 FM PROFILE                      X
                 TSPROF=X'07', LU6.2 TS PROFILE                      X
                 PRIPROT=X'B0', LU6.2 PRIMARY PROTOCOLS              X
                 SECPROT=X'B0', LU6.2 SECONDARY PROTOCOLS            X
                 COMPROT=X'50A5', LU6.2 COMMON PROTOCOLS             X
                 PSERVIC=X'060200000000000000122F00' LU6.2 LU TYPE
**********************************************************************
** DB2 DEFAULT FOR SYSTEM 1 AND SYSTEM 2 DRDA ACCESS                **
**********************************************************************
IBMRDB MODEENT LOGMODE=IBMRDB,                                       X
               TYPE=0,                                               X
               PSNDPAC=X'00',                                        X
               SSNDPAC=X'02',                                        X
               SRCVPAC=X'00',                                        X
               RUSIZES=X'8989',                                      X
               FMPROF=X'13', LU6.2 FM PROFILE                        X
               TSPROF=X'07', LU6.2 TS PROFILE                        X
               PRIPROT=X'B0', LU6.2 PRIMARY PROTOCOLS                X
               SECPROT=X'B0', LU6.2 SECONDARY PROTOCOLS              X
               COMPROT=X'50A5', LU6.2 COMMON PROTOCOLS               X
               PSERVIC=X'060200000000000000122F00' LU6.2 LU TYPE
***********************************************************************
**    DB2 SYSTOSYS MODE FOR SYSTEM 1 AND SYSTEM 2                     *
***********************************************************************
SYSTOSYS MODEENT LOGMODE=SYSTOSYS,                                    X
                 TYPE=0,                                              X
                 PSNDPAC=X'00',                                       X
                 SSNDPAC=X'02',                                       X
                 SRCVPAC=X'00',                                       X
                 RUSIZES=X'8989',                                     X
                 FMPROF=X'13', LU6.2 FM PROFILE                       X
                 TSPROF=X'07', LU6.2 TS PROFILE                       X
                 PRIPROT=X'B0', LU6.2 PRIMARY PROTOCOLS               X
                 SECPROT=X'B0', LU6.2 SECONDARY PROTOCOLS             X
                 COMPROT=X'50A5', LU6.2 COMMON PROTOCOLS              X
                 PSERVIC=X'060200000000000000122F00' LU6.2 LU TYPE
         MODEEND END
***********************************************************************
** ATCSTRTA VTAM START OPTIONS FOR SYSTEM 1, INCLUDES IOBUF
***********************************************************************
  CONFIG=TA,                                                          X
  SSCPID=53,MAXSUBA=150,HOSTSA=53,                                    X
  SSCPNAME=SSCP004,NETID=USIBMSY,                                     X
  IOBUF=(328,441,20,,64,48,768)
***********************************************************************
** ATCSTRTB VTAM START OPTIONS FOR SYSTEM 2, INCLUDES IOBUF
***********************************************************************
  CONFIG=TB,                                                          X
  SSCPID=54,MAXSUBA=150,HOSTSA=54,                                    X
  SSCPNAME=SSCP00E,NETID=USIBMSY,                                     X
  IOBUF=(328,441,20,,64,48,768)
```

Figure 96. Basic VTAM definitions (Part 2 of 2)

Channel-connected DB2s

When determining your channel-to-channel definitions, remember that MAXBFRU must be large enough to handle the largest PIU. Since DB2 is sending 4096 bytes, you need enough 4KB buffers to accept 4096 + 29 bytes (the 29 bytes is for the network header). Thus MAXBFRU must be at least 2 in our example.

In many cases, the DB2 RU size is larger than any other PIUs used on existing CTCs, which can mean you must examine your MAXBFRU values on existing CTC definitions. If the values are too small, you get an SNA X'800A' sense code, indicating that the PIU was truncated during transmission.

```
***********************************************************************
**       CTC DEFINITIONS FOR SYSTEM 1                                 *
***********************************************************************
DB1CTC   VBUILD TYPE=CA          CTC MAJOR NODE DEFINITION
DB1GRPB  GROUP  LNCTL=CTCA,      CTCA LINE TYPE                       X
                MIH=YES,REPLYTO=10.0
DB1CTCL  LINE   ADDRESS=(500),   CTC ADDRESS FOR THIS LINE            X
                DELAY=0,         CTC DELAY                            X
                MAXBFRU=8,       MAX BUFFER USED                      X
                ISTATUS=ACTIVE    INITIAL STATUS IS ACTIVE
DB1CTCP  PU     ISTATUS=ACTIVE
***********************************************************************
**       CTC DEFINITIONS FOR SYSTEM 2                                 *
***********************************************************************
DB2CTC   VBUILD TYPE=CA          CTC MAJOR NODE DEFINITION
DB2GRPB  GROUP  LNCTL=CTCA,      CTCA LINE TYPE                       X
                MIH=YES,REPLYTO=10.0
DB2CTCL  LINE   ADDRESS=(500),   CTC ADDRESS FOR THIS LINE            X
                DELAY=0,         CTC DELAY                            X
                MAXBFRU=8,       MAX BUFFER USED                      X
                ISTATUS=ACTIVE    INITIAL STATUS IS ACTIVE
DB2CTCP  PU     ISTATUS=ACTIVE
***********************************************************************
**       PATH - NETWORK ROUTES FOR SYSTEM 1                           *
***********************************************************************
MVSDB2   PATH DESTSA=2,ER1=(2,1),VR1=1,                               X
              VRPWS10=(2,30),VRPWS11=(2,30),VRPWS12=(2,30)
***********************************************************************
**       PATH - NETWORK ROUTES FOR SYSTEM 2                           *
***********************************************************************
MVSDB1   PATH DESTSA=1,ER1=(1,1),VR1=1,                               X
              VRPWS10=(2,30),VRPWS11=(2,30),VRPWS12=(2,30)
***********************************************************************
**      CDRSC DEFINITIONS FOR SYSTEM 1                                *
***********************************************************************
         VBUILD TYPE=CDRSC
DB2APPL  CDRSC CDRM=DB2CDRM,ISTATUS=ACTIVE
***********************************************************************
**      CDRSC DEFINITIONS FOR SYSTEM 2                                *
***********************************************************************
         VBUILD TYPE=CDRSC
DB1APPL CDRSC CDRM=DB1CDRM,ISTATUS=ACTIVE
```

Figure 97. Channel-to-channel (CTC) definitions (Part 1 of 2)

```
**********************************************************************
**      CDRM DEFINITIONS FOR SYSTEM 1 AND 2 (SAME DEFINITION USED)    *
**********************************************************************
         VBUILD TYPE=CDRM
DB1CDRM CDRM SUBAREA=1,ISTATUS=ACTIVE,CDRSC=OPT
DB2CDRM CDRM SUBAREA=2,ISTATUS=ACTIVE,CDRSC=OPT

**********************************************************************
**    ATCCONTA - NETWORK CONFIGURATION LIST FOR SYSTEM 1              *
**********************************************************************
DB1PATH,DB1CTC,DB1RSC,DB1APPLS,DBCDRMS
**********************************************************************
**    ATCCONTB - NETWORK CONFIGURATION LIST FOR SYSTEM 2              *
**********************************************************************
DB2PATH,DB2CTC,DB2RSC,DB2APPLS,DBCDRMS
```

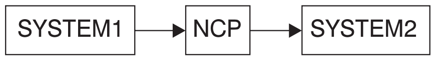
Figure 97. Channel-to-channel (CTC) definitions (Part 2 of 2)

NCP-connected DB2s

The Advanced Communications Facility/Network Control Program (ACF/NCP) is a product you use to generate a network control program load module, which is loaded from the host into a communications controller. The network control program controls the lines and devices attached to it. It transfers data to and from the devices and handles any errors that occur, including retries after line errors.

A communications controller can be locally attached to a host via a channel, or it can be link-attached to another communications controller that is channel-attached.

Our sample definitions are used for the following setup:



When you are defining your NCP connections, remember the following:

- MAXBFRU must be large enough to handle the biggest PIU that is sent to the NCP. In our example, DB2 is sending 4125 bytes per PIU (4096 + a 29-byte network header). Given an IOBUF buffer size of 441 bytes, MAXBFRU must therefore be at least 10 (10 × 441 = 4410, which is greater than 4125).
- The MAXDATA option must also be large enough to handle biggest PIU (RUSIZE + 29 bytes).

If DB2 is using existing NCP definitions, you should make sure your MAXBFRU and MAXDATA options are large enough. If these values are too small, you get an SNA X'800A' sense code, indicating that the PIU was truncated during transmission.

```
***********************************************************************
*       PCCU SPECIFICATION - FOR SYSTEM 1                             *
***********************************************************************
PCCU1    PCCU  CUADDR=C02,         3745 BLOCK CHANNEL                 X
               AUTOSYN=YES,                                           X
               AUTODMP=NO,                                            X
               AUTOIPL=NO,                                            X
               BACKUP=YES,                                            X
               DELAY=0,                                               X
               DUMPDS=DUMPDS,      DUMP DATA SET                      X
               CDUMPDS=CDUMPDS,    CSP DUMP DATA SET                  X
               MDUMPDS=MDUMPDS,    MOSS DUMP DATA SET                 X
               INITEST=NO,         NO 3745 INITIAL TESTS AT LOAD TIME X
               MAXDATA=4302,       = BFRS*TRANSFR - 18                X
               OWNER=HOST1,                                           X
               SUBAREA=3,          HOST SUBAREA                       X
               VFYLM=YES
***********************************************************************
*       PCCU SPECIFICATION - FOR SYSTEM 2                             *
***********************************************************************
PCCU2    PCCU  CUADDR=C02,         3745 BLOCK CHANNEL                 X
               AUTOSYN=YES,                                           X
               AUTODMP=NO,                                            X
               AUTOIPL=NO,                                            X
               BACKUP=YES,                                            X
               DELAY=0,                                               X
               DUMPDS=DUMPDS,      DUMP DATA SET                      X
               CDUMPDS=CDUMPDS,    CSP DUMP DATA SET                  X
               MDUMPDS=MDUMPDS,    MOSS DUMP DATA SET                 X
               INITEST=NO,         NO 3745 INITIAL TESTS AT LOAD TIME X
               MAXDATA=4302,       = BFRS*TRANSFR - 18                X
               OWNER=HOST2,                                           X
               SUBAREA=4,          HOST SUBAREA                       X
               VFYLM=YES
```

Figure 98. Network control program (NCP) definitions (Part 1 of 4)

```
**********************************************************************
*       BUILD MACRO SPECIFICATIONS                                   *
**********************************************************************
ACFNCPBD BUILD BFRS=(240),         NCP BUFFER SIZE,# EP BUFFERS       X
               BRANCH=1000,                                           X
               CATRACE=(YES,10),                                      X
               CSMHDR=27F5C711C3F0405C40C8C4D9405C,                   X
               CSMHDRC=40E3C5E7E3405C5C,                              X
               CSMSG=5C5C40E5E3C1D440E2C8E4E3C4D6E6D540,              X
               CSMSGC=6040C8C1E240C2C5C7E4D5405C5C,                   X
               DIALTO=60,          WAIT 1 MIN FOR AUTOCALL ANSWER     X
               DR3270=NO,          NO DYNAMIC RECONFIG                X
               DSABLTO=3.0,        TIME TO DETECT DSR DROP            X
               ENABLTO=2.2,        TIME TO DETECT DSR AFTER ENABLE    X
               LOADLIB=NCPLOAD,    LIBRARY FOR ACF/NCP LOAD MODULE    X
               LTRACE=8,           UP TO 8 LINES CONCURRENTLY TRACED  X
               MAXSSCP=8,          NUMBER OF SSCPS IN SESSION         X
               MAXSUBA=63,         MUST BE SAME AS IN ATCSTRXX        X
               MEMSIZE=4M,         AMOUNT OF MEMORY                   X
               MODEL=3745,         3745 MODEL 410                     X
               NETID=BCR1,         3745 MODEL 410                     X
               NEWNAME=DDBLC0,     LOAD MODULE NAME                   X
               PUNAME=DDB,                                            X
               NPA=YES,            NPA WILL NOT COLLECT DATA          X
               OLT=NO,             INCLUDE ONLINE TEST FACILITY-OLTEP X
               PRTGEN=NOGEN,       DON'T PRINT ASSEMBLED STATEMENTS   X
               PWROFF=NO,                                             X
               SLODOWN=15,         SLOWDOWN AFTER 15% BUFFERS AVAIL   X
               SUBAREA=26,         NCP SUBAREA                        X
               TRACE=(YES,10),     10-16 BYTE ADDRESS TRACE ENTRIES   X
               TRANSFR=18,         =(4096+51)/BFRS--ROUNDED UP        X
               TRCPIU=2000,        SIZE OF LINE AND SIT TRACE         X
               TYPGEN=NCP,                                            X
               TYPSYS=MVS,         MVS OPERATING SYSTEM               X
               USGTIER=5,          NCP USAGE TIER - REQUIRED          X
               VERSION=V5R3,                                          X
               XBREAK=NONE
```

Figure 98. Network control program (NCP) definitions (Part 2 of 4)

```
**********************************************************************
**                                                                  **
*        SYSCNTRL OPTIONS - REQUIRED BY VTAM                         *
**                                                                  **
**********************************************************************
         SYSCNTRL OPTIONS=(ENDCALL,MODE,RCNTRL,RCOND,RECMD,RIMM,     X
               SESSION,NAKLIM,LNSTAT,SSPAUSE,XMTLMT,BHSASSC,STORDSP)
**********************************************************************
HOST2    HOST  BFRPAD=0,           VTAM REQUIREMENT FOR OS           X
               INBFRS=18,          INITIAL BUFFERS FOR EACH RECEIVE  X
               MAXBFRU=10,         < BASENO IN IOBUF FOR VTAM        X
               SUBAREA=4,                                            X
               UNITSZ=441          = BUFSIZE IN IOBUF FOR VTAM

HOST1    HOST  BFRPAD=0,           VTAM REQUIREMENT FOR OS           X
               INBFRS=18,          INITIAL BUFFERS FOR EACH RECEIVE  X
               MAXBFRU=10,         < BASENO IN IOBUF FOR VTAM        X
               SUBAREA=3,                                            X
               UNITSZ=441          = BUFSIZE IN IOBUF FOR VTAM

**********************************************************************
*             PATH STATEMENTS                                        *
**********************************************************************
          PATH DESTSA=3,           SYS1                              X
               ER4=(3,1),          SYS1

          PATH DESTSA=4,           SYS2                              X
               ER4=(4,1),          SYS2
```

Figure 98. Network control program (NCP) definitions (Part 3 of 4)

```
*******************************************************************
*                                                                 *
*   HOST 1 CHANNEL ADAPTER                                        *
*                                                                 *
*   LINE ADDR = 0; PHYSICAL POSITION = 5.                       *
*******************************************************************
DDBCA5   GROUP LNCTL=CA,                                            X
               ISTATUS=INACTIVE    STOP VTAM FROM ACT CHAN LINK

DDBL05   LINE  ADDRESS=0,         1ST CA PHYSICAL POSITION 1         X
               CA=TYPE6,          3745 CHANNEL ADAPTER TYPE          X
               CASDL=120,         TIME ALLOWED TO BLOCK INBOUND DATA X
               DELAY=0,           CHAN ATTN DELAY                    X
               DYNADMP=NONE,      NO EP SUBCHANNELS TO DUMP          X
               INBFRS=18,         # BUFS FOR EACH TRANSFER TO HOST   X
               NPACOLL=YES,       NPA WILL COLLECT DATA ON CHANNEL   X
               TIMEOUT=120        INTERVAL BEFORE CHANNEL DISCONTACT

DDBP05   PU    PUTYPE=5,          INTERMEDIATE SUBAREA FUNCTION      X
               TGN=1              MUST BE 1 FOR PUTYPE5
*********************************************************************
*                                                                   *
*   HOST 2 CHANNEL ADAPTER                                          *
*   LINE ADDR = 2; PHYSICAL POSITION = 7.                         *
*********************************************************************
DDBCA7   GROUP LNCTL=CA,                                            X
               STATUS=INACTIVE    ACT CHAN LINK

DDBL07   LINE  ADDRESS=2,         3RD CA PHYSICAL POSITION 3         X
               CA=TYPE6,          3745 CHANNEL ADAPTER TYPE          X
               CASDL=120,         TIME ALLOWED TO BLOCK INBOUND DATA X
               DELAY=0,           CHAN ATTN DELAY                    X
               DYNADMP=NONE,      NO EP SUBCHANNELS TO DUMP          X
               INBFRS=18,         #BUFS FOR EACH TRANSFER TO HOST    X
               NPACOLL=YES,       NPA WILL COLLECT DATA ON CHANNEL   X
               TIMEOUT=120        INTERVAL BEFORE CHANNEL DISCONTACT

DDBP07 PU      PUTYPE=5,          INTERMEDIATE SUBAREA FUNCTION      X
               TGN=1              MUST BE 1 FOR PUTYPE5
```

Figure 98. Network control program (NCP) definitions (Part 4 of 4)

Using the change log inventory utility to update the BSDS

Use the options of the DDF statement of the change log inventory utility to insert or update the values listed below.

To **update** any value, you need only the option for that value.

To **insert** new values, you need values for LOCATION and LUNAME as well as any other values.

Value	**Option for inserting or updating**
Location alias	ALIAS = *name(s)*
Location name	LOCATION=*name*
LU name	LUNAME=*name*
Generic LU name	GENERIC=*name*
Password	PASSWORD=*password*
TCP/IP connection port	PORT=*number*
TCP/IP resync port	RESPORT=*number*

PASSWORD is optional, depending on whether you entered a password in the VTAM APPL statement. GENERIC, ALIAS, PORT, and RESPORT are also optional.

If you specify values for PORT and RESPORT, the values must be different and nonzero. You can delete both ports by entering zero.

You can add or delete aliases by respecifying the ALIAS names. The new list of names replaces the existing list.

To **delete** either a generic LU name, a password, or an alias, use one of these keywords:

Value	**Statement for deleting**
Generic LU name	NGENERIC
Password	NOPASSWD
Alias	NOALIAS

For more information about the change log inventory utility, see Part 3 of *DB2 Utility Guide and Reference*.

Chapter 14. Connecting systems with TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is a standard communication protocol for network communications. Previous versions of DB2 supported TCP/IP requesters, although additional software and configuration was required. Native TCP/IP eliminates these requirements, allowing gateway-less connectivity to DB2 for systems running UNIX System Services.

Terminology: The following communications terms are used in this chapter:

IP address
Uniquely identifies a host within the TCP/IP network. This is sometimes called an internet address. A DB2 subsystem resides on a TCP/IP host. The IP address is a four-byte address displayed in dotted decimal format: X'05041020' displays as 5.4.16.32

Domain name
The fully qualified name that identifies an IP address. This can be used instead of the IP address. An example of a domain name is *stlmvs1.stl.ibm.com*. Some software refers to *stlmvs1* as the host name and *stl.ibm.com* as the domain name. DB2 allows the network administrator to identify a host using a domain name.

Domain name server (DNS)
Manages a distributed directory of domain names and related IP addresses. Domain names can be translated into IP addresses and you can find a domain name associated with a given IP address. DB2 uses the *gethostbyname* service to get a list of IP addresses for a given domain name.

Port Identifies an application executing in a host. For example, a port number identifies a DB2 subsystem to TCP/IP. A port number is a two byte integer value that is displayed in decimal format. This number identifies the application within a TCP/IP instance. A port number of X'01D2' displays as *466*. There are three basic kinds of TCP/IP ports:

Well-known port
This is a port number between 1 and 1023 that is reserved in the TCP/IP architecture for a specific TCP/IP application. Some typical well-known port numbers are:
- FTP is port number 21
- Telnet is port number 23
- DRDA relational database is port number 446

Ephemeral port
Port numbers that are dynamically assigned to a client process by the client's TCP/IP instance. DB2 uses an ephemeral port when it is acting as the DRDA requester. This ephemeral port is associated with the requester for the life of the thread, or connection.

Server port
Port numbers that are used when a TCP/IP program does not have a well-known port number, or another instance of the server program is already installed using the well-known port number. As a requester, DB2 defaults to using the DRDA relational database well-known port number to connect to a server location. We suggest that your DB2 subsystem be defined with the DRDA well-known port number of 446. However, the network administrator can assign a server port to the DB2 subsystem. If two

different DB2 subsystems reside on the same host, acting as two different locations (a non-data-sharing group), each DB2 subsystem must have a unique port. In this case, only one DB2 subsystem can use the DRDA well-known port number.

Service name
Another way to refer to a port number. A network administrator can assign a service name for a remote location instead of using the port number.

The domain name (IP address) and service name (port number) uniquely identify a DB2 subsystem in the TCP/IP network. The domain name and the service name of the database server **must** be defined in the communications database (CDB) so that a DB2 subsystem can connect to a remote location. The domain name and service name **must** be defined to the TCP/IP host so that a DB2 subsystem can accept connections from remote locations. When DDF is started, the DB2 subsystem binds itself to the port.

The following topics provide additional information:
- "Enabling TCP/IP communication"
- "Step 1: Prepare the Language Environment runtime library" on page 497
- "Step 2: Enable DDF for UNIX System Services" on page 497
- "Step 3: Define the DB2 subsystem to TCP/IP" on page 498
- "Step 4: Populate the communications database" on page 501
- "Step 5: Start TCP/IP support" on page 504
- "Step 6: Tuning TCP/IP" on page 506
- "Step 7: Specify security requirements" on page 506

Enabling TCP/IP communication

These steps enable TCP/IP communication between DRDA partners and DB2. You do not have to do the steps in any particular order, but steps 1, 2, and 3 should be completed prior to the other steps.
- "Step 1: Prepare the Language Environment runtime library" on page 497.
- "Step 2: Enable DDF for UNIX System Services" on page 497.
- "Step 3: Define the DB2 subsystem to TCP/IP" on page 498.
- "Step 4: Populate the communications database" on page 501.
- "Step 5: Start TCP/IP support" on page 504.
- "Step 6: Tuning TCP/IP" on page 506.

DB2 obtains more information from the TCP/IP stack, therefore requiring more configuration than many other daemons.

If you do not use VTAM to communicate with remote sites, you still must define VTAM to DB2 as described in Chapter 13, "Connecting systems with VTAM," on page 453 because DB2 TCP/IP communications uses NETID and LUNAME to identify units of work.

See *DB2 Data Sharing: Planning and Administration* for information about TCP/IP and data sharing.

DDF enhancements enable TCP/IP communication with DRDA partners with DRDA Level 3 support and earlier. To utilize the new functions, clients must have the updated versions of DB2 Connect or any DRDA requester or server that supports the latest DRDA database protocols. TCP/IP connectivity lets you connect DDF to clients on multiple platforms directly.

Attention: TCP/IP is the recommended communication protocol when accessing remote systems.

TCP/IP limitations

TCP/IP does not support DB2 private protocol connections, but you can use SNA for DB2 private protocol connections while using TCP/IP for DRDA connections.

TCP/IP does not have built-in security features that SNA has, such as SNA partner LU verification. Because IP addresses are not as reliable as LU names, DDF support for TCP/IP differs from support for SNA in these ways:

- There is no support for inbound name translation.
- A DB2 subsystem parameter defines the minimum security requirements for all TCP/IP clients because inbound security requirements cannot be established on individual clients. See the description of the TCP/IP ALREADY VERIFIED field on installation panel DSNTIP5 ("Distributed data facility panel 2: DSNTIP5" on page 225).
- You cannot use the CDB for *come from* checking of TCP/IP clients.

Initializing a TCP stack for use with a VIPA

When initializing a TCP stack for use with a virtual IP address (VIPA), the PROFILE.TCPIP configuration should contain the HOME list with the VIPA followed by the PRIMARYINTERFACE statement pointing at the appropriate VIPA to be used by DB2. It is not recommended to change the stack's HOME list or the PRIMARYINTERFACE while DDF is started. The PRIMARYINTERFACE and the stack's HOME list may be changed (with the VARY OBEY command) while the stack is running without recycling the stack. If a new HOME list is specified with the VARY OBEY file, the PRIMARYINTERFACE should also be specified.

Using two-phase commit

DB2 supports two types of 2-phase commit for TCP/IP clients:

- The DRDA client coordinates the 2-phase commit. If a failure occurs during the commit process, DB2 might need to resynchronize with the DRDA client.
- The DRDA client gives responsibility for the resynchronization to DB2. The client sends DB2 a list of server LOCATION names, domain names, and resync IP addresses that are part of the client's unit of work. DB2 Connect V5 utilizes this support by allowing DB2 to act as its Transaction Manager Database (TM_DATABASE), thereby eliminating the need to have a local database to manage the 2-phase commit process. If a failure occurs during the 2-phase commit process, DB2 might need to resynchronize with one or more of the server locations sent by the client.

DB2 uses the port specified on the RESYNC PORT field of installation panel DSNTIP5 for 2-phase commit resynchronization. DB2 begins resynchronization using the partner's IP address and LOCATION name, and the RESYNC PORT obtained at the time of initial connection. If the partner's IP address changed, the resynchronization fails. For example, if the partner was DB2 UDB for z/OS, the IP address can change when the automatic restart manager (ARM) restarts a data sharing member on a different CPC. The IP address can also change when an z/OS adapter fails and virtual IP addresses were not used.

If the IP address fails, DB2 uses the partner's domain name to determine the IP address for resynchronization. DRDA requesters receive the port number and domain name to be used for 2-phase commit resynchronization from the server during DRDA connect processing.

No CDB definition is required to do resynchronization.

Multiple DB2 subsystems on a single z/OS image

It is no longer a recommended configuration to run multiple TCP/IP stacks with DB2. TCP scalability and reliability are such that running two stacks may cause problems due to increased storage and CPU consumption than running one stack, with few or no expected benefits. These configuration improvements are especially helpful when there are multiple DB2 subsystems installed within a data sharing group on the single z/OS image, but these improvements can be useful in a non-data sharing environment.

Although multiple TCP/IP stacks are not recommended for DB2, there may be reasons why multiple stacks may be appropriate for your installation. A possible reason for using multiple TCP/IP stacks could be to separate internet and intranet traffic. The firewall on the stack handling external traffic can be configured only to forward very selected traffic internally, whereas the internal stack does whatever is indicated with internal traffic, including forwarding to the "external" stack. Second, a denial-of-service attack on the stack handling external traffic will not affect traffic on the stack handling internal traffic, so there is an element of resistance to attack with two stacks.

Multiple DB2 subsystems with multiple TCP/IP stacks

z/OS UNIX System Services provides a way to configure DB2 to use a single transport stack. This allows each DB2 subsystem to be restricted to a single stack. Each would be reachable via any IP address owned by the selected stack, but not via the other stack(s). Adding BPXTCAFF as a new job step for the *ssnm*DIST procedure allows DB2 to bind to a single stack when there are at least two stacks running on a single z/OS image. You must specify TIME=1440 because the SYST parameter is disabled when a second job step is added to the *ssnm*DIST procedure.

Multiple DB2 subsystems with one TCP/IP stack

The TCP Server Bind Control allows multiple DB2 subsystems to use the same port number within the same z/OS image. This allows multiple DB2 subsystems that bind to any IP address (also known as INADDR_ANY in sockets API terms) and the same port number on the same stack to be bound to separate IP addresses. For example, consider two DB2 subsystems in a data sharing group where both have the same LOCATION and same DRDA PORT. A new 'BIND ipaddr' parameter is added to the PORT statement. When DB2 (identified by job name *ssnm*DIST) issues a bind to the port number in the PORT statement and to INADDR_ANY, the bind is restricted to the IP address specified on the PORT statement for that DB2 subsystem. This allows two DB2s to share the same stack by limiting them to different single IP addresses (virtual IP addresses are recommended). This allows them to be reached by any physical connectivity that can get to the stack. For more information about using the PORT statement, see *OS/390 V2R10.0 IBM CS IP Configuration Reference*.

Step 1: Prepare the Language Environment runtime library

Because DDF uses some functions in the Language Environment library, DDF needs access to the runtime library. The standard way to handle this is to include the Language Environment library in a STEPLIB concatenation for the DDF JCL procedure. The Language Environment library must be APF authorized to be added to the DDF JCL procedure. The DB2 installation automatically adds the library to the DDF STEPLIB concatenation.

The alternative method is to concatenate the Language Environment library in the z/OS link list, which does not require the library to be APF authorized. If you choose this alternative method, remove the Language Environment library concatenation from the DDF JCL procedure.

Step 2: Enable DDF for UNIX System Services

DDF uses the asynchronous I/O assembler callable interface of UNIX System
Services to perform TCP/IP services. Any address space that has to use UNIX
System Services must have a z/OS user ID that is defined with an OMVS segment.
The address space can also have a z/OS group name.

Use either of the following z/OS Security Server (RACF) commands to assign an
OMVS segment to a z/OS user ID:

• ADDUSER *ddfuid* OMVS(UID(*nnn*))...
• ALTUSER *ddfuid* OMVS(UID(*nnn*))...

where *ddfuid* is the z/OS user ID and *nnn* is any valid, unique identifier. If you set
nnn to 0, the process has UNIX System Services superuser authorization.

During initialization, DDF calls UNIX System Services to raise the maximum
number of open files per process to 65535. UNIX System Services considers an
open TCP/IP socket an open file. This interface call requires that the process be a
UNIX System Services superuser (that is, UID(0)) to raise the current limit. DDF
executes as an authorized program and is protected against any unauthorized use
# of this privilege.

DDF obtains the current limit from the value of the MAXFILEPROC parameter.
The MAXFILEPROC parameter is in the active z/OS UNIX System Services
member (BPXPRMnnn) of the z/OS PARMLIB. Assuming the current value of
MAXFILEPROC is 65535 or greater, the UNIX System Services call that DDF makes
to set the limit to 65535 succeeds regardless of the value that you give to *nnn*.

Because setting the MAXFILEPROC parameter within the active BPXPRMnn z/OS
PARMLIB member to 65535 affects the entire system, any UNIX System Services
process can open 65535 files or sockets concurrently. If you do not want to set the
system-wide MAXFILEPROC parameter to 65535 and give the z/OS user ID
superuser authority at the same time, you can explicitly authorize the z/OS user
ID to raise the limit to 65535 itself by using one of the following z/OS Security
Server (RACF) commands:

• ADDUSER *ddfuid* OMVS(UID(*nnn*) FILEPROCMAX(65535))...
• ALTUSER *ddfuid* OMVS(UID(*nnn*) FILEPROCMAX(65535))...

where *ddfuid* is the z/OS user ID and *nnn* is any valid, unique identifier.

If you also want to assign a z/OS group name to the address space, assign an OMVS segment to the z/OS group name by using one of the following RACF commands:

- ADDGROUP *ddfgnm* OMVS(GID(*nnn*))...
- ALTGROUP *ddfgnm* OMVS(GID(*nnn*))...

where *ddfgnm* is the z/OS group name and *nnn* is any valid, unique identifier.

The standard way to assign a z/OS userid and a z/OS group name to a started address space is to use the z/OS Security Server (RACF) STARTED resource class. This method enables you to dynamically assign a z/OS user ID by using commands instead of requiring an IPL to have the assignment take effect.

If you actively administer the STARTED resource class, you can issue the following RACF commands to assign a z/OS user ID to the DDF started procedure:

- RDEFINE STARTED (*V81ADIST.**) STDATA(USER(*ddfuid*)) ...
- RALTER STARTED (*V81ADIST.**) STDATA(USER(*ddfuid*)) ...

where *V81ADIST.** is the DDF started procedure and *ddfuid* is the z/OS user ID.

You can issue the following RACF commands to assign both a z/OS user ID and a z/OS group name to the DDF started procedure. DDF requires only a z/OS user ID; a z/OS group name is optional.

- RDEFINE STARTED (*V81ADIST.**) STDATA(USER(*ddfuid*) GROUP(*ddfgnm*)) ...
- RALTER STARTED (*V81ADIST.**) STDATA(USER(*ddfuid*) GROUP(*ddfgnm*)) ...

where *V81ADIST.** is the DDF started procedure, *ddfuid* is the z/OS user ID, and *ddfgnm* is the z/OS group name.

Requirement: The profile name that you specify in the RACF command must be in the generic format; that is, the profile name must end with a period followed by an asterisk (.*). After one of the commands is executed, issue the following RACF command so that the changed profile takes effect:

- SETROPTS RACLIST(STARTED) REFRESH

For more information about using the RACF STARTED resource class, see *z/OS Security Server RACF Security Administrator's Guide*.

The alternative method to assign a z/OS user ID and a z/OS group name to a started address space is to change the RACF started procedures table, ICHRIN03. For more information about this table, see *z/OS Security Server RACF System Programmer's Guide*.

Step 3: Define the DB2 subsystem to TCP/IP

The DB2 subsystem uses different TCP/IP ports to do different tasks:

- As a requester, DB2 uses an ephemeral port. You do not need to specify this port.
- As a server processing TCP/IP connection requests for DRDA SQL applications, DB2 uses a server port or the well-known port, 446, which is used for relational database communications.
- A server resynchronization port is used for processing 2-phase commit resynchronization requests.

This requires some planning because the port number is used to pass the network requests to the right DB2 subsystem. Each location must have a unique port number. *DB2 Data Sharing: Planning and Administration* has information for assigning port numbers for systems that have enabled data sharing.

Figure 99 shows some typical z/OS system configurations that demonstrate some typical configurations.

- In SYSTEM1, there is only one DB2 subsystem, so the DRDA well-known port (446) can be assigned to DB2. In the example, port number 5020 is assigned for 2-phase commit resynchronization.
- In SYSTEM2, there are two DB2 subsystems, making it impossible to assign the port numbers 446 and 5020 to both DB2 subsystems, because TCP/IP can only support one server at each port number. The problem is resolved by assigning the 446 and 5020 port numbers to DB2C, and port numbers 5021 and 5022 to DB2D.

Be sure to consider the impact of future system consolidations. If SYSTEM1 and SYSTEM2 are consolidated so that DB2A, DB2C, and DB2D run on a single z/OS system, you must take special precautions because DB2A and DB2C have the same TCP/IP port numbers. You can resolve this by changing the port numbers of either DB2A or DB2C to eliminate the duplicate port numbers.

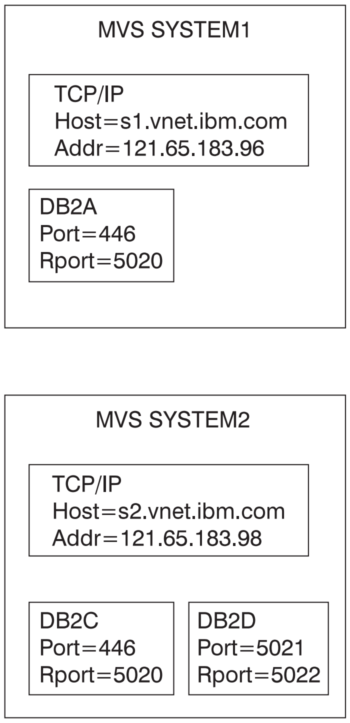


Figure 99. Typical z/OS configurations

Customize the TCP/IP data sets or files

Follow these steps to customize your TCP/IP data sets or files. **UNIX System Services should already be installed.** See the DB2 Program Directory for required maintenance levels. For details on customizing your TCP/IP data sets see *z/OS UNIX System Services Planning*.

1. Find the TCPIP.TCPIP.DATA data set.

This data set defines the high level qualifier (*hlq*) which is added to the beginning of other data set names used by TCP/IP.

2. Find the *hlq*.TCPPARMS(PROFILE) data set.

   This data set contains the PORT statement used to make DRDA and resync port reservations. The following example from Figure 99 on page 499 shows a sample *hlq*.TCPPARMS(PROFILE) entry for SYSTEM2:

```
PORT  446  TCP  DB2CDIST         ; DRDA SQL port for DB2C
PORT 5020  TCP  DB2CDIST         ; Resync port for DB2C

PORT 5021  TCP  DB2DDIST         ; DRDA SQL port for DB2D
PORT 5022  TCP  DB2DDIST         ; Resync port for DB2D
```

   This example assumes that *DB2CDIST* and *DB2DDIST* are the DB2 started procedure names.

3. Define the TCP/IP host names that DB2 needs to know.

   The local host name **must** be defined before DDF is started. All domain names referenced in the table SYSIBM.IPNAMES **must** be defined. You define the host names by configuring the *hlq*.HOSTS.LOCAL data set, the */etc/hosts* file in the hierarchical file system (HFS), or the domain name server (DNS).

   After the *hlq*.HOSTS.LOCAL data set is configured, you have to execute the utility MAKESITE. This utility generates the *hlq*.HOSTS ADDRINFO and the *hlq*.HOSTS.SITEINFO data sets that are used to translate between domain names and IP addresses. MAKESITE generates under the userid that issued the MAKESITE. Therefore, those files have to be moved to the high level qualifier that represents TCP/IP. The host name for the local DB2 subsystem must be defined in at least one of these places.

   If domain names are present in the CDB (in field IPADDR of table SYSIBM.IPNAMES), they must be defined in the z/OS data sets, the HFS or the DNS.

   Recommendation: To support a Parallel Sysplex environment, use Dynamic Virtual IP Addresses (VIPA). To configure a DB2 subsystem to perform Dynamic VIPA routing, specify the DB2's group Dynamic VIPA on the TCP/IP PORT statement for the DRDA PORT number. This Dynamic VIPA must be the same for all DB2 members in the data sharing group. All clients must use the Dynamic VIPA to route requests to the DB2 group. To DB2 member-specific Dynamic VIPA is specified on the TCP/IP RESYNC PORT number in the TCP/IP profile data set for each DB2 member of the sysplex.

   The following example shows a sample *hlq*.TCPPARMS(PROFILE) entry for SYSTEM2:

```
PORT 446  TCP  DB2DIST  BIND  db2_sysplex_VIPA       ;DRDA SQL port for DB2
PORT 5001 TCP  DB2DIST  BIND  member_specific_VIPA   ;RESYNC port for DB2
```

   More information about using Dynamic VIPA is available in *DB2 Data Sharing: Planning and Administration*

4. Define the TCP/IP service names that DB2 needs to know.

   Configure the *hlq*.ETC.SERVICES data set or the */etc/services* file in the HFS. If service names are present in the CDB (in field PORT of table SYSIBM.LOCATIONS), they must be defined in the z/OS data set or the HFS.

   The following example shows a sample *hlq*.ETC.SERVICES entry:

```
DRDA       446/tcp            ; DRDA databases
```

For more detailed information on these steps see *IBM TCP/IP for MVS: Customization & Administration Guide*.

Modify the change log inventory job

To use TCP/IP, the DDF statement of the change log inventory job (DSNJU003) must specify values for the parameters PORT and RESPORT.

The parameter PORT is the TCP/IP port number used by DDF to accept incoming DRDA connection requests. The parameter RESPORT is the TCP/IP port number used by DDF to accept incoming DRDA 2-phase commit resynchronization requests. The values for each of these parameters must be a decimal number between 0 and 65534, where zero indicates that DDF's TCP/IP support is being deactivated. The non-zero value for PORT must not be the same as the non-zero value for RESPORT.

For data sharing, all the members of the DB2 data sharing group must have the same value for PORT. RESPORT must be uniquely assigned to each DB2 member so that no two DB2 members use the same TCP/IP port for 2-phase commit resynchronization. The parameters PORT and RESPORT can be changed on any DB2 member by running the utility change log inventory. After running the utility, you must stop and then restart DDF. Since PORT is the same for all members of the DB2 group, this process has to be repeated on every member of the group when PORT is changed.

If you use Dynamic VIPA to support a Parallel Sysplex environment, specify the DB2's group Dynamic VIPA on the TCP/IP PORT statement for the DRDA PORT number. More information about using Dynamic VIPA in a Parallel Sysplex environment is available in *DB2 Data Sharing: Planning and Administration*.

You can define an alias location for all or selected members of a data sharing group by using the ALIAS parameter. See *DB2 Data Sharing: Planning and Administration* for more information.

Remember, a zero value for either PORT or RESPORT is the same as **deactivating** DB2's TCP/IP support.

Step 4: Populate the communications database

The information under this heading, up to "Step 5: Start TCP/IP support" on page 504 is General-use Programming Interface, as defined in "Notices" on page 545.

If you plan to use DB2 only as a server, you do not need to populate the CDB. For example, Spiffy's USIBMSTODB21 subsystem works as a server for many requesters. It is not necessary for Spiffy to register those requesters in DB2's CDB.

However, if you intend to request data, you need to enter port numbers or service names in field PORT of table SYSIBM.LOCATIONS, and IP addresses or domain names in field IPADDR of table SYSIBM.IPNAMES. The LINKNAME in table SYSIBM.LOCATIONS is used to search tables SYSIBM.IPNAMES and SYSIBM.LUNAMES (see Chapter 13, "Connecting systems with VTAM," on page 453). If RACF PassTickets are used, the LINKNAME must match the LUNAME of the remote site. Part 3 (Volume 1) of *DB2 Administration Guide* discusses the requirements for the other tables.

After you populate these tables, you can write queries that access data at a remote system. For instructions on sending SQL statements to other systems, see *DB2*

Application Programming and SQL Guide. For instructions about granting privileges to users on remote DB2 subsystems, see Part 3 (Volume 1) of *DB2 Administration Guide*.

SYSIBM.LOCATIONS table

The table LOCATIONS is used to determine the port number or service name used to connect to the remote location. The column LINKNAME maps to the corresponding row in table IPNAMES.

LOCATIONS has the following columns relating to TCP/IP:

DBALIAS VARCHAR(128) NOT NULL	The name that is associated with the server. This name is used to access a remote database server. If DBALIAS is blank, the location name is used to access the remote database server. This column does not change database object names that are sent to the remote site using a location qualifier. Use the DBALIAS column to access data at two or more different remote locations when those remote locations have the same name.
LOCATION CHAR(16)	The unique network location name, or DRDA RDBNAM, that is assigned to a remote or local system. You must provide location names for any systems from which you request data. This column is the primary key for this table.
LINKNAME CHAR(8)	Identifies the TCP/IP attributes that are associated with this location. For each specified LINKNAME, you must have a row in SYSIBM.IPNAMES whose LINKNAME matches the value that is specified in this column. Because this table is used for outbound requests, you must provide a LINKNAME or your requests fail. Do not enter blanks in this column.
PORT CHAR(32)	If blank, the default port, 446, is used for TCP/IP communications. Otherwise, the value can be either of the following values: • The port number of the remote database server. The number must be one to five characters and left-justified. • A TCP/IP service name. The service name is converted to a TCP/IP port number with the *getservbyname* socket call.

Spiffy’s USIBMSTODB21 location wants a LOCATIONS table that looks like Table 120. The location USIBMSTODB21 uses the default DRDA PORT, 446.

Table 120. Spiffy’s LOCATIONS table

LOCATION	LINKNAME	PORT
USIBMSTODB21	LUDB21	
USIBMSTODB22	LUDB22	
USIBMSTOSQL1	LUSQLDS	1234

Table 120. Spiffy's LOCATIONS table (continued)

LOCATION	LINKNAME	PORT
USIBMSTOSQL2	LUSQLDS	DRDA

For example, add the second row with this statement:

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, LINKNAME)
  VALUES ('USIBMSTODB22','LUDB22');
```

Since no port number is specified, location USIBMSTODB22 uses the default DRDA port number, 446.

A row for the local location: You do not need a row for the local DB2 in the IPNAMES and LOCATIONS tables. For example, Spiffy's USIBMSTODB21 subsystem does not require a row that shows its own LINKNAME and location name.

SYSIBM.IPLIST table

IPLIST contains list of multiple IP addresses that are specified for a given location. IPLIST has the following columns:

LINKNAME CHAR(8) NOT NULL	This column is associated with the value of the LINKNAME column in SYSIBM.LOCATIONS and SYSIBM.IPNAMES. The values of the other columns in the SYSIBM.IPNAMES row apply to the server that is identified by the LINKNAME column in this row.
IPADDR VARCHAR(256) NOT NULL	This column contains the IP address or domain name of a remote TCP/IP host of the server. If using WLM domain name server workload balancing, this column must contain the member-specific domain name. If you use dynamic VIPA workload balancing, this column must contain the member-specific dynamic VIPA address.
IBMREQD CHAR(1) NOT NULL WITH DEFAULT 'N'	This columns indicates whether the row came from the basic machine-readable material (MRM) tape: N=no, Y=yes

SYSIBM.IPNAMES table

IPNAMES defines the outbound security and host names used to connect to other systems using TCP/IP. IPNAMES has the following columns:

LINKNAME CHAR(8)	This value matches that specified in the LINKNAME column of the associated row in SYSIBM.LOCATIONS.
SECURITY_OUT CHAR(1)	Defines the security option that is used when local DB2 SQL applications connect to any remote server associated with this TCP/IP host. The default, A, means that outgoing connection requests contain an authorization ID without a password.

USERNAMES CHAR(1)	This column is used for outbound requests to control translations of authorization IDs. The values 'O' or 'B' are valid for TCP/IP connections.
IPADDR VARCHAR(254)	This column contains the IP address or domain name of a remote TCP/IP host. • An IP address must be 1 to 15 characters and left justified. An example of an IP address is *9.112.46.111*. • A domain name is converted to an IP address by the domain name server. An example of a domain name is *stlmvs1.stl.ibm.com*. The *gethostbyname* socket call is used to resolve the domain name.

SYSIBM.USERNAMES table

USERNAMES contains information needed for outbound translation only.
Reminder: Inbound ID translation and *come from* checking are not done for TCP/IP requesters.

TYPE CHAR(1)	Whether the row is for outbound translation. The value 'O' is valid for TCP/IP connections.
AUTHID CHAR(8)	Authorization ID to translate. If blank, it applies to all authorization IDs.
LINKNAME CHAR(8)	Identifies the TCP/IP network location associated with the row. A blank indicates it applies to all TCP/IP partners. For nonblank values, this value must match the LINKNAME value in SYSIBM.IPNAMES.
NEWAUTHID CHAR(8)	The translated value of AUTHID.
PASSWORD CHAR(8)	The password to accompany an outbound request. This column is ignored if RACF PassTickets, or already verified USERIDs are used.

Step 5: Start TCP/IP support

Before you start DDF, UNIX System Services and TCP/IP must be started and the local host name must be defined in */etc/hosts*, *TCPIP.HOSTS.LOCAL*, or the domain name server (DNS).

DDF executes the following steps when it is started:

1. Binds a socket to the SQL port.
2. Issues a gethostid to obtain the TCP/IP stack's primary interface IP address. If an address was specified on the PORT statement, uses that address instead of issuing a gethostid.

   The IP address can be determined by issuing a ″netstat home″ command. In the output, the primary interface is marked with a P in the Flg column. For example:

```
EZZ2350I MVS TCP/IP NETSTAT CS V2R8       TCPIP NAME: TCPIP           19:02:54
EZZ2700I Home address list:
EZZ2701I Address         Link            Flg
EZZ2702I -------         ----            ---
EZZ2703I 10.2.1.21       TR1
EZZ2703I 10.2.2.21       TR2
```

```
EZZ2703I 10.1.1.22        CTC02L
EZZ2703I 10.1.1.18        MPC00L
EZZ2703I 10.1.1.2         ENDER2L
EZZ2703I 10.2.10.21       FENET1
EZZ2703I 192.2.30.21      ATMLEC1
EZZ2703I 10.1.1.253       C1TOC2
EZZ2703I 10.1.100.21      LVIPA01          P
EZZ2703I 127.0.0.1        LOOPBACK
```

3. If using Dynamic VIPA:
   a. Binds a socket to the resync port to obtain the member-specific address.
   b. Binds a socket to the SQL port and the member-specific address obtained above.
   c. Listens for incoming SQL requests on both sockets bound to the SQL port.

   If not using Dynamic VIPA, listens for incoming SQL requests on the socket bound to the SQL port.
4. Issues a gethostname to register the HOST to WLM. TCP/IP returns the HOSTNAME found in the stack's TCPIP.DATA file
5. If not using VIPA, issues a gethostbyaddr using the local IP address obtained from the gethostid to obtain the fully qualified domain name. If using VIPA, issues gethostbyaddr using the member-specific and group VIPA addresses to obtain the member-specific and group domains. The domain name is returned to clients to be used to open a connection to DB2 to perform resynchronization after a failure.
6. Issues set_sock_opt for all inbound and outbound connections for the following options:
   - SO_KEEPALIVE to enable keepalive processing.
   - SO_REUSEADDR to allow the use of the same port.
   - SO_SNDBUF to set the send buffer size if less than 64K.
   - SO_EIOIFNEWTP to recycle DDF dynamically when a new stack becomes active.
   - TCP_KEEPALIVE to override the TCP/IP keepalive value with the one specified during installation or migration on panel DSNTIP5.

To support data sharing workload balancing and commit resynchronization, DDF requires the fully qualified host name to be established during startup processing. During startup, DDF issues a gethostid and gethostbyaddr socket request. The gethostid service obtains the local IP address and then a gethostbyaddr service is issued to get the fully qualified host name. The gethostbyaddr service attempts to resolve the hostname through a nameserver.

Important: When using Dynamic VIPA, the group and member-specific addresses must be registered with the DNS.

If the nameserver is not present, or does not have an answer, then the local host tables are used. The local host tables search order for the resolver as follows:
1. /etc/resolv.conf file
2. jobname.TCPIP.DATA
3. SYS1.TCPPARMS(TCPDATA)
4. TCP*hlq*.TCPIP.DATA

More information about using Dynamic VIPA is available in *DB2 Data Sharing: Planning and Administration*

Step 6: Tuning TCP/IP

This is an optional step, but the recommendations here can protect DB2 from TCP/IP outages.

The recommendations are:

- Use the IDLE THREAD TIMEOUT installation option on the Distributed Data Facility panel, DSNTIPR, to limit the time an idle thread can hold locks.
- Specify a small value, five minutes or less, for the TCP/IP keep_alive timer. If the network fails between the server's reply and the next client request, TCP/IP waits until the keep_alive timer expires, then notifies the DB2 subsystem of the failure.

  The server thread hangs while the timer is running. Because the timer default is 2 hours, threads can hang for up to 2 hours if you use the default. The hung thread can cause unpredictable results, depending on what resources it has locked.

If you are connecting to a location whose LINKNAME is associated with a row in the table SYSIBM.IPNAMES and it has a domain name in the IPADDR field, *gethostbyname* can return a list of IP addresses associated with the LINKNAME. When trying to connect to this location, DB2 will try each of these IP addresses in a round-robin fashion (starting with the first address) until the connection is successful, or the attempt to connect to each IP address has timed out.

Step 7: Specify security requirements

You must specify security requirements. See Part 3 (Volume 1) of *DB2 Administration Guide* for more information about establishing RACF protection for DB2.

Part 4. Appendixes

Appendix A. Character conversion

This appendix describes how DB2 handles character conversion for distributed data. The following topics are discussed in this appendix:

- "Character conversion concepts"
- "Specifying a system-coded character set identifier" on page 510
- "How an entry in SYSIBM.SYSSTRINGS works" on page 522
- "Uppercase and lowercase conversion of Unicode data" on page 524

Character conversion concepts

In different database management systems (DBMSs), character data can be represented by different encoding schemes. Within an encoding scheme, there are multiple coded character set identifiers (CCSIDs). EBCDIC, ASCII, and Unicode are ways of encoding character data. Character data that is transmitted from one DBMS to another might need to be converted to a different coded character set.

The Unicode character encoding standard is a character encoding scheme that includes characters from almost all living languages of the world. DB2 supports two implementations of the Unicode encoding scheme: UTF-8 (a mixed-byte form) and UTF-16 (a double-byte form). For more information about Unicode, see "Unicode support" on page 510.

All character data has a CCSID. Character conversion is described in terms of CCSIDs of the source and of the target. When you install DB2, you must specify a CCSID for DB2 character data in either of the following situations:

- You specify AUTO or COMMAND for the DDF STARTUP OPTION field on panel DSNTIPR.
- Your system will have any ASCII data, Unicode data, EBCDIC mixed character data, or EBCDIC graphic data. In this case, you must specify YES in the MIXED DATA field of panel DSNTIPF, and the CCSID that you specify is the mixed data CCSID for the encoding scheme.

The CCSID that you specify depends on the national language that you use. "Specifying a system-coded character set identifier" on page 510 lists the choices.

DB2 performs most character conversion automatically, based on system CCSIDs, when data is sent to DB2 or when data is stored in DB2. If character conversion must occur, DB2 uses the following methods:

1. DB2 searches the catalog table SYSIBM.SYSSTRINGS.
2. DB2 uses z/OS Unicode Conversion Services. For more information about z/OS Unicode Conversion Services, see *z/OS Support for Unicode: Using Conversion Services*.

If DB2 or z/OS Unicode Conversion Services does not provide a conversion for a certain combination of source and target CCSIDs, you receive an error message. If the conversion is incorrect, you might get an error message or unexpected output. To correct the problem, you need to understand the rules for assigning source and target CCSIDs in SQL operations. Chapter 3 of *DB2 SQL Reference* explains those rules.

Specifying a system-coded character set identifier

To support character conversion, the IBM Distributed Relational Database Architecture uses CCSIDs to label the various character representation schemes.

The CCSID is a two-byte binary number that uniquely identifies one or more pairs of character sets and code pages. The coded character set defines how bit configurations are mapped for character data. For a complete description of all IBM-registered CCSIDs and conversion tables, see *Character Data Representation Architecture Reference and Registry*. For general information about character conversion, see *Character Data Representation Architecture Overview*.

The CCSID of character strings at your site is determined by the CCSID that you specify on installation panel DSNTIPF. **If this CCSID is not correct, character conversion produces incorrect results.** The correct CCSID is the number that identifies the coded character set that is supported by your site's I/O devices, local applications such as IMS and QMF, and remote applications such as CICS Transaction Server.

Unicode support

Unicode is a universal encoding scheme for written characters and text that enables the exchange of data internationally. Unicode provides a character set standard that can be used all over the world. Unicode uses a 16-bit encoding scheme that provides code points for more than 65 000 characters. An extension called UTF-16 allows for encoding as many as a million more characters. Unicode provides the ability to encode all characters used for the written languages of the world. Unicode treats alphabetic characters, ideographic characters, and symbols equivalently because it specifies a numeric value and a name for each of its characters. Unicode includes punctuation marks, mathematical symbols, technical symbols, geometric shapes, and dingbats.

DB2 provides the following Unicode encoding forms:

- UTF-8: Unicode Transformation Format, 8-bit encoding form that is designed for ease of use with existing ASCII-based systems.
- UTF-16: Unicode Transformation Format, 16-bit encoding form that is designed to provide code values for over a million characters and a superset of UCS-2. UCS-2 is Universal Character Set and is coded in 2 octets, which means that characters are represented in 16 bits per character.

Unicode CCSIDs: The Unicode CCSID field of panel DSNTIPF is pre-filled with 1208. DB2 chooses the CCSIDs for double-byte and single-byte values (1200 for DBCS and 367 for SBCS). CCSID 1200 corresponds to UTF-16 and CCSID 367 is for 7-bit ASCII.

Customizing support for Unicode

You must customize z/OS Unicode support in order to use Unicode data in DB2. Take the following steps to customize support for Unicode:

1. Ensure that the conversion environment is active. The steps for this process can be found in *z/OS Support for Unicode: Using Conversion Services*. DB2 can use the conversion services of z/OS support for Unicode only when the conversion environment is active. The infrastructure provides tools to create a conversion image. When the image is loaded into a common data space, the conversion environment is activated, and the conversion services are ready to be used by DB2.

| 2. Customize the job card. The jobs in *hlq*.SCUNJCL, as shipped by IBM, have placeholders for values on the JOB statement, such as the following example:

```
//$JOBPREF$$JOBNAME$ JOB ($ACCOUNT$), '$USER$',
//    NOTIFY=$NOTIFY$,MSGCLASS=$MC$,MSGLEVEL=$ML$,
//    TIME=$TI$,CLASS=$CL$,REGION=$REGION0M$
```

Use the REXX EXEC CUNRUCST in *hlq*.SCUNREXX to customize these values. When you run the REXX EXEC CUNRUALL, the values that you specify are supplied on all of the JCL images. You can choose to customize your system by displaying Japanese messages or displaying English messages with modified date and time formats. You can set up the z/OS Message Service to specify how you want messages to be displayed.

3. Set up the conversion image. The following example is the sample JCL member in *hlq*.SCUNJCL (CUNJIUTL):

```
//CUNMIUTL EXEC PGM=CUNMIUTL
//SYSPRINT DD   SYSOUT=*
//TABIN    DD   DISP=SHR.DSN=hlq.SCUNTBL
//SYSIMG   DD   DSN=hlq.IMAGES(CUNIMG00),DISP=SHR
//SYSIN    DD   *
   /*******************************************
    * INPUT STATEMENTS FOR THE IMAGE GENERATOR *
    *******************************************/
      CASE NORMAL;          /* ENABLE TOUPPER AND TOLOWER */
      CONVERSION 1047,850;  /* EBCDIC -> ASCII */
      CONVERSION 850,1047;  /* ASCII -> EBCDIC */
/*
```

In the preceding example, the two CONVERSION statements provide conversion between the EBCDIC code page 1047 and the ASCII code page 850 in both directions.The DD names that are passed to CUNMIUTL are described as follows:

SYSPRINT
A listing that shows the processed setups and error messages, if applicable.

TABIN
Conversion tables for character conversion and case conversion. They are supplied by IBM; in this example, they are in data set *hlq*.SCUNTBL. The image transforms the conversion tables into an internal format and stores them in the conversion image.

SYSIMG
Output is a single image of the entire conversion environment. The conversion image is built according to the specification in the SYSIN DD name. The conversion image resides in either a sequential data set or a member of a partitioned data set with a fixed-block 80-byte format. In this example, the image resides in the partitioned data set member *hlq*.IMAGES(CUNIMG00).

SYSIN
Two types of statements are recognized in this DD statement: case conversion, which is identified by the CASE control statement, and character conversion, which is identified by the CONVERSION control statement.

CASE control statement:
Case conversion is defined as converting Unicode characters (for example, UTF-8) to their uppercase equivalent or their lowercase equivalent. In the preceding example, CASE NORMAL is specified, which means basic case conversion is

provided. This basic case conversion is based on the UnicodeData.txt file that is provided by the Unicode consortium. It does not include special casing as described in the SpecialCasing.txt file that is provided by the Unicode consortium. Special casing typically includes characters that have significant differences in the case-based appearance. For example, the German ″hard S″, which appears as a flat B, appears as ″SS″ in uppercase German text. Because DB2 does not use the case conversion service, you do not need to specify a conversion.

CONVERSION control statement:

Character conversion is also referred to as conversion between specified CCSIDs. An application such as DB2 invokes the CUNLCNV function to convert characters between the specified code pages. You must identify the conversions that are possible on the CONVERSION control statement.

Important: Specify CONVERSION statements for DB2 as follows:

```
CONVERSION xxx,yyy,ER;
CONVERSION yyy,xxx,ER;
```

Many code page conversions are possible. They are documented in *z/OS Support for Unicode: Using Conversion Services*. However, when identifying the conversion that DB2 is to use, you need be concerned only with conversion for the national languages that you use and with conversion from these code pages to and from all Unicode CCSIDs.

Example: If you use an EBCDIC CCSID of 37 and an ASCII CCSID of 819, you need to use the following conversions:

```
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;

CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

Multiple conversion tables might be available for converting one CCSID to another. A technique search order can be used to specify which table should be used. The technique search order consists of up to eight technique characters. If you specify more than one technique character, the image generator tries to find a matching table for the leftmost technique character in the sequence of the technique-search-order. If one is not found, the search continues with the second one, and so on. Especially for mixed conversion, use more than one technique character because one of the subconversions might exist only in round-trip mode, and one might exist only in an enforced subset. In this case, a technique search order of ′RE′ or ′ER′

would be required. Technique search order is optional. If you do not specify a technique search order, RECLM is used.

Language products such as Enterprise Cobol might use the RECLM technique search order, while DB2 uses the ER technique search order. Therefore, you might also need to add the RECLM conversions, such as these:

```
CONVERSION 1047,850,RECLM;  /* EBCDIC -> ASCII */
CONVERSION 850,1047,RECLM;  /* ASCII -> EBCDIC */
```

The important technique characters for DB2 are E (enforced subset) and R (round-trip). Enforced subset conversions map only those characters from one CCSID to another that have a corresponding character in the second CCSID. All other characters are replaced by a substitution character. Round-trip conversions between two CCSIDs assure that all characters making the 'round trip' arrive as they were originally, even if the receiving CCSID does not support a given character. Round-trip conversions ensure that code points that are converted from CCSID A to CCSID B, and back to CCSID A are preserved, even if CCSID B is not capable of representing these code points.

After performing these steps, you should now have an updated CUNJIUTL JCL member.

4. Submit the batch job in the CUNJIUTL member. At completion, the batch job writes its output to the SYSPRINT DD (that is, SYSOUT in this example). Expect a return code of zero from the CUNJIUTL program. If you receive anything other than return code zero, refer to the error situations in *z/OS Support for Unicode: Using Conversion Services*. This information helps you correct environmental, syntactical, and semantic errors that might occur.
5. After generating the conversion image, copy it to SYS1.PARMLIB or any other data set in the logical PARMLIB concatenation. In this example, you copy *hlq*.IMAGES(CUNIMG00) to SYS1.PARMLIB(CUNIMG00).
6. Calculate the storage that is needed for a conversion image. When the conversion image is created on disk, you need to determine the amount of virtual storage that the image is to occupy. You specify this number as the number of pages on the REALSTORAGE parameter in the CUNUNI*xx* PARMLIB member that you create in the next step. The REALSTORAGE parameter protects the system from a shortage of main storage caused by loading a conversion image that exceeds the amount of available storage. The minimum value for the REALSTORAGE parameter depends on how the image is activated.
   - If the image is activated during IPL, the needed storage is the size of the image plus one page.
   - If the image is activated using the SET UNI command (PARMLIB member with keyword IMAGE), the needed storage is the size of the currently active image plus the size of the new conversion image.

   If you set up a conversion environment before or if you are not activating a conversion environment during IPL, you must determine the amount of storage that the currently active image occupies. To do this, issue the following command:

   ```
   D UNI,STORAGE
   ```

The system displays the number of active pages.

To determine the storage that the new conversion image occupies, find the CUN1017I message in the SYSPRINT log that was created in the previous step. This message indicates the number of pages that are required for the new conversion image. For example:

```
CUN1017I GENERATED IMAGE SIZE 291 PAGES........
```

As an alternative, specify a REALSTORAGE value of zero, which indicates that unlimited storage is available. In this case, a value of 524 287 pages is used.

7. Create the PARMLIB member CUNUNI*xx* (PARMLIB member for activating a conversion environment). Normally the member is created in SYS1.PARMLIB, but in this case, you create it in another data set in the logical PARMLIB concatenation. This example uses SYS1.PARMLIB.

   The *xx* can be any two alphanumeric characters, or the special characters @, #, or $. This example uses 00. Here is the sample PARMLIB member in SYS1.PARMLIB(CUNUNI00):

   ```
   REALSTORAGE 292;
   IMAGE CUNIMG00;
   ```

   Because CUNMIUTL requires 291 pages, and an additional page is required during IPL, the REALSTORAGE statement indicates that 292 pages of real storage are required.

   The IMAGE parameter indicates that the system searches in SYS1.PARMLIB (or a data set in the logical concatenation) member CUNIMG00 for the conversion image.

   You can create a PARMLIB member to delete a current conversion environment. Refer to *z/OS Support for Unicode: Using Conversion Services* for more information.

8. Take one of the following actions:
   - Edit IEASYSxx.

     This parameter specifies one or more CUNUNI*xx* PARMLIB members that contain the keywords that configure the conversion environment. Each suffix *xx* identifies one CUNUNI*xx* member in the PARMLIB concatenation. If several PARMLIB members are specified, they are concatenated in the specified sequence. The concatenated contents is handled internally as a single member. This means that the lines are numbered consecutively, and error messages about syntax errors refer to the concatenated text. Restrictions for keywords apply for the entire concatenated text.
   - Check parameter MAXCAD in IEASYSxx. It limits the amount of common data spaces in a system. If MAXCAD is specified, consider that z/OS support for Unicode creates up to two common data spaces. More information can be found in *z/OS Support for Unicode: Using Conversion Services*.
9. Initialize the conversion environment with an IPL.
10. After the system is initialized, you can use the DISPLAY UNI system command to show the current z/OS Unicode status or use the SET UNI system command to change the conversion environment. For more information, refer to *z/OS Support for Unicode: Using Conversion Services*.

EBCDIC and ASCII support

If you specify MIXED DATA = NO on panel DSNTIPF, you can use any compatible SBCS CCSID in the EBCDIC CODED CHAR SET and ASCII CODED CHAR SET fields. Table 121 lists a selection of common SBCS CCSIDs that might be used as source or target CCSIDs for EBCDIC or ASCII data. DB2 does not support the storing of data into all of these CCSIDs. That is, not all of the numbers listed in Table 121 are supported as target CCSIDs in conversion. When you choose CCSIDs, you should choose an EBCDIC CCSID and an ASCII CCSID that are listed in the same row in Table 121.

Table 121. Single-byte coded character set identifiers (CCSIDs)

Country or national language	EBCDIC	ASCII PC	ASCII AIX	ASCII Windows
Australia (U.S. English)	37/1140*	437	819	1252/5348*
Austria (German)	273/1141*	850/858*	819	1252/5348*
Belarus (Cyrillic)	1025			1251/5347*
Belgium	500/1148*	850/858*	819	1252/5348*
Bosnia and Herzegovina (Cyrillic)	1025			1251/5347*
Bosnia and Herzegovina (Latin)	870	852	912	1250/5346*
Brazil (U.S. English)	37/1140*	850/858*	819	1252/5348*
Bulgaria (Cyrillic Multilingual)	1025			1251/5347*
Canada (U.S. English)	37/1140*	850/858*	819	1252/5348*
Croatia	870	852	912	1250/5346*
Czech Republic	870	852	912	1250/5346*
Denmark	277/1142*	850/858*	819	1252/5348*
Finland (Swedish)	278/1143*	850/858*	819	1252/5348*
France	297/1147*	850/858*	819	1252/5348*
Germany	273/1141*	850/858*	819	1252/5348*
Greece	875 or 423	869	813	1253/5349*
Hungary	870	852	912	1250/5346*
Iceland	871/1149*	850/858*	819	1252/5348*
International Latin-1	500/1148*			
Israel	424	862	916	1255/5351*
Italy	280/1144*	850/858*	819	1252/5348*
Latin America (Spanish)	284/1145*	850/858*	819	1252/5348*
FYR Macedonia	1025			1251/5347*
Netherlands (U.S. English)	37/1140*	850/858*	819	1252/5348*
New Zealand (U.S. English)	37/1140*	437	819	1252/5348*
Norway	277/1142*	850/858*	819	1252/5348*
Poland	870	852	912	1250/5346*

Table 121. Single-byte coded character set identifiers (CCSIDs) (continued)

Country or national language	EBCDIC	ASCII PC	ASCII AIX	ASCII Windows
Portugal (U.S. English)	37/1140*	850/858*	819	1252/5348*
Russia (Cyrillic)	1025			1251/5347*
Serbia and Montenegro (Cyrillic)	1025			1251/5347*
Serbia and Montenegro (Latin)	870	852	912	1250/5346*
Spain	284/1145*	850/858*	819	1252/5348*
Sweden	278/1143*	850/858*	819	1252/5348*
Switzerland	500 /1148*	850/858*	819	1252/5348*
Thailand	838			
Turkey (Latin 5)	1026	857	920	1254/5350*
United Kingdom	285/1146*	850/858*	819	1252/5348*
U.S.A. (U.S. English)	37/1140*	437	819	1252/5348*

Note: * This number represents the equivalent CCSIDs using the euro symbol.

To determine which combinations DB2 supports with SYSSTRINGS, issue the following SQL statement:

General-use Programming Interface

```
SELECT * FROM SYSIBM.SYSSTRINGS;
```

End of General-use Programming Interface

For more information about Unicode and Unicode CCSIDs, see “Unicode support” on page 510.

If you specify MIXED DATA=NO on installation panel DSNTIPF, specify an SBCS CCSID from Table 122 on page 517 in the EBCDIC CCSID field on DSNTIPF. You must specify a SBCS CCSID from Table 123 on page 517 in the ASCII CCSID field on DSNTIPF. Mixed character data and graphic data cannot be defined on a system when you specify MIXED DATA=NO.

If you specify MIXED DATA=YES on installation panel DSNTIPF, specify a mixed data CCSID from Table 122 on page 517 in the EBCDIC CCSID field on DSNTIPF. You must specify a mixed CCSID from Table 123 on page 517 in the ASCII CCSID field. Table 122 on page 517 and Table 123 on page 517 show the associated CCSIDs that DB2 assigns for SBCS and DBCS data when you specify a specific MCCSID.

In these tables, the terms are used as follows:

SCCSID
: single-byte coded character set identifier

MCCSID
: mixed coded character set identifier

GCCSID
graphic coded character set identifier

Table 122. EBCDIC double-byte coded character set identifiers (CCSIDs)

National language	MCCSID	SCCSID	GCCSID	User-defined characters
Japanese (Extended Katakana)	930	290	300	4370
# Japanese (Katakana-Kanji)	1390	8482	16684	6205
Japanese (Extended Katakana)	5026	290	4396	1880
Japanese (Extended English)	939	1027	300	4370
# Japanese (Latin-Kanji)	1399	5123	16684	6205
Japanese (Extended English)	5035	1027	4396	1880
Korean	933	833	834	1880
Korean	1364	13121	4930	1880
Simplified Chinese	935	836	837	1880
Simplified Chinese	1388	13124	4933	1880
Traditional Chinese	937	28709	835	6204

Then specify a mixed CCSID from Table 123 in the ASCII CCSID field on DSNTIPF. By specifying a CCSID for mixed data (an MCCSID), you also receive system CCSIDs for SBCS and DBCS (graphic) data.

Table 123. ASCII double-byte coded character set identifiers (CCSIDs)

National language	MCCSID	SCCSID	GCCSID	User-defined characters
Japanese	932	897	301	1880
Japanese (Extended)	942	1041*	301	1880
# # Japanese (Open environment)	943	1041*	941	1880
Japanese (HP)	5039	1041*	1351	940
Korean	949	1088	951	1880
Korean (EUC)	970	367	971	1880
Korean	1363	1126	1362	1880
Simplified Chinese	1381	1115	1380	1880
Simplified Chinese (EUC)	1383	367	1382	
Simplified Chinese	1386	5210	1385	1880
Traditional Chinese	938	904	927	6204
Traditional Chinese	948	1043	927	6204
Traditional Chinese (IBM Big-5)	950	1114	947	

Table 123. ASCII double-byte coded character set identifiers (CCSIDs) (continued)

National language	MCCSID	SCCSID	GCCSID	User-defined characters
Note: * The SCCSID 1041 is a superset of SCCSID 897.				

In Table 122 on page 517 and Table 123 on page 517, four CCSIDs are listed for Japanese to allow for all possible combinations of two single-byte code pages and two double-byte character sets. The difference between the single-byte code pages is in the code points for lowercase Latin letters and Katakana characters. In the code page for Japanese (Extended English, SCCSID 1027), lowercase letters have the same code points as other EBCDIC code pages.

Special considerations

Expanding conversions: An *expanding conversion* occurs when the length of the converted string is greater than that of the source string. For example, an expanding conversion occurs when an ASCII mixed data string that contains DBCS characters is converted to EBCDIC mixed data. Because of the addition of shift characters, an error occurs when an expanding conversion is performed on a fixed-length input host variable that requires conversion from ASCII mixed data to EBCDIC mixed data. The solution is to use a varying-length string variable with a maximum length that is sufficient to contain the expansion. Expanding conversions also can occur when string data is converted to or from Unicode.

Contracting conversions: A *contracting conversion* occurs when the length of the converted string is smaller than that of the source string. For example, a contracting conversion occurs when an EBCDIC mixed data string that contains DBCS characters is converted to ASCII mixed data due to the removal of shift characters. Contracting conversions also can occur when string data is converted to or from Unicode.

| ***Multiple CCSIDs referenced in an SQL statement:*** After DB2 is in enabling-new-function mode, multiple CCSIDs can be referenced from the same SQL statement. In previous versions of DB2, an error was returned if an SQL statement referenced more than one CCSID. This restriction is removed in Version 8 to allow table objects (tables, views, temporary tables, query tables, and user-defined functions) with different CCSID sets that are to be referenced in a statement.

Converting to the euro symbol

DB2 enables users to migrate to CCSIDs that support the euro symbol. This support is limited to conversion from specific CCSIDs that do not define the euro symbol to specific CCSIDs that define the euro symbol. See Table 124 on page 521 or Table 125 on page 521 for the list of ASCII and EBCDIC CCSIDs that can be converted to the euro symbol. In most cases, the euro symbol replaces an existing code point such as the International Currency Symbol (ICS). Unicode UTF-8 (1208) and UTF-16 (1200) support the euro symbol. Unicode SBCS data (367) does not support the euro symbol.

| **Attention:** Altering of CCSIDs can be very disruptive to a system. Perform alternations of CCSIDs only during a maintenance window. DB2 supports only one set of CCSIDs per encoding scheme (ASCII or EBCDIC). All databases and table spaces within an encoding scheme must be altered at the same time. DB2 checks compatibility of encoding schemes for SQL statements such as joins. **Failure to**

alter all databases and table spaces within an encoding scheme can result in unpredictable results, including data corruption. Altering CCSIDs without specific guidance from IBM Software Support is strongly discouraged.

Recommendation: If your subsystem communicates via DRDA with another DB2 subsystem that has an incorrect CCSID, the other DB2 subsystem should be altered at the same time. Contact IBM Software Support for guidance.

Before attempting to alter the CCSID of a system, obtain the following information:

1. Current CCSID
2. Planned CCSID
3. List of databases that are created with current CCSID
4. List of table spaces that are created with current CCSID
5. List of table spaces containing tables that define LOBs
6. List of LOB table spaces
7. List of views that are defined on tables in the table spaces that are to have the CCSID altered
8. Definitions of those views
9. List of all views created in Version 7 and Version 8
10. Authorization information on the views

Two methods are available for changing CCSIDs. You might need to use one method to change the CCSID on one table space, and the other method to change the CCSID on another table space.

Use the first method if you are certain that your data does not contain the international currency symbol.

1. Modify the CCSID data in DSNHDECP by running the installation CLIST and specifying UPDATE on installation panel DSNTIPA1. From panel DSNTIPB, select the OPERATOR FUNCTIONS panel and specify a unique name in the PARAMETER MODULE field. After returning to panel DSNTIPB, select the Applications Programming Defaults Panel 1. For details about the update process, see “The update process” on page 247.
2. Edit member DSN6SPRC of the SDSNMACS library to change the setting for SPRMCTU to 1. Save this change
3. Run the first six steps of job DSNTIJUZ, which you created in the preceding steps.
4. Stop DB2.
5. Start DB2 using the system parameter module that you specified in step 2 to use the new parameters.
6. Alter databases. This affects only the default for new table spaces that are created in the altered database.
7. Drop any views on tables that exist in any table space that you want to alter.
8. Drop any views on tables that were created in Version 7 and Version 8.
9. Alter the CCSIDs on the table spaces. This invalidates any plans or packages that reference these table spaces.

   Important: If the table space contains LOBs, you cannot use the ALTER TABLESPACE command. You must update the base table spaces and the AUX table spaces.
10. Run the REPAIR utility. Issue the following command:
    ```
    REPAIR DBD DIAGNOSE
    ```

If any database descriptors need to be repaired, issue the following command:

```
REPAIR DBD REBUILD
```

Run the REPAIR utility again to verify that all changes were performed successfully. Issue the following command:

```
REPAIR DBD DIAGNOSE
```

11. Run job DSNJU003 to delete CCSIDs that are in the BSDS.
12. Re-create views.
13. Re-create authorizations on the views.
14. Update the CCSID fields in SYSIBM.SYSPARMS to reflect the new CCSID.
15. Update the PARAMETER_CCSID field in SYSIBM.SYSROUTINES if necessary.
16. Edit member DSN6SPRC of the SDSNMACS library to return the setting of SPRMCTU to 0. Save this change.
17. Rebind the invalidated plans and packages either manually or with autobind.
18. Stop DB2 and restart it using your usual system parameter module.

Recommendation: After you complete this process, run job DSNTIJIC to create an image copy of the Version 8 catalog.

If you are not certain if your data contains the international currency symbol, you must use the following method:

1. Modify the CCSID data in DSNHDECP by running the installation CLIST and specifying UPDATE on installation panel DSNTIPA1. From panel DSNTIPB, select the OPERATOR FUNCTIONS panel and specify a unique name in the PARAMETER MODULE field. After returning to panel DSNTIPB, select the Applications Programming Defaults Panel 1. For details about the update process, see "The update process" on page 247.
2. Edit member DSN6SPRC of the SDSNMACS library to change the setting for SPRMCTU to 1. Save this change
3. Run the first six steps of job DSNTIJUZ, which you created in the preceding steps.
4. Stop DB2.
5. Start DB2 using the system parameter module that you specified in step 2 to use the new parameters.
6. Unload data from the tables that you want to alter.
7. Alter databases. This affects only the default for new table spaces that are created in the altered database.
8. Alter the CCSIDs on the table spaces. This invalidates any plans or packages that reference these table spaces.

   Important: If the table space contains LOBs, you cannot use the ALTER TABLESPACE command. You must update the base table spaces and the AUX table spaces.
9. Run the REPAIR utility. Issue the following command:

   ```
   REPAIR DBD DIAGNOSE
   ```

   If any database descriptors need to be repaired, issue the following command:

   ```
   REPAIR DBD REBUILD
   ```

   Run the REPAIR utility again to verify that all changes were performed successfully. Issue the following command:

   ```
   REPAIR DBD DIAGNOSE
   ```

10. Reload data from tables.
11. Run job DSNJU003 to delete CCSIDs that are in the BSDS.
12. Re-create views.
13. Re-create authorizations on the views.
14. Update the CCSID fields in SYSIBM.SYSPARMS to reflect the new CCSID.
15. Update the PARAMETER_CCSID field in SYSIBM.SYSROUTINES if necessary.
16. Edit member DSN6SPRC of the SDSNMACS library to return the setting of SPRMCTU to 0. Save this change.
17. Rebind the invalidated plans and packages either manually or with autobind.
18. Stop DB2 and restart it using your usual system parameter module.

Recommendation: After you complete this process, run job DSNTIJIC to create an image copy of the Version 8 catalog.

The list of CCSIDs that you can modify are listed according to the encoding scheme. Table 124 lists the EBCDIC CCSIDs, and Table 125 lists the ASCII CCSIDs.

Table 124. EBCDIC CCSID values that convert to euro CCSIDs

CCSIDs without euro symbol	CCSIDs with euro symbol
37	1140
273	1141
277	1142
278	1143
280	1144
284	1145
285	1146
297	1147
500	1148
871	1149

Table 125. ASCII CCSID values that convert to euro CCSIDs

CCSIDs without euro symbol	CCSIDs with euro symbol
850	858
1250	5346
1251	5347
1252	5348
1253	5349
1254	5350
1255	5351
1256	5352
1257	5353
874	4970

You cannot convert other CCSIDs to the euro-supported CCSIDs. Alter all databases and all table spaces within an encoding scheme at the same time.

How an entry in SYSIBM.SYSSTRINGS works

The catalog table SYSIBM.SYSSTRINGS contains the following columns:

INCCSID	The source CCSID of a character conversion.
OUTCCSID	The target CCSID of a character conversion.
TRANSTYPE	The type of conversion:
	SS SBCS data to SBCS data
	SM SBCS data to EBCDIC MIXED data
	MS EBCDIC MIXED data to SBCS (EBCDIC and ASCII) data
	PS ASCII MIXED data to SBCS (EBCDIC and ASCII) data
	GG GRAPHIC data to GRAPHIC data
	PM ASCII MIXED data to EBCDIC MIXED data
	MM EBCDIC MIXED data to EBCDIC MIXED data
	MP EBCDIC MIXED to ASCII MIXED data
	PP ASCII MIXED to ASCII MIXED data
	SP SBCS (ASCII and EBCDIC) to ASCII MIXED data
ERRORBYTE	Specifies the byte that is used in the conversion table (TRANSTAB) as an error indicator. For example, if ERRORBYTE is X'3E', that byte is used in the conversion table to indicate that no conversion is defined for code points that map to X'3E'. Null indicates the absence of an error indicator.
SUBBYTE	Specifies the byte that is used in the conversion table (TRANSTAB) as a substitution character. For example, if SUBBYTE is X'3F', that byte is used in the conversion table as a substitute for code points that map to X'3F'. A warning occurs when a code point maps to the value of SUBBYTE. Null indicates the absence of a substitution character.
TRANSPROC	The name of a module or a blank string. If IBMREQD is N, a non-blank value of TRANSPROC is the name of a user-provided conversion procedure. If IBMREQD is Y, a non-blank value of TRANSPROC is the name of a DB2 module that contains DBCS conversion tables.
IBMREQD	Y indicates that the row is provided by IBM. N indicates that the row has been inserted by the user.
TRANSTAB	A 256-byte conversion table or an empty string.

Each row of SYSSTRINGS contains information about the conversion of character strings from the coded character set that is identified by INCCSID to the coded character set that is identified by OUTCCSID. The conversion function is automatically invoked when a conversion from the coded character set that is identified by the INCCSID column to the coded character set that is identified by the OUTCCSID column is required.

For example, the row of SYSSTRINGS in which the value of INCCSID is 500 and the value of OUTCCSID is 37 describes the conversion from CCSID 500 to CCSID 37. The row in which the value of INCCSID is 37 and the value of OUTCCSID is 500 describes the conversion from CCSID 37 to CCSID 500.

DB2 enforces a distinction between IBM-supplied rows and user-provided rows with the following constraints:

- Rows with IBMREQD=Y cannot be updated or deleted.

- Rows with IBMREQD=N can be inserted, updated, and deleted.
- The same pair of CCSIDs can be in two rows, if one is in an IBM-supplied row and the other is in a user-provided row. In this case, the user-provided row is used for the character conversion.

Table 126 lists the types of rows that are possible in SYSSTRINGS.

Table 126. Types of rows in SYSSTRINGS

The value of TRANSPROC is	The value of TRANSTAB is	The value of IBMREQD is	The result is
blank	an empty string	–	No conversion is performed.
not blank	–	NO	Conversion is performed by the conversion procedure module name identified in the TRANSPROC column
blank	not empty	–	Conversion is performed by the DB2 module using the conversion table identified in TRANSTAB
–	–	–	Refer to *z/OS C/C++ Programming Guide* for additional conversions that are supported

Be aware of the following rules for SYSSTRINGS entries:

- An INSERT, UPDATE, DELETE, or LOAD is allowed only if IBMREQD=N.
- The values in the INCCSID and OUTCCSID columns must be in the range of 1 to 65533.
- For any given row, the INCCSID and OUTCCSID columns cannot contain the same value.
- The value in the TRANSTYPE column must be SS, SM, MS, PS, MM, PM, GG, MP, PP, or SP.
- For any given row, the ERRORBYTE and SUBBYTE columns cannot contain the same nonnull value.
- The TRANSPROC column must either be blank or contain a string that conforms to the rules for z/OS program names.
- The length that is specified in the TRANSTAB column must be either 0 or 256.

When remote packages should be rebound

Certain conversion-related changes at the local DBMS or at a remote DBMS might force the rebinding of a package. These include the following changes:

- The system CCSID at the remote DBMS was changed. In this case, always rebind the package.
- The system CCSID at the local DBMS was changed. This could happen, for example, if the wrong system CCSID was specified during installation. If so, string constants in static SQL statements might have been converted incorrectly during the binding of the package. Rebinding corrects the conversion. Other problems might also arise as a result of the change. Indeed, rebinding is generally recommended.
- The subtype of a character column is changed at the remote DBMS. The pertinent changes are from BIT to either SBCS or MIXED, and from SBCS or MIXED to BIT. The change was probably made by modifying the FOREIGNKEY column of the SYSIBM.SYSCOLUMNS table in the remote system catalog. Alternatively, the change might have occurred if the table was dropped and re-created with a different subtype (BIT to either SBCS or MIXED, or either SBC or MIXED to BIT) after the application was bound. A statement that refers to a

column with a modified subtype might fail with an SQLCODE of -333. If this occurs, rebind the package containing the statement.

Specifying locales for uppercase and lowercase conversion of EBCDIC data

Rules for uppercase and lowercase usage vary according to language and country. A *locale* defines the subset of a user's environment that depends on language and cultural conventions. DB2 uses the information that is associated with a locale to execute UPPER, LOWER, and TRANSLATE functions in a culturally correct manner. A locale consists of two components: the first component represents a specific language and country, and the second component is a CCSID.

Example: In the locale, Fr_CA.IBM-1047, Fr_CA represents the language and country (French Canadian), and IBM-1047 is the associated CCSID.

The symbol for euro currency is supported through the modifier @EURO.

Example: To display results in euro dollars instead of French Francs, specify Fr_FR@EURO.

DB2 uses the SCEELKED and SCEERUN Language Environment (LE) libraries. Both libraries are PDS libraries and are for non-XPLINK linkage only. If you need to customize the locales using LE libraries, see *z/OS C/C++ Programming Guide*.

Table 127 shows a partial list of locales that are supplied with z/OS C/C++. For a more complete list of locales, see *z/OS C/C++ Programming Guide*.

Table 127. Examples of locales supplied with z/OS C/C++. Excerpt of table from z/OS C/C++ Programming Guide

Locale	Language	Country	Code set	Load module name
De_CH.IBM-500	German	Switzerland	IBM-500	EDC$DCEO
De_CH.IBM-1047	German	Switzerland	IBM-1047	EDC$DCEY
De_DE.IBM-273	German	Germany	IBM-273	EDC$DDEB
De_DE.IBM-1047	German	Germany	IBM-1047	EDC$DDEY
Fr_CA.IBM-037	French	Canada	IBM-037	EDC$FCEA
Fr_CA.IBM-1047	French	Canada	IBM-1047	EDC$FCEY
It_IT.IBM-280	Italian	Italy	IBM-280	EDC$ITEG
It_IT.IBM-1047	Italian	Italy	IBM-1047	EDC$ITEY
Ja_JP.IBM-290	Japanese	Japan	IBM-290	EDC$JAEL
Ja_JP.IBM-930	Japanese	Japan	IBM-930	EDC$JAEU
Ja_JP.IBM-939	Japanese	Japan	IBM-939	EDC$JAEV
Ja_JP.IBM-1027	Japanese	Japan	IBM-1027	EDC$JAEX

Uppercase and lowercase conversion of Unicode data

If you want to use the UPPER or LOWER built-in functions to process Unicode
data, and if you want to performing uppercase conversions or lowercase
conversions for characters other than A-Z or a-z, you must perform additional
setup. You must add control statements to the configuration of z/OS Support for
Unicode services.

Example: Assume that your EBCDIC CCSID is 37 and your ASCII CCSID is 819, and that you have defined the following conversions:

```
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;
CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

Modify your conversion image as follows:

```
CASE NORMAL;   /* normal casing */
CASE SPECIAL;  /* additional locale-independent casing */
CASE LOCALE;   /* additional locale-dependent casing */
CONVERSION 37,367,ER;
CONVERSION 37,1208,ER;
CONVERSION 37,1200,ER;
CONVERSION 367,37,ER;
CONVERSION 1208,37,ER;
CONVERSION 1200,37,ER;
CONVERSION 819,367,ER;
CONVERSION 819,1208,ER;
CONVERSION 819,1200,ER;
CONVERSION 367,819,ER;
CONVERSION 1208,819,ER;
CONVERSION 1200,819,ER;
```

The three additional CASE statements provide the necessary infrastructure for the UPPER and LOWER functions to process Unicode data according to the Unicode Standard.

Appendix B. CATENFM and CATMAINT

This appendix describes the “CATENFM” and “CATMAINT” on page 529 utilities. These utilities are used to tailor and repair the catalog.

CATENFM

The CATENFM utility enables a DB2 subsystem to enter DB2 Version 8 enabling-new-function mode and Version 8 new-function mode. It also enables a DB2 subsystem to return to enabling-new-function mode from new-function mode.

All new Version 8 functions are unavailable when the subsystem is in compatibility mode or enabling-new-function mode.

Output: Output from the CATENFM utility consists of:

- If you specify the CONVERT option, the CATENFM utility converts 18 table spaces during the enabling-new-function mode process.
- If you specify the ALTER option, some objects in the DB2 catalog are altered or created.
- For other options, there is no output.

Authorization required: The required authorization for CATENFM is installation SYSADM.

Execution phases of CATENFM: The CATENFM utility operates in these phases:

Phase	**Description**
UTILINIT	Performs initialization and setup
UTILTERM	Performs cleanup

Syntax and options of the control statement

The CATENFM utility is invoked by jobs DSNTIJNE, DSNTIJNF, DSNTIJNH, and DSNTIJEN.

Syntax diagram

For guidance in interpreting syntax diagrams, see “How to read the syntax diagrams” on page xiv.

```
►►──CATENFM───┬─START──────────────────────────┬──────────────────────────────────────────►◄
              ├─COMPLETE───────────────────────┤
              ├─HALTENFM───────────────────────┤
              ├─ENFMON─────────────────────────┤
              └─CONVERT──INPUT──table-space-name─┘
```

Option descriptions

DB2 Utility Guide and Reference provides general information about specifying options for DB2 utilities.

START
Invokes the CATENFM utility and indicates the start of enabling-new-function mode processing. Creates the index and table in the SYSALTER tablespace. Moves catalog table SYSDUMMY1 to SYSEDCBC table space. No start processing is done if CATENFM START was run previously; however, you can run the CATENFM utility as many times as needed.

ALTER

COMPLETE
Checks if the DB2 subsystem has completed enabling-new-function mode processing. If the subsystem has completed this processing, the CATENFM utility returns 0, and the subsystem enters new-function mode.

ENFMON
Returns DB2 to enabling-new-function mode. New Version 8 are not available in Version 8 enabling-new-function mode.

HALTENFM
Stops enabling-new-function mode processing after the completion of the step that is currently executing.

CONVERT
Starts enabling-new-function mode processing for the table space that is listed after the INPUT keyword.

INPUT *table-space-name*
Specifies the table space for which enabling-new-function mode processing should begin.

table-space-name
The name of the table space for which enabling-new-function mode processing should begin.

Instructions for converting the catalog

To convert the catalog, you must run the DSNTIJNF job.

Before converting the catalog

Before you run the CATENFM utility to convert the catalog, take image copies of all catalog and directory objects and save your entire subsystem.

Data sets that CATENFM uses when converting the catalog

A CATENFM job allocates all of the data sets that it needs. CATENFM uses data sets only when the CONVERT option is specified. Table 128 lists the data sets that CATENFM uses during conversion. The table lists the DD name that is used to identify the data set, a description of the data set, and an indication of whether it is required.

Table 128. Data sets that CATENFM uses during conversion

Data set	Description	Required?
SYSIN	Input data set that contains the utility control statement.	Yes
SYSPRINT	Output data set for messages.	Yes

Instructions for specific tasks

The following task is required when you convert to DB2 Version 8 new-function mode.

Converting to new-function mode: When you migrate to DB2 Version 8, the DB2 subsystem enters compatibility mode. In compatibility mode, the DB2 subsystem can coexist with other data sharing members that are at either Version 7 or Version 8 compatibility mode.

The DB2 subsystem leaves compatibility mode and enters enabling-new-function mode when you invoke CATENFM START by running the DSNTIJNE job. The subsystem cannot begin enabling-new-function mode processing if any Version 7 members are active in the data sharing group. All members, including members that are not converting to Version 8 new-function mode, must be running Version 8 when the subsystem enters enabling-new-function mode. Note that when a member starts enabling-new-function mode, the group enters enabling-new-function mode.

After enabling-new-function mode completes, the DB2 subsystem can enter Version 8 new-function mode. All new Version 8 functions are unavailable until the DB2 subsystem enters new-function mode.

The DSNTIJNE job runs CATENFM START, which causes the DB2 subsystem to enter enabling-new-function mode. Run CATENFM START only when you are ready to begin the enabling-new-function mode conversion process.

Terminating or halting CATENFM

You can terminate CATENFM by using the TERM UTILITY command.

You can stop the enabling-new-function mode processing by specifying CATENFM HALTENFM. This statement stops the enabling-new-function mode processing at the completion of the step that is currently executing.

CATENFM CONVERT cannot be restarted. If you attempt to restart CATENFM CONVERT, you receive message DSNU191I, which states that the utility cannot be restarted. You must terminate the job, and rerun job DSNTIJNE from the beginning to convert the catalog.

Concurrency and compatibility

Certain catalog and directory objects are not available during some of the CATENFM phases. The objects that are unavailable vary based on the CATENFM option that you specify. The unavailability of these objects can cause other jobs to time out with message DSNT3761 or DSNT5011. You cannot run CATMAINT when the DB2 catalog or directory are in UT status.

CATMAINT

The CATMAINT utility updates the catalog; run this utility during migration to a new release of DB2 or when IBM Software Support instructs you to do so.

Output: Output for CATMAINT UPDATE is the updated catalog.

Authorization required: The required authorization for CATMAINT is installation SYSADM.

Execution phases of CATMAINT: The CATMAINT utility operates in these phases:

Phase	Description
UTILINIT	Performs initialization

UTILTERM Performs cleanup

Syntax and options of the control statement

The utility control statement defines the function that the utility job performs. Use the ISPF/PDF edit function to create a control statement and to save it in a sequential or partitioned data set. When you create the JCL for running the job, use the SYSIN DD statement to specify the name of the data set that contains the utility control statement.

Syntax diagram

For guidance in interpreting syntax diagrams, see "How to read the syntax diagrams" on page xiv.

```
►►──CATMAINT──UPDATE──────────────────────────────────────────────────────►◄
```

Option descriptions

DB2 Utility Guide and Reference provides general information about specifying options for DB2 utilities.

UPDATE
Indicates that you want to update the catalog. Run this option only when you migrate to a new release of DB2 or when IBM Software Support instructs you to do so.

Instructions for running CATMAINT

To run CATMAINT, you must:

1. Read "Before running CATMAINT."
2. Prepare the necessary data sets, as described in "Data sets that CATMAINT uses."
3. Create JCL statements by using one of the methods that are described in *DB2 Utility Guide and Reference*.
4. Prepare a utility control statement that specifies the options for the tasks you want to perform, as described in "Instructions for specific tasks" on page 531.
5. Check "Concurrency and compatibility" on page 531 if you want to run other jobs concurrently on the same target objects.
6. Run CATMAINT by using one of the methods that are described in *DB2 Utility Guide and Reference*.

Before running CATMAINT

The work file database is used for CATMAINT sorting. Prior to executing the CATMAINT utility, calculate the size of the work file database.

To calculate the size of the work file database, see "Work file database storage requirements" on page 22.

Data sets that CATMAINT uses

Include DD statements for all data sets that your job uses. Table 129 lists the data sets that CATMAINT uses. The table lists the DD name that is used to identify the data set, a description of the data set, and an indication of whether it is required.

Table 129. Data sets that CATMAINT uses

Data	Description	Required?
SYSIN	An input data set that contains the utility control statement	Yes
SYSPRINT	An output data set for messages	Yes

Instructions for specific tasks

To perform the following task, specify the options and values for that task in your utility control statement.

Updating the catalog for a new release: When you install or migrate to a new release of DB2, you must update the catalog for the prior release to the new version. The DSNTIJTC job runs CATMAINT UPDATE to update the catalog. DB2 displays migration status message DSNU777I at several points during CATMAINT execution.

If an abend occurs during migration processing, message DSNU776I or DSNU778I can give you information about the problem.

Terminating or restarting CATMAINT

You can terminate CATMAINT by using the TERM UTILITY command, but the termination can leave some indexes in REBUILD-pending status. See *DB2 Utility Guide and Reference* for information about resetting this status.

CATMAINT cannot be restarted. If you attempt to restart CATMAINT, you receive message DSNU191I, which states that the utility cannot be restarted. You must terminate the job with the TERM UTILITY command, and rerun CATMAINT from the beginning.

Concurrency and compatibility

Many catalog and directory indexes are not available while CATMAINT is running. The unavailability of these indexes can cause other jobs to time out with message DSNT318I, DSNT376I or DSNT501I.

Appendix C. Directory of subsystem parameters

This appendix describes the subsystem parameters.

The following topics provide additional information:
• "Editing the subsystem parameters and DSNHDECP values"
• "Directory of subsystem parameters and DSNHDECP values"

Editing the subsystem parameters and DSNHDECP values

The subsystem parameter module is generated by job DSNTIJUZ each time you install, migrate, or update DB2. Seven macros expand to form this data-only subsystem parameter load module. It contains the DB2 execution-time parameters that you selected using the ISPF panels. These seven macros are DSN6ARVP, DSN6ENV, DSN6FAC, DSN6LOGP, DSN6SPRM, DSN6SYSP, and DSN6GRP.

The data-only load module DSNHDECP is also generated by job DSNTIJUZ. It contains the application programming defaults.

Directory of subsystem parameters and DSNHDECP values

Table 130 shows you each macro parameter, the macro where it is located, installation panel name, whether it can be updated online, and its corresponding page number. Online update capability does not apply to macro DSNHDECP so values are not listed for those parameters.

| Some parameters, when updated online, result in a change in system behavior.
| These parameters include:
| • PARTKEYU
| • SYSADM/SYSADM2
| • CACHEDYN
| • MAXKEEPD
| • XLKUPDLT

Table 130. Directory of subsystem parameters and DSNHDECP values

	Parameter	Macro	Panel	Update Online	Page
	ABEXP	DSN6SPRM	DSNTIPO	Yes	155
	ABIND	DSN6SPRM	DSNTIPO	Yes	155
|	ACCUMACC	DSN6SYSP	DSNTIPN	Yes	149
|	ACCUMUID	DSN6SYSP	DSNTIPN	Yes	149
|	AEXITLIM	DSN6SPRM	DSNTIPP	Yes	194
	AGCCSID	DSNHDECP	DSNTIPF	—	163
	ALCUNIT	DSN6ARVP	DSNTIPA	Yes	210
#	ALL/dbname	DSN6SPRM	DSNTIPS	No	217
	AMCCSID	DSNHDECP	DSNTIPF	—	163
	APPENSCH	DSNHDECP	DSNTIPF	—	163
	ARCPFX1	DSN6ARVP	DSNTIPH	Yes	109
	ARCPFX2	DSN6ARVP	DSNTIPH	Yes	109
	ARCRETN	DSN6ARVP	DSNTIPA	Yes	210
	ARCWRTC	DSN6ARVP	DSNTIPA	Yes	210

Table 130. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online	Page
	ARCWTOR	DSN6ARVP	DSNTIPA	Yes	210
	ARC2FRST	DSN6LOGP	DSNTIPO	Yes	155
	ASCCSID	DSNHDECP	DSNTIPF	—	163
	ASSIST	DSN6GRP	DSNTIPK	No	106
	AUDITST	DSN6SYSP	DSNTIPN	No	149
	AUTH	DSN6SPRM	DSNTIPP	No	194
	AUTHCACH	DSN6SPRM	DSNTIPP	Yes	194
	BACKODUR	DSN6SYSP	DSNTIPL	No	204
	BINDNV	DSN6SPRM	DSNTIPP	Yes	194
	BLKSIZE	DSN6ARVP	DSNTIPA	Yes	210
	BMPTOUT	DSN6SPRM	DSNTIPI	Yes	182
|	CACHEDYN	DSN6SPRM	DSNTIP8	Yes	176
	CACHEPAC	DSN6SPRM	DSNTIPP	No	194
	CACHERAC	DSN6SPRM	DSNTIPP	No	194
	CATALOG	DSN6ARVP	DSNTIPA	Yes	210
	"	DSN6SPRM	DSNTIPA2	—	101
|	CDSSRDEF	DSN6SPRM	DSNTIP8	Yes	176
	CHARSET	DSNHDECP	DSNTIPF	—	163
|	CHGDC	DSN6SPRM	DSNTIPO	Yes	155
	CHKFREQ	DSN6SYSP	DSNTIPL	Yes	204
	CMTSTAT	DSN6FAC	DSNTIPR	No	219
#	COMCRIT	DSN6SPRM	—	Yes	259
	COMPACT	DSN6ARVP	DSNTIPA	Yes	210
	COMPAT	DSNHDECP	—	—	note 1
	CONDBAT	DSN6SYSP	DSNTIPE	Yes	140
	CONTSTOR	DSN6SPRM	DSNTIPE	Yes	140
	COORDNTR	DSN6GRP	DSNTIPK	No	106
	CTHREAD	DSN6SYSP	DSNTIPE	Yes	140
	DBACRVW	DSN6SPRM	DSNTIPP	Yes	194
	DBPROTCL	DSN6SYSP	DSNTIP5	Yes	225
	DATE	DSNHDECP	DSNTIP4	—	171
	DATELEN	DSNHDECP	DSNTIP4	—	171
	DB2SUPLD	DSNHDECP	—	—	note 1
	DDF	DSN6FAC	DSNTIPR	No	219
	DEALLCT	DSN6LOGP	DSNTIPA	Yes	210
	DECARTH	DSNHDECP	DSNTIP4	—	171
	DECDIV3	DSN6SPRM	DSNTIPF	No	163
	DECIMAL	DSNHDECP	DSNTIPF	—	163
	DEFLANG	DSNHDECP	DSNTIPF	—	163
	DEFLTID	DSN6SPRM	DSNTIPP	No	194
	DELIM	DSNHDECP	DSNTIPF	—	163
	DESCSTAT	DSN6SPRM	DSNTIP4	Yes	171
	DLDFREQ	DSN6SYSP	DSNTIPL	Yes	204
	DLITOUT	DSN6SPRM	DSNTIPI	Yes	182
	DSHARE	DSN6GRP	DSNTIPA1	No	94
	DSMAX	DSN6SPRM	DSNTIPC	Yes	238
	DSQLDELI	DSNHDECP	DSNTIPF	—	163
	DSSTIME	DSN6SYSP	DSNTIPN	Yes	149
|	DSVCI	DSN6SYSP	DSNTIP7	Yes	137
	DYNRULES	DSNHDECP	DSNTIP4	—	171
|	EDMBFIT	DSN6SPRM	DSNTIP8	Yes	176
|	EDMDBDC	DSN6SPRM	DSNTIPC	Yes	238

Table 130. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online	Page
|	EDMSTMTC	DSN6SPRM	DSNTIPC	Yes	238
	EDMPOOL	DSN6SPRM	DSNTIPC	Yes	238
|	EDPROP	DSN6SPRM	DSNTIPO	Yes	155
	ENSCHEME	DSNHDECP	DSNTIPF	—	163
|	EVALUNC	DSN6SPRM	DSNTIP8	Yes	176
|	EXTRAREQ	DSN6SYSP	DSNTIP5	Yes	225
|	EXTRASRV	DSN6SYSP	DSNTIP5	Yes	225
	EXTSEC	DSN6SYSP	DSNTIPR	Yes	219
	GCCSID	DSNHDECP	DSNTIPF	—	163
	GRPNAME	DSN6GRP	DSNTIPK	No	106
	HOPAUTH	DSN6SPRM	DSNTIP5	No	225
	IDBACK	DSN6SYSP	DSNTIPE	Yes	140
	IDFORE	DSN6SYSP	DSNTIPE	Yes	140
|	IDTHTOIN	DSN6FAC	DSNTIPR	Yes	219
	IDXBPOOL	DSN6SYSP	DSNTIP1	Yes	146
|	IMMEDWRI	DSN6GRP	DSNTIP8	Yes	176
	INLISTP	DSN6SPRM	—	Yes	259
	IRLMAUT	DSN6SPRM	DSNTIPI	No	182
	IRLMPRC	DSN6SPRM	DSNTIPI	No	182
	IRLMRWT	DSN6SPRM	DSNTIPI	No	182
	IRLMSID	DSN6SPRM	DSNTIPI	No	182
	IRLMSWT	DSN6SPRM	DSNTIPI	Yes	182
|	IXQTY	DSN6SYSP	DSNTIP7	Yes	137
	LBACKOUT	DSN6SYSP	DSNTIPL	No	204
	LC_CTYPE	DSNHDECP	DSNTIPF	—	163
	LEMAX	DSN6SPRM	DSNTIP7	No	137
	LOBVALA	DSN6SYSP	DSNTIP7	Yes	137
	LOBVALS	DSN6SYSP	DSNTIP7	Yes	182
	LOGAPSTG	DSN6SYSP	DSNTIPL	No	204
|	LRDRTHLD	DSN6SYSP	DSNTIPE	Yes	140
|	MAINTYPE	DSN6SPRM	DSNTIP8	Yes	176
|	MAX_NUM_CUR	DSN6SPRM	DSNTIPX	Yes	229
|	MAX_ST_PROC	DSN6SPRM	DSNTIPX	Yes	229
	MAXARCH	DSN6LOGP	DSNTIPA	No	210
	MAXDBAT	DSN6SYSP	DSNTIPE	Yes	140
|	MAXKEEPD	DSN6SPRM	DSNTIPE	Yes	140
	MAXRBLK	DSN6SPRM	DSNTIPC	Yes	238
	MAXRTU	DSN6LOGP	DSNTIPA	Yes	210
|	MAXTYPE1	DSN6FAC	DSNTIPR	Yes	219
	MCCSID	DSNHDECP	DSNTIPF	—	163
	MEMBNAME	DSN6GRP	DSNTIPK	No	106
|	MGEXTSZ	DSN6SYSP	DSNTIP7	Yes	137
	MINSTOR	DSN6SPRM	DSNTIPE	Yes	140
	MIXED	DSNHDECP	DSNTIPF	—	163
	MON	DSN6SYSP	DSNTIPN	No	149
	MONSIZE	DSN6SYSP	DSNTIPN	No	149
|	NEWFUN	DSNHDECP	DSNTIPA1	—	210
	NPGTHRSH	DSN6SPRM	—	Yes	259
	NUMLKTS	DSN6SPRM	DSNTIPJ	Yes	188
	NUMLKUS	DSN6SPRM	DSNTIPJ	Yes	188
#	OJPERFEH	DSN6SPRM	—	No	259
|	OPTHINTS	DSN6SPRM	DSNTIP8	Yes	176

Table 130. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online	Page
	OUTBUFF	DSN6LOGP	DSNTIPL	No	204
|	PADIX	DSN6SPRM	DSNTIPE	Yes	140
|	PADNTSTR	DSNHDECP	DSNTIP4	No	171
|	PARAMDEG	DSN6SPRM	DSNTIP8	Yes	176
	PARTKEYU	DSN6SPRM	DSNTIP8	Yes	176
	PCLOSEN	DSN6SYSP	DSNTIPL	Yes	204
	PCLOSET	DSN6SYSP	DSNTIPL	Yes	204
|	POOLINAC	DSN6FAC	DSNTIP5	Yes	225
	PRIQTY	DSN6ARVP	DSNTIPA	Yes	210
	PROTECT	DSN6ARVP	DSNTIPP	Yes	194
	PTASKROL	DSN6SYSP	—	Yes	259
	QUIESCE	DSN6ARVP	DSNTIPA	Yes	210
	RECALL	DSN6SPRM	DSNTIPO	No	155
	RECALLD	DSN6SPRM	DSNTIPO	Yes	155
|	REFSHAGE	DSN6SPRM	DSNTIP8	Yes	176
|	RELCURHL	DSN6SPRM	DSNTIP8	Yes	176
#	RESTART/DEFER	DSN6SPRM	DSNTIPS	No	217
|	RESYNC	DSN6FAC	DSNTIPR	Yes	219
	RETLWAIT	DSN6SPRM	DSNTIPI	Yes	182
|	RETVLCFK	DSN6SPRM	DSNTIP8	Yes	176
	RGFCOLID	DSN6SPRM	DSNTIPZ	No	232
	RGFDBNAM	DSN6SPRM	DSNTIPZ	No	232
	RGFDEDPL	DSN6SPRM	DSNTIPZ	No	232
	RGFDEFLT	DSN6SPRM	DSNTIPZ	No	232
	RGFESCP	DSN6SPRM	DSNTIPZ	No	232
	RGFFULLQ	DSN6SPRM	DSNTIPZ	No	232
	RGFINSTL	DSN6SPRM	DSNTIPZ	No	232
	RGFNMORT	DSN6SPRM	DSNTIPZ	No	232
	RGFNMPRT	DSN6SPRM	DSNTIPZ	No	232
	RLF	DSN6SYSP	DSNTIPO	No	155
	RLFAUTH	DSN6SYSP	DSNTIPP	Yes	194
	RLFERR	DSN6SYSP	DSNTIPO	Yes	155
	RLFERRD	DSN6FAC	DSNTIPR	Yes	219
	RLFTBL	DSN6SYSP	DSNTIPO	Yes	155
	ROUTCDE	DSN6SYSP	DSNTIPO	No	155
	RRULOCK	DSN6SPRM	DSNTIPI	Yes	182
	SCCSID	DSNHDECP	DSNTIPF	—	163
	SECQTY	DSN6ARVP	DSNTIPA	Yes	210
	SEQCACH	DSN6SPRM	DSNTIPE	Yes	140
	SEQPRES	DSN6SPRM	DSNTIPE	Yes	140
	SITETYP	DSN6SPRM	DSNTIPO	No	155
|	SJMXPOOL	DSN6SPRM	DSNTIP8	Yes	176
	SJTABLES	DSN6SPRM	—	Yes	259
	SKIPUNCI	DSN6SPRM	DSNTIP8	Yes	176
	SMF89	DSN6SYSP	—	Yes	259
	SMFACCT	DSN6SYSP	DSNTIPN	No	149
	SMFSTAT	DSN6SYSP	DSNTIPN	No	149
	SMSDCFL	DSN6SPRM	—	Yes	259
	SMSDCIX	DSN6SPRM	—	Yes	259
	SQLDELI	DSNHDECP	DSNTIPF	—	163
|	SRTPOOL	DSN6SPRM	DSNTIPC	Yes	238
	SSID	DSNHDECP	DSNTIPM	—	199

Table 130. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online	Page
|	STARJOIN	DSN6SPRM	DSNTIP8	Yes	176
|	STATHIST	DSN6SPRM	DSNTIPO	Yes	155
	STATIME	DSN6SYSP	DSNTIPN	Yes	149
	STATROLL	DSN6SPRM	DSNTIPO	Yes	155
	STATSINT	DSN6SPRM	DSNTIPO	Yes	155
	STDSQL	DSNHDECP	DSNTIP4	—	171
	STORMXAB	DSN6SYSP	DSNTIPX	Yes	229
	STORPROC	DSN6SYSP	DSNTIPX	No	229
	STORTIME	DSN6SYSP	DSNTIPX	Yes	229
	SUPERRS	DSN6SPRM	DSNTIPM	Yes	199
#	SUPPRESS_TS_CONV_WARNING	DSN6SPRM	—	Yes	259
|	SVOLARC	DSN6ARVP	DSNTIPA	Yes	210
	SYNCVAL	DSN6SYSP	DSNTIPN	Yes	149
|	SYSADM	DSN6SPRM	DSNTIPP	Yes	194
|	SYSADM2	DSN6SPRM	DSNTIPP	Yes	194
|	SYSOPR1	DSN6SPRM	DSNTIPP	Yes	194
|	SYSOPR2	DSN6SPRM	DSNTIPP	Yes	194
	TBSBPOOL	DSN6SYSP	DSNTIP1	Yes	146
|	TCPALVER	DSN6FAC	DSNTIP5	Yes	225
|	TCPKPALV	DSN6FAC	DSNTIP5	Yes	225
	TIME	DSNHDECP	DSNTIP4	—	171
	TIMELEN	DSNHDECP	DSNTIP4	—	171
	TRACSTR	DSN6SYSP	DSNTIPN	No	149
	TRACTBL	DSN6SYSP	DSNTIPN	No	149
	TRKRSITE	DSN6SPRM	DSNTIPO	No	155
|	TSQTY	DSN6SYSP	DSNTIP7	Yes	137
	TSTAMP	DSN6ARVP	DSNTIPH	Yes	109
	TWOACTV	DSN6LOGP	DSNTIPH	No	109
	TWOARCH	DSN6LOGP	DSNTIPH	No	109
	TWOBSDS	DSN6LOGP	—	No	331
	UGCCSID	DSNHDECP	DSNTIPF	—	163
|	UIFCIDS	DSN6SYSP	DSNTIPN	Yes	149
	UMCCSID	DSNHDECP	DSNTIPF	—	163
#	UNION_COLNAME_7	DSN6SPRM	—	Yes	259
	UNIT	DSN6ARVP	DSNTIPA	Yes	210
	UNIT2	DSN6ARVP	DSNTIPA	Yes	210
	URCHKTH	DSN6SYSP	DSNTIPL	Yes	204
	URLGWTH	DSN6SYSP	DSNTIPL	Yes	204
	USCCSID	DSNHDECP	DSNTIPF	—	163
	UTIMOUT	DSN6SPRM	DSNTIPI	Yes	182
	VOLTDEVT	DSN6SPRM	DSNTIPA2	Yes	210
	WLMENV	DSN6SYSP	DSNTIPX	Yes	229
|	XLKUPDLT	DSN6SPRM	DSNTIPI	Yes	182

Note 1: Serviceability parameter

Appendix D. DB2 utilities packaging

The following utilities are core utilities, which are included (at no extra charge) with Version 8 of DB2 UDB for z/OS:
- CATENFM
- CATMAINT
- DIAGNOSE
- LISTDEF
- OPTIONS
- QUIESCE
- REPAIR
- REPORT
- TEMPLATE
- All DSN stand-alone utilities

All other utilities are available as a separate product called the **DB2 Utilities Suite** (5655-K61, FMIDs JDB881K and JDB881M), which includes the following utilities:
- BACKUP SYSTEM
- CHECK DATA
- CHECK INDEX
- CHECK LOB
- COPY
- COPYTOCOPY
- EXEC SQL
- LOAD
- MERGECOPY
- MODIFY RECOVERY
- MODIFY STATISTICS
- REBUILD INDEX
- RECOVER
- REORG INDEX
- REORG TABLESPACE
- RESTORE SYSTEM
- RUNSTATS
- STOSPACE
- UNLOAD

All DB2 utilities operate on catalog, directory, and sample objects, without requiring any additional products.

The following topics provide additional information:
- "SMP/E jobs for DB2 utility products"
- "The operation of DB2 utilities in a mixed-release data sharing environment" on page 540

SMP/E jobs for DB2 utility products

To load the DB2 utility products, use System Modification Program Extended (SMP/E). SMP/E processes the installation tapes or cartridges and creates DB2 distribution target libraries.

DB2 provides several jobs that invoke SMP/E. These jobs are on the tape or cartridge that you received with the utility product. The job prologues in these jobs contain directions on how to tailor the job for your site. Follow these directions carefully to ensure that your DB2 UDB for z/OS SMP/E process works correctly. To copy the jobs from the tapes, submit the copy job that is listed in *DB2 Program Directory*.

The SMP/E RECEIVE job, DSNRECVS, loads the DB2 Diagnostic and Recovery Utilities program modules, macros, and procedures into temporary data sets (SMPTLIBs). The SMP/E RECEIVE job, DSNRECVK, loads the DB2 Operational Utilities program modules, macros, and procedures into temporary data sets (SMPTLIBs). If these jobs fail or abnormally terminate, correct the problem and rerun the jobs. Use "SMP/E step 4: Run the RECEIVE jobs: DSNRECV1, DSNRECV2, DSNRECV3, DSNRECV4" on page 61, as a guide to help you with the RECEIVE job.

The SMP/E APPLY job, DSNAPPLS, copies and link-edits the program modules, macros, and procedures for both the DB2 Diagnostic and Recovery Utilities and the DB2 Operational Utilities into the DB2 target libraries. Use "SMP/E step 6: Run the apply jobs: DSNAPPL1, DSNAPPL2" on page 62, as a guide to help you with the APPLY job.

The SMP/E ACCEPT job, DSNACCPS, copies the program modules, macros, and procedures for both the DB2 Diagnostic and Recovery Utilities and the DB2 Operational Utilities into the DB2 distribution libraries. Use "SMP/E step 7: Run the accept jobs: DSNACEP1, DSNACEP2" on page 63, as a guide to help you with the ACCEPT job.

The operation of DB2 utilities in a mixed-release data sharing environment

The utilities batch module, DSNUTILB, is split into multiple parts: a release-independent module called DSNUTILB and multiple release-dependent modules, DSNUT810 and the utility-dependent load modules that are listed in Table 131. To operate in a mixed-release data sharing environment, you must have the release-dependent modules from both releases and all applicable utility-dependent modules available to the utility jobs that operate across the data sharing group. The procedure for sharing utility modules is explained in Chapter 4 of *DB2 Data Sharing: Planning and Administration*. Use the information in Table 131 and the procedures that are outlined in Chapter 4 of *DB2 Data Sharing: Planning and Administration* to implement a mixed-release data sharing environment.

With Version 7 and subsequent releases, each utility has separate load modules and aliases. Table 131. lists the alias name and load module name or names for each utility.

| *Table 131. Relationship between utility names, aliases, and load modules*

Utility name	Alias name	Load module name
BACKUP SYSTEM and RESTORE SYSTEM	DSNU81AV	DSNU8RLV
CATMAINT and CATENFM [1]	DSNU81AA	DSNU8CLA
CHECK	DSNU81AB	DSNU8RLB
COPY	DSNU81AC	DSNU8OLC or DSNU8RLC
COPYTOCOPY	DSNU81AT	DSNU8RLT

Table 131. Relationship between utility names, aliases, and load modules (continued)

Utility name	Alias name	Load module name
DIAGNOSE	DSNU81AD	DSNU8CLD
EXEC SQL	DSNU81AU	DSNU8OLU
LISTDEF	DSNU81AE	DSNU8CLE
LOAD	DSNU81AF	DSNU8OLF
MERGECOPY	DSNU81AG	DSNU8RLG
MODIFY RECOVERY and MODIFY STATISTICS	DSNU81AH	DSNU8RLH
OPTIONS	DSNU81AI	DSNU8CLI
QUIESCE	DSNU81AJ	DSNU8CLJ
REBUILD INDEX	DSNU81AK	DSNU8OLK or DSNU8RLK
RECOVER	DSNU81AL	DSNU8OLL or DSNU8RLL
REORG INDEX and REORG TABLESPACE	DSNU81AM	DSNU8OLM
REPAIR	DSNU81AN	DSNU8CLN
REPORT	DSNU81AO	DSNU8CLO
RUNSTATS	DSNU81AP	DSNU8OLP
STOSPACE	DSNU81AQ	DSNU8OLQ
TEMPLATE	DSNU81AR	DSNU8CLR
UNLOAD	DSNU81AS	DSNU8OLS

Note: [1] The code for CATENFM is in the load module for CATMAINT.

Appendix E. How to obtain DB2 information

This section provides information that you can use to find valuable information about the DB2 product:

- "DB2 on the Web"
- "DB2 publications"
- "DB2 education" on page 544
- "How to order the DB2 library" on page 544

DB2 on the Web

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

You can view and search DB2 publications on the Web, or you can download and print many of the most current DB2 books. Follow links to other Web sites with more information about DB2 family and z/OS solutions. Access DB2 on the Web at the following Web site: www.ibm.com/software/db2zos.

DB2 publications

The publications for DB2 UDB for z/OS are available in various formats and delivery methods. IBM provides mid-version updates in softcopy on the Web and on CD-ROM.

DB2 Information Center for z/OS solutions

DB2 UDB for z/OS product information is viewable in the DB2 Information Center for z/OS solutions. The information center, introduced in Version 8 of DB2 UDB for z/OS, is a delivery vehicle for information about DB2 UDB for z/OS, IMS, QMF, and related tools. This information center enables users to search across related product information in multiple languages for data management solutions for the z/OS environment. Product technical information is provided in a format that offers more options and tools for accessing, integrating, and customizing information resources. The information center is based on Eclipse open source technology.

The DB2 Information Center for z/OS solutions is viewable at the following Web site: http://publib.boulder.ibm.com/infocenter/db2zhelp.

CD-ROMs and DVD

Books for Version 8 of DB2 UDB for z/OS are available on a CD-ROM that is included with your product shipment:

- DB2 UDB for z/OS Version 8 Licensed Library Collection, LK3T-7128, in English

The CD-ROM contains the collection of books for DB2 UDB for z/OS in PDF and BookManager formats. Periodically, IBM refreshes the books on subsequent editions of this CD-ROM.

The books for Version 8 of DB2 UDB for z/OS are also available on the following CD-ROM and DVD collection kits, which contain online books for many IBM products:

- IBM eServer zSeries Online Library: z/OS Software Products Collection, SK3T-4270, in English
- IBM eServer zSeries Online Library: z/OS Software Products DVD Collection, SK3T–4271, in English

PDF format

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

BookManager format

You can use online books on CD-ROM to read, search across books, print portions of the text, and make notes in these BookManager books. Using the IBM Softcopy Reader, appropriate IBM Library Readers, or the BookManager Read product, you can view these books in the z/OS, Windows, and VM environments. You can also view and search many of the DB2 BookManager books on the Web.

DB2 education

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

IBM also offers classes at your location, at a time that suits your needs. IBM can customize courses to meet your exact requirements. For more information, including the current local schedule, please contact your IBM representative.

How to order the DB2 library

You can order DB2 publications and CD-ROMs through your IBM representative or the IBM branch office that serves your locality. If your location is within the United States or Canada, you can place your order by calling one of the toll-free numbers:
- In the U.S., call 1-800-879-2755.
- In Canada, call 1-800-565-1234.

To order additional copies of licensed publications, specify the SOFTWARE option. To order additional publications or CD-ROMs, specify the PUBLICATIONS option. Be prepared to give your customer number, the product number, and either the feature codes or order numbers that you want.

You can also order books from the IBM Publication Center on the Web: www.elink.ibmlink.ibm.com.

From the IBM Publication Center, you can go to the Publication Notification System (PNS). PNS users receive electronic notifications of updated publications in their profiles. You have the option of ordering the updates by using the publications direct ordering application or any other IBM publication ordering channel. The PNS application does not send automatic shipments of publications. You will receive updated publications and a bill for them if you respond to the electronic notification.

Notices

Programming interface information

This book is intended to help you to install DB2 Universal Database for z/OS (DB2 UDB for z/OS).

This book also documents General-use Programming Interface and Associated Guidance Information provided by DB2 Universal Database for z/OS.

General-use programming interfaces allow the customer to write programs that obtain the services of DB2 UDB for z/OS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs, by an introductory statement to a chapter or section or by the following markings:

General-use Programming Interface

General-use Programming Interface and Associated Guidance Information ....

End of General-use Programming Interface

Product-sensitive Programming Interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this IBM software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive Programming Interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs either by an introductory statement to a chapter or section or by the following marking:

Product-sensitive Programming Interface

Product-sensitive Programming Interface and Associated Guidance Information ....

End of Product-sensitive Programming Interface

Trademarks

UNIX is a registered trademark of The Open Group in the United States and other countries.

Glossary

The following terms and abbreviations are defined as they are used in the DB2 library.

A

abend. Abnormal end of task.

abend reason code. A 4-byte hexadecimal code that uniquely identifies a problem with DB2.

abnormal end of task (abend). Termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve during execution.

access method services. The facility that is used to define and reproduce VSAM key-sequenced data sets.

access path. The path that is used to locate data that is specified in SQL statements. An access path can be indexed or sequential.

active log. The portion of the DB2 log to which log records are written as they are generated. The active log always contains the most recent log records, whereas the archive log holds those records that are older and no longer fit on the active log.

active member state. A state of a member of a data sharing group. The cross-system coupling facility identifies each active member with a group and associates the member with a particular task, address space, and z/OS system. A member that is not active has either a failed member state or a quiesced member state.

address space. A range of virtual storage pages that is identified by a number (ASID) and a collection of segment and page tables that map the virtual pages to real pages of the computer's memory.

address space connection. The result of connecting an allied address space to DB2. Each address space that contains a task that is connected to DB2 has exactly one address space connection, even though more than one task control block (TCB) can be present. See also *allied address space* and *task control block*.

| **address space identifier (ASID).** A unique
| system-assigned identifier for and address space.

administrative authority. A set of related privileges that DB2 defines. When you grant one of the administrative authorities to a person's ID, the person has all of the privileges that are associated with that administrative authority.

after trigger. A trigger that is defined with the trigger activation time AFTER.

agent. As used in DB2, the structure that associates all processes that are involved in a DB2 unit of work. An *allied agent* is generally synonymous with an *allied thread*. *System agents* are units of work that process tasks that are independent of the allied agent, such as prefetch processing, deferred writes, and service tasks.

aggregate function. An operation that derives its
result by using values from one or more rows. Contrast
with *scalar function*.

alias. An alternative name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 subsystem.

allied address space. An area of storage that is external to DB2 and that is connected to DB2. An allied address space is capable of requesting DB2 services.

allied thread. A thread that originates at the local DB2 subsystem and that can access data at a remote DB2 subsystem.

allocated cursor. A cursor that is defined for stored procedure result sets by using the SQL ALLOCATE CURSOR statement.

already verified. An LU 6.2 security option that allows DB2 to provide the user's verified authorization ID when allocating a conversation. With this option, the user is not validated by the partner DB2 subsystem.

| **ambiguous cursor.** A database cursor that is in a plan
| or package that contains either PREPARE or EXECUTE
| IMMEDIATE SQL statements, and for which the
| following statements are true: the cursor is not defined
| with the FOR READ ONLY clause or the FOR UPDATE
| OF clause; the cursor is not defined on a read-only
| result table; the cursor is not the target of a WHERE
| CURRENT clause on an SQL UPDATE or DELETE
| statement.

American National Standards Institute (ANSI). An organization consisting of producers, consumers, and general interest groups, that establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States.

ANSI. American National Standards Institute.

APAR. Authorized program analysis report.

APAR fix corrective service. A temporary correction of an IBM software defect. The correction is temporary, because it is usually replaced at a later date by a more permanent correction, such as a program temporary fix (PTF).

APF. Authorized program facility.

API. Application programming interface.

APPL. A VTAM network definition statement that is used to define DB2 to VTAM as an application program that uses SNA LU 6.2 protocols.

application. A program or set of programs that performs a task; for example, a payroll application.

application-directed connection. A connection that an application manages using the SQL CONNECT statement.

application plan. The control structure that is produced during the bind process. DB2 uses the application plan to process SQL statements that it encounters during statement execution.

application process. The unit to which resources and locks are allocated. An application process involves the execution of one or more programs.

application programming interface (API). A functional interface that is supplied by the operating system or by a separately orderable licensed program that allows an application program that is written in a high-level language to use specific data or functions of the operating system or licensed program.

application requester. The component on a remote system that generates DRDA® requests for data on behalf of an application. An application requester accesses a DB2 database server using the DRDA application-directed protocol.

application server. The target of a request from a remote application. In the DB2 environment, the application server function is provided by the distributed data facility and is used to access DB2 data from remote applications.

archive log. The portion of the DB2 log that contains log records that have been copied from the active log.

ASCII. An encoding scheme that is used to represent strings in many environments, typically on PCs and workstations. Contrast with *EBCDIC* and *Unicode*.

| **ASID.** Address space identifier.

attachment facility. An interface between DB2 and TSO, IMS, CICS, or batch address spaces. An attachment facility allows application programs to access DB2.

attribute. A characteristic of an entity. For example, in database design, the phone number of an employee is one of that employee's attributes.

authorization ID. A string that can be verified for connection to DB2 and to which a set of privileges is allowed. It can represent an individual, an organizational group, or a function, but DB2 does not determine this representation.

authorized program analysis report (APAR). A report of a problem that is caused by a suspected defect in a current release of an IBM supplied program.

authorized program facility (APF). A facility that permits the identification of programs that are authorized to use restricted functions.

| **automatic query rewrite.** A process that examines an
| SQL statement that refers to one or more base tables,
| and, if appropriate, rewrites the query so that it
| performs better. This process can also determine
| whether to rewrite a query so that it refers to one or
| more materialized query tables that are derived from
| the source tables.

auxiliary index. An index on an auxiliary table in which each index entry refers to a LOB.

auxiliary table. A table that stores columns outside the table in which they are defined. Contrast with *base table*.

B

backout. The process of undoing uncommitted changes that an application process made. This might be necessary in the event of a failure on the part of an application process, or as a result of a deadlock situation.

backward log recovery. The fourth and final phase of restart processing during which DB2 scans the log in a backward direction to apply UNDO log records for all aborted changes.

base table. (1) A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with *result table* and *temporary table*.

(2) A table containing a LOB column definition. The actual LOB column data is not stored with the base table. The base table contains a row identifier for each row and an indicator column for each of its LOB columns. Contrast with *auxiliary table*.

base table space. A table space that contains base tables.

basic predicate. A predicate that compares two values.

basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential-access or a direct-access device.

| **batch message processing program.** In IMS, an
| application program that can perform batch-type
| processing online and can access the IMS input and
| output message queues.

before trigger. A trigger that is defined with the trigger activation time BEFORE.

binary integer. A basic data type that can be further classified as small integer or large integer.

binary large object (BLOB). A sequence of bytes in
which the size of the value ranges from 0 bytes to
2 GB−1. Such a string has a CCSID value of 65535.

binary string. A sequence of bytes that is not associated with a CCSID. For example, the BLOB data type is a binary string.

bind. The process by which the output from the SQL precompiler is converted to a usable control structure, often called an access plan, application plan, or package. During this process, access paths to the data are selected and some authorization checking is performed. The types of bind are:

automatic bind. (More correctly, *automatic rebind*) A process by which SQL statements are bound automatically (without a user issuing a BIND command) when an application process begins execution and the bound application plan or package it requires is not valid.
dynamic bind. A process by which SQL statements are bound as they are entered.
incremental bind. A process by which SQL statements are bound during the execution of an application process.
static bind. A process by which SQL statements are bound after they have been precompiled. All static SQL statements are prepared for execution at the same time.

bit data. Data that is character type CHAR or
VARCHAR and has a CCSID value of 65535.

BLOB. Binary large object.

block fetch. A capability in which DB2 can retrieve, or fetch, a large set of rows together. Using block fetch can significantly reduce the number of messages that are being sent across the network. Block fetch applies only to cursors that do not update data.

BMP. Batch Message Processing (IMS). See *batch message processing program*.

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.

BSAM. Basic sequential access method.

BSDS. Bootstrap data set.

buffer pool. Main storage that is reserved to satisfy the buffering requirements for one or more table spaces or indexes.

built-in data type. A data type that IBM supplies. Among the built-in data types for DB2 UDB for z/OS are string, numeric, ROWID, and datetime. Contrast with *distinct type*.

built-in function. A function that DB2 supplies. Contrast with *user-defined function*.

business dimension. A category of data, such as products or time periods, that an organization might want to analyze.

C

cache structure. A coupling facility structure that stores data that can be available to all members of a Sysplex. A DB2 data sharing group uses cache structures as group buffer pools.

CAF. Call attachment facility.

call attachment facility (CAF). A DB2 attachment facility for application programs that run in TSO or z/OS batch. The CAF is an alternative to the DSN command processor and provides greater control over the execution environment.

call-level interface (CLI). A callable application programming interface (API) for database access, which is an alternative to using embedded SQL. In contrast to embedded SQL, DB2 ODBC (which is based on the CLI architecture) does not require the user to precompile or bind applications, but instead provides a standard set of functions to process SQL statements and related services at run time.

cascade delete. The way in which DB2 enforces referential constraints when it deletes all descendent rows of a deleted parent row.

CASE expression. An expression that is selected based on the evaluation of one or more conditions.

cast function. A function that is used to convert instances of a (source) data type into instances of a different (target) data type. In general, a cast function has the name of the target data type. It has one single argument whose type is the source data type; its return type is the target data type.

castout. The DB2 process of writing changed pages from a group buffer pool to disk.

castout owner. The DB2 member that is responsible for casting out a particular page set or partition.

catalog. In DB2, a collection of tables that contains descriptions of objects such as tables, views, and indexes.

catalog table. Any table in the DB2 catalog.

CCSID. Coded character set identifier.

CDB. Communications database.

CDRA. Character Data Representation Architecture.

CEC. Central electronic complex. See *central processor complex*.

central electronic complex (CEC). See *central processor complex*.

central processor (CP). The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

central processor complex (CPC). A physical collection of hardware (such as an ES/3090™) that consists of main storage, one or more central processors, timers, and channels.

| **CFRM.** Coupling facility resource management.

CFRM policy. A declaration by a z/OS administrator regarding the allocation rules for a coupling facility structure.

character conversion. The process of changing characters from one encoding scheme to another.

Character Data Representation Architecture (CDRA). An architecture that is used to achieve consistent representation, processing, and interchange of string data.

character large object (CLOB). A sequence of bytes representing single-byte characters or a mixture of single- and double-byte characters where the size of the value can be up to 2 GB−1. In general, character large object values are used whenever a character string might exceed the limits of the VARCHAR type.

character set. A defined set of characters.

character string. A sequence of bytes that represent bit data, single-byte characters, or a mixture of single-byte and multibyte characters.

check constraint. A user-defined constraint that specifies the values that specific columns of a base table can contain.

check integrity. The condition that exists when each row in a table conforms to the check constraints that are defined on that table. Maintaining check integrity requires DB2 to enforce check constraints on operations that add or change data.

| **check pending.** A state of a table space or partition
| that prevents its use by some utilities and by some SQL
| statements because of rows that violate referential
| constraints, check constraints, or both.

checkpoint. A point at which DB2 records internal status information on the DB2 log; the recovery process uses this information if DB2 abnormally terminates.

| **child lock.** For explicit hierarchical locking, a lock that
| is held on either a table, page, row, or a large object
| (LOB). Each child lock has a parent lock. See also *parent*
| *lock*.

CI. Control interval.

| **CICS.** Represents (in this publication): CICS
| Transaction Server for z/OS: Customer Information
| Control System Transaction Server for z/OS.

CICS attachment facility. A DB2 subcomponent that uses the z/OS subsystem interface (SSI) and cross-storage linkage to process requests from CICS to DB2 and to coordinate resource commitment.

CIDF. Control interval definition field.

claim. A notification to DB2 that an object is being accessed. Claims prevent drains from occurring until the claim is released, which usually occurs at a commit point. Contrast with *drain*.

claim class. A specific type of object access that can be one of the following isolation levels:
Cursor stability (CS)
Repeatable read (RR)
Write

claim count. A count of the number of agents that are accessing an object.

class of service. A VTAM term for a list of routes through a network, arranged in an order of preference for their use.

class word. A single word that indicates the nature of a data attribute. For example, the class word PROJ indicates that the attribute identifies a project.

clause. In SQL, a distinct part of a statement, such as a SELECT clause or a WHERE clause.

CLI. Call- level interface.

client. See *requester*.

CLIST. Command list. A language for performing TSO tasks.

CLOB. Character large object.

closed application. An application that requires exclusive use of certain statements on certain DB2 objects, so that the objects are managed solely through the application's external interface.

CLPA. Create link pack area.

| **clustering index.** An index that determines how rows
| are physically ordered (*clustered*) in a table space. If a
| clustering index on a partitioned table is not a
| partitioning index, the rows are ordered in cluster
| sequence within each data partition instead of spanning
| partitions. Prior to Version 8 of DB2 UDB for z/OS, the
| partitioning index was required to be the clustering
| index.

coded character set. A set of unambiguous rules that establish a character set and the one-to-one relationships between the characters of the set and their coded representations.

coded character set identifier (CCSID). A 16-bit number that uniquely identifies a coded representation of graphic characters. It designates an encoding scheme identifier and one or more pairs consisting of a character set identifier and an associated code page identifier.

code page. (1) A set of assignments of characters to code points. In EBCDIC, for example, the character 'A' is assigned code point X'C1' (2) , and character 'B' is assigned code point X'C2'. Within a code page, each code point has only one specific meaning.

code point. In CDRA, a unique bit pattern that represents a character in a code page.

code unit. The fundamental binary width in a
computer architecture that is used for representing
character data, such as 7 bits, 8 bits, 16 bits, or 32 bits.
Depending on the character encoding form that is used,
each code point in a coded character set can be
represented internally by one or more code units.

coexistence. During migration, the period of time in which two releases exist in the same data sharing group.

cold start. A process by which DB2 restarts without processing any log records. Contrast with *warm start*.

collection. A group of packages that have the same qualifier.

column. The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer).

column function. See *aggregate function*.

"come from" checking. An LU 6.2 security option that defines a list of authorization IDs that are allowed to connect to DB2 from a partner LU.

command. A DB2 operator command or a DSN subcommand. A command is distinct from an SQL statement.

command prefix. A one- to eight-character command identifier. The command prefix distinguishes the command as belonging to an application or subsystem rather than to MVS.

command recognition character (CRC). A character that permits a z/OS console operator or an IMS subsystem user to route DB2 commands to specific DB2 subsystems.

command scope. The scope of command operation in a data sharing group. If a command has *member scope*, the command displays information only from the one member or affects only non-shared resources that are owned locally by that member. If a command has *group scope*, the command displays information from all members, affects non-shared resources that are owned locally by all members, displays information on sharable resources, or affects sharable resources.

commit. The operation that ends a unit of work by releasing locks so that the database changes that are made by that unit of work can be perceived by other processes.

commit point. A point in time when data is considered consistent.

committed phase. The second phase of the multisite update process that requests all participants to commit the effects of the logical unit of work.

common service area (CSA). In z/OS, a part of the common area that contains data areas that are addressable by all address spaces.

communications database (CDB). A set of tables in the DB2 catalog that are used to establish conversations with remote database management systems.

comparison operator. A token (such as =, >, or <) that is used to specify a relationship between two values.

composite key. An ordered set of key columns of the same table.

compression dictionary. The dictionary that controls the process of compression and decompression. This dictionary is created from the data in the table space or table space partition.

concurrency. The shared use of resources by more than one application process at the same time.

conditional restart. A DB2 restart that is directed by a user-defined conditional restart control record (CRCR).

connection. In SNA, the existence of a communication path between two partner LUs that allows information to be exchanged (for example, two DB2 subsystems that are connected and communicating by way of a conversation).

connection context. In SQLJ, a Java object that represents a connection to a data source.

connection declaration clause. In SQLJ, a statement that declares a connection to a data source.

connection handle. The data object containing information that is associated with a connection that DB2 ODBC manages. This includes general status information, transaction status, and diagnostic information.

connection ID. An identifier that is supplied by the attachment facility and that is associated with a specific address space connection.

consistency token. A timestamp that is used to generate the version identifier for an application. See also *version*.

constant. A language element that specifies an unchanging value. Constants are classified as string constants or numeric constants. Contrast with *variable*.

constraint. A rule that limits the values that can be inserted, deleted, or updated in a table. See *referential constraint*, *check constraint*, and *unique constraint*.

context. The application's logical connection to the data source and associated internal DB2 ODBC connection information that allows the application to direct its operations to a data source. A DB2 ODBC context represents a DB2 thread.

contracting conversion. A process that occurs when the length of a converted string is smaller than that of the source string. For example, this process occurs when an EBCDIC mixed-data string that contains DBCS characters is converted to ASCII mixed data; the converted string is shorter because of the removal of the shift codes.

control interval (CI). A fixed-length area or disk in which VSAM stores records and creates distributed free space. Also, in a key-sequenced data set or file, the set of records that an entry in the sequence-set index record points to. The control interval is the unit of information that VSAM transmits to or from disk. A control interval always includes an integral number of physical records.

control interval definition field (CIDF). In VSAM, a field that is located in the 4 bytes at the end of each control interval; it describes the free space, if any, in the control interval.

conversation. Communication, which is based on LU 6.2 or Advanced Program-to-Program Communication (APPC), between an application and a remote transaction program over an SNA logical unit-to-logical unit (LU-LU) session that allows communication while processing a transaction.

coordinator. The system component that coordinates the commit or rollback of a unit of work that includes work that is done on one or more other systems.

| **copy pool.** A named set of SMS storage groups that
| contains data that is to be copied collectively. A copy
| pool is an SMS construct that lets you define which
| storage groups are to be copied by using FlashCopy
| functions. HSM determines which volumes belong to a
| copy pool.

| **copy target.** A named set of SMS storage groups that
| are to be used as containers for copy pool volume
| copies. A copy target is an SMS construct that lets you
| define which storage groups are to be used as
| containers for volumes that are copied by using
| FlashCopy functions.

| **copy version.** A point-in-time FlashCopy copy that is
| managed by HSM. Each copy pool has a version
| parameter that specifies how many copy versions are
| maintained on disk.

correlated columns. A relationship between the value of one column and the value of another column.

correlated subquery. A subquery (part of a WHERE or HAVING clause) that is applied to a row or group of rows of a table or view that is named in an outer subselect statement.

correlation ID. An identifier that is associated with a specific thread. In TSO, it is either an authorization ID or the job name.

correlation name. An identifier that 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.

cost category. A category into which DB2 places cost estimates for SQL statements at the time the statement is bound. A cost estimate can be placed in either of the following cost categories:
- A: Indicates that DB2 had enough information to make a cost estimate without using default values.
- B: Indicates that some condition exists for which DB2 was forced to use default values for its estimate.

The cost category is externalized in the COST_CATEGORY column of the DSN_STATEMNT_TABLE when a statement is explained.

coupling facility. A special PR/SM™ LPAR logical partition that runs the coupling facility control program and provides high-speed caching, list processing, and locking functions in a Parallel Sysplex.

coupling facility resource management. A component of z/OS that provides the services to manage coupling facility resources in a Parallel Sysplex. This management includes the enforcement of CFRM policies to ensure that the coupling facility and structure requirements are satisfied.

CP. Central processor.

CPC. Central processor complex.

C++ member. A data object or function in a structure, union, or class.

C++ member function. An operator or function that is declared as a member of a class. A member function has access to the private and protected data members and to the member functions of objects in its class. Member functions are also called methods.

C++ object. (1) A region of storage. An object is created when a variable is defined or a new function is invoked. (2) An instance of a class.

CRC. Command recognition character.

CRCR. Conditional restart control record. See also *conditional restart*.

create link pack area (CLPA). An option that is used during IPL to initialize the link pack pageable area.

created temporary table. A table that holds temporary data and is defined with the SQL statement CREATE GLOBAL TEMPORARY TABLE. Information about created temporary tables is stored in the DB2 catalog, so this kind of table is persistent and can be shared across application processes. Contrast with *declared temporary table*. See also *temporary table*.

cross-memory linkage. A method for invoking a program in a different address space. The invocation is synchronous with respect to the caller.

cross-system coupling facility (XCF). A component of z/OS that provides functions to support cooperation between authorized programs that run within a Sysplex.

cross-system extended services (XES). A set of z/OS services that allow multiple instances of an application or subsystem, running on different systems in a Sysplex environment, to implement high-performance, high-availability data sharing by using a coupling facility.

CS. Cursor stability.

CSA. Common service area.

CT. Cursor table.

current data. Data within a host structure that is current with (identical to) the data within the base table.

current SQL ID. An ID that, at a single point in time, holds the privileges that are exercised when certain dynamic SQL statements run. The current SQL ID can be a primary authorization ID or a secondary authorization ID.

current status rebuild. The second phase of restart processing during which the status of the subsystem is reconstructed from information on the log.

cursor. A named control structure that an application program uses to point to a single row or multiple rows within some ordered set of rows of a result table. A cursor can be used to retrieve, update, or delete rows from a result table.

cursor sensitivity. The degree to which database updates are visible to the subsequent FETCH statements in a cursor. A cursor can be sensitive to changes that are made with positioned update and delete statements specifying the name of that cursor. A cursor can also be sensitive to changes that are made with searched update or delete statements, or with cursors other than this cursor. These changes can be made by this application process or by another application process.

cursor stability (CS). The isolation level that provides maximum concurrency without the ability to read uncommitted data. With cursor stability, a unit of work holds locks only on its uncommitted changes and on the current row of each of its cursors.

cursor table (CT). The copy of the skeleton cursor table that is used by an executing application process.

cycle. A set of tables that can be ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table. A self-referencing table is a cycle with a single member.

D

DAD. See *Document access definition.*

disk. A direct-access storage device that records data magnetically.

database. A collection of tables, or a collection of table spaces and index spaces.

database access thread. A thread that accesses data at the local subsystem on behalf of a remote subsystem.

database administrator (DBA). An individual who is responsible for designing, developing, operating, safeguarding, maintaining, and using a database.

database alias. The name of the target server if different from the location name. The database alias name is used to provide the name of the database server as it is known to the network. When a database alias name is defined, the location name is used by the application to reference the server, but the database alias name is used to identify the database server to be accessed. Any fully qualified object names within any SQL statements are not modified and are sent unchanged to the database server.

database descriptor (DBD). An internal representation of a DB2 database definition, which reflects the data definition that is in the DB2 catalog. The objects that are defined in a database descriptor are table spaces, tables, indexes, index spaces, relationships, check constraints, and triggers. A DBD also contains information about accessing tables in the database.

database exception status. An indication that something is wrong with a database. All members of a data sharing group must know and share the exception status of databases.

database identifier (DBID). An internal identifier of the database.

database management system (DBMS). A software system that controls the creation, organization, and modification of a database and the access to the data that is stored within it.

database request module (DBRM). A data set member that is created by the DB2 precompiler and that contains information about SQL statements. DBRMs are used in the bind process.

database server. The target of a request from a local application or an intermediate database server. In the DB2 environment, the database server function is provided by the distributed data facility to access DB2 data from local applications, or from a remote database server that acts as an intermediate database server.

data currency. The state in which data that is retrieved into a host variable in your program is a copy of data in the base table.

data definition name (ddname). The name of a data definition (DD) statement that corresponds to a data control block containing the same name.

data dictionary. A repository of information about an organization's application programs, databases, logical data models, users, and authorizations. A data dictionary can be manual or automated.

data-driven business rules. Constraints on particular data values that exist as a result of requirements of the business.

Data Language/I (DL/I). The IMS data manipulation language; a common high-level interface between a user application and IMS.

data mart. A small data warehouse that applies to a single department or team. See also *data warehouse*.

data mining. The process of collecting critical business information from a data warehouse, correlating it, and uncovering associations, patterns, and trends.

data partition. A VSAM data set that is contained within a partitioned table space.

data-partitioned secondary index (DPSI). A secondary index that is partitioned. The index is partitioned according to the underlying data.

data sharing. The ability of two or more DB2 subsystems to directly access and change a single set of data.

data sharing group. A collection of one or more DB2 subsystems that directly access and change the same data while maintaining data integrity.

data sharing member. A DB2 subsystem that is assigned by XCF services to a data sharing group.

data source. A local or remote relational or non-relational data manager that is capable of supporting data access via an ODBC driver that supports the ODBC APIs. In the case of DB2 UDB for z/OS, the data sources are always relational database managers.

data space. In releases prior to DB2 UDB for z/OS, Version 8, a range of up to 2 GB of contiguous virtual storage addresses that a program can directly manipulate. Unlike an address space, a data space can hold only data; it does not contain common areas, system data, or programs.

data type. An attribute of columns, literals, host variables, special registers, and the results of functions and expressions.

data warehouse. A system that provides critical business information to an organization. The data warehouse system cleanses the data for accuracy and currency, and then presents the data to decision makers so that they can interpret and use it effectively and efficiently.

date. A three-part value that designates a day, month, and year.

date duration. A decimal integer that represents a number of years, months, and days.

datetime value. A value of the data type DATE, TIME, or TIMESTAMP.

DBA. Database administrator.

DBCLOB. Double-byte character large object.

DBCS. Double-byte character set.

DBD. Database descriptor.

DBID. Database identifier.

DBMS. Database management system.

DBRM. Database request module.

DB2 catalog. Tables that are maintained by DB2 and contain descriptions of DB2 objects, such as tables, views, and indexes.

DB2 command. An instruction to the DB2 subsystem that a user enters to start or stop DB2, to display information on current users, to start or stop databases, to display information on the status of databases, and so on.

DB2 for VSE & VM. The IBM DB2 relational database management system for the VSE and VM operating systems.

DB2I. DB2 Interactive.

DB2 Interactive (DB2I). The DB2 facility that provides for the execution of SQL statements, DB2 (operator) commands, programmer commands, and utility invocation.

DB2I Kanji Feature. The tape that contains the panels and jobs that allow a site to display DB2I panels in Kanji.

DB2 PM. DB2 Performance Monitor.

DB2 thread. The DB2 structure that describes an application’s connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services.

DCLGEN. Declarations generator.

DDF. Distributed data facility.

ddname. Data definition name.

deadlock. Unresolvable contention for the use of a resource, such as a table or an index.

declarations generator (DCLGEN). A subcomponent of DB2 that generates SQL table declarations and COBOL, C, or PL/I data structure declarations that conform to the table. The declarations are generated from DB2 system catalog information. DCLGEN is also a DSN subcommand.

declared temporary table. A table that holds temporary data and is defined with the SQL statement DECLARE GLOBAL TEMPORARY TABLE. Information about declared temporary tables is not stored in the DB2 catalog, so this kind of table is not persistent and can be used only by the application process that issued the DECLARE statement. Contrast with *created temporary table*. See also *temporary table*.

default value. A predetermined value, attribute, or option that is assumed when no other is explicitly specified.

deferred embedded SQL. SQL statements that are neither fully static nor fully dynamic. Like static statements, they are embedded within an application, but like dynamic statements, they are prepared during the execution of the application.

deferred write. The process of asynchronously writing changed data pages to disk.

degree of parallelism. The number of concurrently executed operations that are initiated to process a query.

delete-connected. A table that is a dependent of table P or a dependent of a table to which delete operations from table P cascade.

delete hole. The location on which a cursor is positioned when a row in a result table is refetched and the row no longer exists on the base table, because another cursor deleted the row between the time the cursor first included the row in the result table and the time the cursor tried to refetch it.

delete rule. The rule that tells DB2 what to do to a dependent row when a parent row is deleted. For each relationship, the rule might be CASCADE, RESTRICT, SET NULL, or NO ACTION.

delete trigger. A trigger that is defined with the triggering SQL operation DELETE.

delimited identifier. A sequence of characters that are enclosed within double quotation marks ("). The sequence must consist of a letter followed by zero or more characters, each of which is a letter, digit, or the underscore character (_).

delimiter token. A string constant, a delimited identifier, an operator symbol, or any of the special characters that are shown in DB2 syntax diagrams.

denormalization. A key step in the task of building a physical relational database design. Denormalization is the intentional duplication of columns in multiple tables, and the consequence is increased data redundancy. Denormalization is sometimes necessary to minimize performance problems. Contrast with *normalization*.

dependent. An object (row, table, or table space) that has at least one parent. The object is also said to be a dependent (row, table, or table space) of its parent. See also *parent row*, *parent table*, *parent table space*.

dependent row. A row that contains a foreign key that matches the value of a primary key in the parent row.

dependent table. A table that is a dependent in at least one referential constraint.

DES-based authenticator. An authenticator that is generated using the DES algorithm.

descendent. An object that is a dependent of an object or is the dependent of a descendent of an object.

descendent row. A row that is dependent on another row, or a row that is a descendent of a dependent row.

descendent table. A table that is a dependent of another table, or a table that is a descendent of a dependent table.

deterministic function. A user-defined function whose result is dependent on the values of the input arguments. That is, successive invocations with the same input values produce the same answer. Sometimes referred to as a *not-variant* function. Contrast this with an *nondeterministic function* (sometimes called a *variant function*), which might not always produce the same result for the same inputs.

DFP. Data Facility Product (in z/OS).

DFSMS. Data Facility Storage Management Subsystem (in z/OS). Also called *Storage Management Subsystem (SMS)*.

| **DFSMSdss™.** The data set services (dss) component of
| DFSMS (in z/OS).

| **DFSMShsm™.** The hierarchical storage manager (hsm)
| component of DFSMS (in z/OS).

dimension. A data category such as time, products, or markets. The elements of a dimension are referred to as members. Dimensions offer a very concise, intuitive way of organizing and selecting data for retrieval, exploration, and analysis. See also *dimension table*.

dimension table. The representation of a dimension in a star schema. Each row in a dimension table represents all of the attributes for a particular member of the dimension. See also *dimension*, *star schema*, and *star join*.

directory. The DB2 system database that contains internal objects such as database descriptors and skeleton cursor tables.

distinct predicate. In SQL, a predicate that ensures
that two row values are not equal, and that both row
values are not null.

distinct type. A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

distributed data. Data that resides on a DBMS other than the local system.

distributed data facility (DDF). A set of DB2 components through which DB2 communicates with another relational database management system.

Distributed Relational Database Architecture (DRDA). A connection protocol for distributed relational database processing that is used by IBM's relational database products. DRDA includes protocols for communication between an application and a remote relational database management system, and for communication between relational database management systems. See also *DRDA access.*

DL/I. Data Language/I.

DNS. Domain name server.

| **document access definition (DAD).** Used to define
| the indexing scheme for an XML column or the
| mapping scheme of an XML collection. It can be used
| to enable an XML Extender column of an XML
| collection, which is XML formatted.

domain. The set of valid values for an attribute.

domain name. The name by which TCP/IP applications refer to a TCP/IP host within a TCP/IP network.

domain name server (DNS). A special TCP/IP network server that manages a distributed directory that is used to map TCP/IP host names to IP addresses.

double-byte character large object (DBCLOB). A sequence of bytes representing double-byte characters where the size of the values can be up to 2 GB. In general, DBCLOB values are used whenever a double-byte character string might exceed the limits of the VARGRAPHIC type.

double-byte character set (DBCS). A set of characters, which are used by national languages such as Japanese and Chinese, that have more symbols than can be represented by a single byte. Each character is 2 bytes in length. Contrast with *single-byte character set* and *multibyte character set*.

double-precision floating point number. A 64-bit approximate representation of a real number.

downstream. The set of nodes in the syncpoint tree that is connected to the local DBMS as a participant in the execution of a two-phase commit.

| **DPSI.** Data-partitioned secondary index.

drain. The act of acquiring a locked resource by quiescing access to that object.

drain lock. A lock on a claim class that prevents a claim from occurring.

DRDA. Distributed Relational Database Architecture.

DRDA access. An open method of accessing distributed data that you can use to can connect to another database server to execute packages that were previously bound at the server location. You use the SQL CONNECT statement or an SQL statement with a three-part name to identify the server. Contrast with *private protocol access*.

DSN. (1) The default DB2 subsystem name. (2) The name of the TSO command processor of DB2. (3) The first three characters of DB2 module and macro names.

duration. A number that represents an interval of time. See also *date duration*, *labeled duration*, and *time duration*.

| **dynamic cursor.** A named control structure that an
| application program uses to change the size of the
| result table and the order of its rows after the cursor is
| opened. Contrast with *static cursor.*

dynamic dump. A dump that is issued during the execution of a program, usually under the control of that program.

dynamic SQL. SQL statements that are prepared and executed within an application program while the program is executing. In dynamic SQL, the SQL source is contained in host language variables rather than being coded into the application program. The SQL statement can change several times during the application program's execution.

| **dynamic statement cache pool.** A cache, located above
| the 2-GB storage line, that holds dynamic statements.

E

EA-enabled table space. A table space or index space that is enabled for extended addressability and that contains individual partitions (or pieces, for LOB table spaces) that are greater than 4 GB.

| **EB.** See *exabyte*.

EBCDIC. Extended binary coded decimal interchange code. An encoding scheme that is used to represent character data in the z/OS, VM, VSE, and iSeries™ environments. Contrast with *ASCII* and *Unicode*.

e-business. The transformation of key business processes through the use of Internet technologies.

| **EDM pool.** A pool of main storage that is used for
| database descriptors, application plans, authorization
| cache, application packages.

EID. Event identifier.

embedded SQL. SQL statements that are coded within an application program. See *static SQL*.

enclave. In Language Environment , an independent collection of routines, one of which is designated as the main routine. An enclave is similar to a program or run unit.

encoding scheme. A set of rules to represent character data (ASCII, EBCDIC, or Unicode).

entity. A significant object of interest to an organization.

enumerated list. A set of DB2 objects that are defined with a LISTDEF utility control statement in which pattern-matching characters (*, %, _ or ?) are not used.

environment. A collection of names of logical and physical resources that are used to support the performance of a function.

environment handle. In DB2 ODBC, the data object that contains global information regarding the state of the application. An environment handle must be allocated before a connection handle can be allocated. Only one environment handle can be allocated per application.

EOM. End of memory.

EOT. End of task.

equijoin. A join operation in which the join-condition has the form *expression = expression*.

error page range. A range of pages that are considered to be physically damaged. DB2 does not allow users to access any pages that fall within this range.

escape character. The symbol that is used to enclose an SQL delimited identifier. The escape character is the double quotation mark ("), except in COBOL applications, where the user assigns the symbol, which is either a double quotation mark or an apostrophe (').

ESDS. Entry sequenced data set.

ESMT. External subsystem module table (in IMS).

EUR. IBM European Standards.

| **exabyte.** For processor, real and virtual storage
| capacities and channel volume:
| 1 152 921 504 606 846 976 bytes or 2^{60}.

exception table. A table that holds rows that violate referential constraints or check constraints that the CHECK DATA utility finds.

exclusive lock. A lock that prevents concurrently executing application processes from reading or changing data. Contrast with *share lock*.

executable statement. An SQL statement that can be embedded in an application program, dynamically prepared and executed, or issued interactively.

execution context. In SQLJ, a Java object that can be used to control the execution of SQL statements.

exit routine. A user-written (or IBM-provided default) program that receives control from DB2 to perform specific functions. Exit routines run as extensions of DB2.

expanding conversion. A process that occurs when the length of a converted string is greater than that of the source string. For example, this process occurs when an ASCII mixed-data string that contains DBCS characters is converted to an EBCDIC mixed-data string; the converted string is longer because of the addition of shift codes.

explicit hierarchical locking. Locking that is used to make the parent-child relationship between resources known to IRLM. This kind of locking avoids global locking overhead when no inter-DB2 interest exists on a resource.

exposed name. A correlation name or a table or view name for which a correlation name is not specified. Names that are specified in a FROM clause are exposed or non-exposed.

expression. An operand or a collection of operators and operands that yields a single value.

extended recovery facility (XRF). A facility that minimizes the effect of failures in z/OS, VTAM , the host processor, or high-availability applications during sessions between high-availability applications and designated terminals. This facility provides an alternative subsystem to take over sessions from the failing subsystem.

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that is a subset of Standardized General Markup Language (SGML). The less complex nature of XML makes it easier to write applications that handle document types, to author and manage structured information, and to transmit and share structured information across diverse computing environments.

external function. A function for which the body is written in a programming language that takes scalar argument values and produces a scalar result for each invocation. Contrast with *sourced function*, *built-in function*, and *SQL function*.

external procedure. A user-written application program that can be invoked with the SQL CALL statement, which is written in a programming language. Contrast with *SQL procedure*.

external routine. A user-defined function or stored procedure that is based on code that is written in an external programming language.

external subsystem module table (ESMT). In IMS, the table that specifies which attachment modules must be loaded.

F

failed member state. A state of a member of a data sharing group. When a member fails, the XCF permanently records the failed member state. This state usually means that the member's task, address space, or z/OS system terminated before the state changed from active to quiesced.

fallback. The process of returning to a previous release of DB2 after attempting or completing migration to a current release.

false global lock contention. A contention indication from the coupling facility when multiple lock names are hashed to the same indicator and when no real contention exists.

fan set. A direct physical access path to data, which is provided by an index, hash, or link; a fan set is the means by which the data manager supports the ordering of data.

federated database. The combination of a DB2 Universal Database server (in Linux, UNIX, and Windows environments) and multiple data sources to which the server sends queries. In a federated database system, a client application can use a single SQL statement to join data that is distributed across multiple database management systems and can view the data as if it were local.

fetch orientation. The specification of the desired placement of the cursor as part of a FETCH statement (for example, BEFORE, AFTER, NEXT, PRIOR, CURRENT, FIRST, LAST, ABSOLUTE, and RELATIVE).

field procedure. A user-written exit routine that is designed to receive a single value and transform (encode or decode) it in any way the user can specify.

filter factor. A number between zero and one that estimates the proportion of rows in a table for which a predicate is true.

fixed-length string. A character or graphic string whose length is specified and cannot be changed. Contrast with *varying-length string*.

FlashCopy. A function on the IBM Enterprise Storage Server® that can create a point-in-time copy of data while an application is running.

foreign key. A column or set of columns in a dependent table of a constraint relationship. The key must have the same number of columns, with the same descriptions, as the primary key of the parent table.

Each foreign key value must either match a parent key value in the related parent table or be null.

| **forest.** An ordered set of subtrees of XML nodes.

forget. In a two-phase commit operation, (1) the vote that is sent to the prepare phase when the participant has not modified any data. The forget vote allows a participant to release locks and forget about the logical unit of work. This is also referred to as the read-only vote. (2) The response to the *committed* request in the second phase of the operation.

forward log recovery. The third phase of restart processing during which DB2 processes the log in a forward direction to apply all REDO log records.

free space. The total amount of unused space in a page; that is, the space that is not used to store records or control information is free space.

full outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of both tables. See also *join*.

fullselect. A subselect, a values-clause, or a number of both that are combined by set operators. *Fullselect* specifies a result table. If UNION is not used, the result of the fullselect is the result of the specified subselect.

| **fully escaped mapping.** A mapping from an SQL
| identifier to an XML name when the SQL identifier is a
| column name.

function. A mapping, which is embodied as a
program (the function body) that is invocable by means
# of zero or more input values (arguments) to a single
value (the result). See also *aggregate function* and *scalar*
function.

Functions can be user-defined, built-in, or generated by
DB2. (See also *built-in function*, *cast function*, *external*
function, *sourced function*, *SQL function*, and *user-defined*
function.)

function definer. The authorization ID of the owner of the schema of the function that is specified in the CREATE FUNCTION statement.

function implementer. The authorization ID of the owner of the function program and function package.

function package. A package that results from binding the DBRM for a function program.

function package owner. The authorization ID of the user who binds the function program's DBRM into a function package.

function resolution. The process, internal to the DBMS, by which a function invocation is bound to a particular function instance. This process uses the function name, the data types of the arguments, and a list of the applicable schema names (called the *SQL path*) to make the selection. This process is sometimes called *function selection*.

function selection. See *function resolution*.

function signature. The logical concatenation of a fully qualified function name with the data types of all of its parameters.

G

GB. Gigabyte (1 073 741 824 bytes).

GBP. Group buffer pool.

GBP-dependent. The status of a page set or page set partition that is dependent on the group buffer pool. Either read/write interest is active among DB2 subsystems for this page set, or the page set has changed pages in the group buffer pool that have not yet been cast out to disk.

generalized trace facility (GTF). A z/OS service program that records significant system events such as I/O interrupts, SVC interrupts, program interrupts, or external interrupts.

generic resource name. A name that VTAM uses to represent several application programs that provide the same function in order to handle session distribution and balancing in a Sysplex environment.

getpage. An operation in which DB2 accesses a data page.

global lock. A lock that provides concurrency control within and among DB2 subsystems. The scope of the lock is across all DB2 subsystems of a data sharing group.

global lock contention. Conflicts on locking requests between different DB2 members of a data sharing group when those members are trying to serialize shared resources.

governor. See *resource limit facility*.

graphic string. A sequence of DBCS characters.

gross lock. The *shared*, *update*, or *exclusive* mode locks on a table, partition, or table space.

group buffer pool (GBP). A coupling facility cache structure that is used by a data sharing group to cache data and to ensure that the data is consistent for all members.

group buffer pool duplexing. The ability to write data to two instances of a group buffer pool structure: a *primary group buffer pool* and a *secondary group buffer* *pool*. z/OS publications refer to these instances as the "old" (for primary) and "new" (for secondary) structures.

group level. The release level of a data sharing group, which is established when the first member migrates to a new release.

group name. The z/OS XCF identifier for a data sharing group.

group restart. A restart of at least one member of a data sharing group after the loss of either locks or the shared communications area.

GTF. Generalized trace facility.

H

handle. In DB2 ODBC, a variable that refers to a data structure and associated resources. See also *statement handle*, *connection handle*, and *environment handle*.

help panel. A screen of information that presents tutorial text to assist a user at the workstation or terminal.

heuristic damage. The inconsistency in data between one or more participants that results when a heuristic decision to resolve an indoubt LUW at one or more participants differs from the decision that is recorded at the coordinator.

heuristic decision. A decision that forces indoubt resolution at a participant by means other than automatic resynchronization between coordinator and participant.

| **hole.** A row of the result table that cannot be accessed because of a delete or an update that has been performed on the row. See also *delete hole* and *update hole*.

home address space. The area of storage that z/OS currently recognizes as *dispatched*.

host. The set of programs and resources that are available on a given TCP/IP instance.

host expression. A Java variable or expression that is referenced by SQL clauses in an SQLJ application program.

host identifier. A name that is declared in the host program.

host language. A programming language in which you can embed SQL statements.

host program. An application program that is written in a host language and that contains embedded SQL statements.

host structure. In an application program, a structure that is referenced by embedded SQL statements.

host variable. In an application program, an application variable that is referenced by embedded SQL statements.

| **host variable array.** An array of elements, each of which corresponds to a value for a column. The dimension of the array determines the maximum number of rows for which the array can be used.

HSM. Hierarchical storage manager.

HTML. Hypertext Markup Language, a standard method for presenting Web data to users.

HTTP. Hypertext Transfer Protocol, a communication protocol that the Web uses.

I

ICF. Integrated catalog facility.

IDCAMS. An IBM program that is used to process access method services commands. It can be invoked as a job or jobstep, from a TSO terminal, or from within a user's application program.

IDCAMS LISTCAT. A facility for obtaining information that is contained in the access method services catalog.

identify. A request that an attachment service program in an address space that is separate from DB2 issues thorough the z/OS subsystem interface to inform DB2 of its existence and to initiate the process of becoming connected to DB2.

identity column. A column that provides a way for DB2 to automatically generate a numeric value for each row. The generated values are unique if cycling is not used. Identity columns are defined with the AS IDENTITY clause. Uniqueness of values can be ensured by defining a unique index that contains only the identity column. A table can have no more than one identity column.

IFCID. Instrumentation facility component identifier.

IFI. Instrumentation facility interface.

IFI call. An invocation of the instrumentation facility interface (IFI) by means of one of its defined functions.

IFP. IMS Fast Path.

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).

implied forget. In the presumed-abort protocol, an implied response of *forget* to the second-phase *committed* request from the coordinator. The response is implied when the participant responds to any subsequent request from the coordinator.

IMS. Information Management System.

IMS attachment facility. A DB2 subcomponent that uses z/OS subsystem interface (SSI) protocols and cross-memory linkage to process requests from IMS to DB2 and to coordinate resource commitment.

IMS DB. Information Management System Database.

IMS TM. Information Management System Transaction Manager.

in-abort. A status of a unit of recovery. If DB2 fails after a unit of recovery begins to be rolled back, but before the process is completed, DB2 continues to back out the changes during restart.

in-commit. A status of a unit of recovery. If DB2 fails after beginning its phase 2 commit processing, it "knows," when restarted, that changes made to data are consistent. Such units of recovery are termed *in-commit*.

independent. An object (row, table, or table space) that is neither a parent nor a dependent of another object.

index. A set of pointers that are logically ordered by the values of a key. Indexes can provide faster access to data and can enforce uniqueness on the rows in a table.

| **index-controlled partitioning.** A type of partitioning
| in which partition boundaries for a partitioned table are
| controlled by values that are specified on the CREATE
| INDEX statement. Partition limits are saved in the
| LIMITKEY column of the SYSIBM.SYSINDEXPART
| catalog table.

index key. The set of columns in a table that is used to determine the order of index entries.

index partition. A VSAM data set that is contained within a partitioning index space.

index space. A page set that is used to store the entries of one index.

indicator column. A 4-byte value that is stored in a base table in place of a LOB column.

indicator variable. A variable that is used to represent the null value in an application program. If the value for the selected column is null, a negative value is placed in the indicator variable.

indoubt. A status of a unit of recovery. If DB2 fails after it has finished its phase 1 commit processing and before it has started phase 2, only the commit coordinator knows if an individual unit of recovery is to be committed or rolled back. At emergency restart, if DB2 lacks the information it needs to make this decision, the status of the unit of recovery is *indoubt* until DB2 obtains this information from the coordinator. More than one unit of recovery can be indoubt at restart.

indoubt resolution. The process of resolving the status of an indoubt logical unit of work to either the committed or the rollback state.

inflight. A status of a unit of recovery. If DB2 fails before its unit of recovery completes phase 1 of the commit process, it merely backs out the updates of its unit of recovery at restart. These units of recovery are termed *inflight*.

inheritance. The passing downstream of class resources or attributes from a parent class in the class hierarchy to a child class.

initialization file. For DB2 ODBC applications, a file containing values that can be set to adjust the performance of the database manager.

inline copy. A copy that is produced by the LOAD or REORG utility. The data set that the inline copy produces is logically equivalent to a full image copy that is produced by running the COPY utility with read-only access (SHRLEVEL REFERENCE).

inner join. The result of a join operation that includes only the matched rows of both tables that are being joined. See also *join*.

inoperative package. A package that cannot be used because one or more user-defined functions or procedures that the package depends on were dropped. Such a package must be explicitly rebound. Contrast with *invalid package.*

| **insensitive cursor.** A cursor that is not sensitive to
| inserts, updates, or deletes that are made to the
| underlying rows of a result table after the result table
| has been materialized.

insert trigger. A trigger that is defined with the triggering SQL operation INSERT.

install. The process of preparing a DB2 subsystem to operate as a z/OS subsystem.

installation verification scenario. A sequence of operations that exercises the main DB2 functions and tests whether DB2 was correctly installed.

instrumentation facility component identifier (IFCID). A value that names and identifies a trace record of an event that can be traced. As a parameter on the START TRACE and MODIFY TRACE commands, it specifies that the corresponding event is to be traced.

instrumentation facility interface (IFI). A programming interface that enables programs to obtain online trace data about DB2, to submit DB2 commands, and to pass data to DB2.

Interactive System Productivity Facility (ISPF). An IBM licensed program that provides interactive dialog services in a z/OS environment.

inter-DB2 R/W interest. A property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members.

intermediate database server. The target of a request from a local application or a remote application requester that is forwarded to another database server. In the DB2 environment, the remote request is forwarded transparently to another database server if the object that is referenced by a three-part name does not reference the local location.

internationalization. The support for an encoding scheme that is able to represent the code points of characters from many different geographies and languages. To support all geographies, the Unicode standard requires more than 1 byte to represent a single character. See also *Unicode*.

internal resource lock manager (IRLM). A z/OS subsystem that DB2 uses to control communication and database locking.

| **International Organization for Standardization.** An
| international body charged with creating standards to
| facilitate the exchange of goods and services as well as
| cooperation in intellectual, scientific, technological, and
| economic activity.

invalid package. A package that depends on an object (other than a user-defined function) that is dropped. Such a package is implicitly rebound on invocation. Contrast with *inoperative package.*

invariant character set. (1) A character set, such as the syntactic character set, whose code point assignments do not change from code page to code page. (2) A minimum set of characters that is available as part of all character sets.

IP address. A 4-byte value that uniquely identifies a TCP/IP host.

IRLM. Internal resource lock manager.

ISO. International Organization for Standardization.

isolation level. The degree to which a unit of work is isolated from the updating operations of other units of work. See also *cursor stability*, *read stability*, *repeatable read*, and *uncommitted read*.

ISPF. Interactive System Productivity Facility.

ISPF/PDF. Interactive System Productivity Facility/Program Development Facility.

iterator. In SQLJ, an object that contains the result set of a query. An iterator is equivalent to a cursor in other host languages.

iterator declaration clause. In SQLJ, a statement that generates an iterator declaration class. An iterator is an object of an iterator declaration class.

J

| **Japanese Industrial Standard.** An encoding scheme
| that is used to process Japanese characters.

| **JAR.** Java Archive.

Java Archive (JAR). A file format that is used for aggregating many files into a single file.

JCL. Job control language.

JDBC. A Sun Microsystems database application programming interface (API) for Java that allows programs to access database management systems by using callable SQL. JDBC does not require the use of an SQL preprocessor. In addition, JDBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time.

JES. Job Entry Subsystem.

JIS. Japanese Industrial Standard.

job control language (JCL). A control language that is used to identify a job to an operating system and to describe the job's requirements.

Job Entry Subsystem (JES). An IBM licensed program that receives jobs into the system and processes all output data that is produced by the jobs.

join. A relational operation that allows retrieval of data from two or more tables based on matching column values. See also *equijoin, full outer join, inner join, left outer join, outer join, and right outer join*.

K

KB. Kilobyte (1024 bytes).

Kerberos. A network authentication protocol that is designed to provide strong authentication for client/server applications by using secret-key cryptography.

Kerberos ticket. A transparent application mechanism that transmits the identity of an initiating principal to its target. A simple ticket contains the principal's identity, a session key, a timestamp, and other information, which is sealed using the target's secret key.

key. A column or an ordered collection of columns that is identified in the description of a table, index, or referential constraint. The same column can be part of more than one key.

key-sequenced data set (KSDS). A VSAM file or data set whose records are loaded in key sequence and controlled by an index.

keyword. In SQL, a name that identifies an option that is used in an SQL statement.

KSDS. Key-sequenced data set.

L

labeled duration. A number that represents a duration of years, months, days, hours, minutes, seconds, or microseconds.

large object (LOB). A sequence of bytes representing bit data, single-byte characters, double-byte characters, or a mixture of single- and double-byte characters. A LOB can be up to 2 GB−1 byte in length. See also *BLOB*, *CLOB*, and *DBCLOB*.

last agent optimization. An optimized commit flow for either presumed-nothing or presumed-abort protocols in which the last agent, or final participant, becomes the commit coordinator. This flow saves at least one message.

latch. A DB2 internal mechanism for controlling concurrent events or the use of system resources.

LCID. Log control interval definition.

LDS. Linear data set.

leaf page. A page that contains pairs of keys and RIDs and that points to actual data. Contrast with *nonleaf page*.

left outer join. The result of a join operation that includes the matched rows of both tables that are being joined, and that preserves the unmatched rows of the first table. See also *join*.

limit key. The highest value of the index key for a partition.

linear data set (LDS). A VSAM data set that contains data but no control information. A linear data set can be accessed as a byte-addressable string in virtual storage.

linkage editor. A computer program for creating load modules from one or more object modules or load modules by resolving cross references among the modules and, if necessary, adjusting addresses.

link-edit. The action of creating a loadable computer program using a linkage editor.

list. A type of object, which DB2 utilities can process, that identifies multiple table spaces, multiple index spaces, or both. A list is defined with the LISTDEF utility control statement.

list structure. A coupling facility structure that lets data be shared and manipulated as elements of a queue.

LLE. Load list element.

L-lock. Logical lock.

| **load list element.** A z/OS control block that controls
| the loading and deleting of a particular load module
| based on entry point names.

load module. A program unit that is suitable for loading into main storage for execution. The output of a linkage editor.

LOB. Large object.

LOB locator. A mechanism that allows an application program to manipulate a large object value in the database system. A LOB locator is a fullword integer value that represents a single LOB value. An application program retrieves a LOB locator into a host variable and can then apply SQL operations to the associated LOB value using the locator.

LOB lock. A lock on a LOB value.

LOB table space. A table space in an auxiliary table that contains all the data for a particular LOB column in the related base table.

local. A way of referring to any object that the local DB2 subsystem maintains. A *local table*, for example, is a table that is maintained by the local DB2 subsystem. Contrast with *remote*.

locale. The definition of a subset of a user's environment that combines a CCSID and characters that are defined for a specific language and country.

local lock. A lock that provides intra-DB2 concurrency control, but not inter-DB2 concurrency control; that is, its scope is a single DB2.

local subsystem. The unique relational DBMS to which the user or application program is directly connected (in the case of DB2, by one of the DB2 attachment facilities).

| **location.** The unique name of a database server. An
| application uses the location name to access a DB2

database server. A database alias can be used to override the location name when accessing a remote server.

location alias. Another name by which a database server identifies itself in the network. Applications can use this name to access a DB2 database server.

lock. A means of controlling concurrent events or access to data. DB2 locking is performed by the IRLM.

lock duration. The interval over which a DB2 lock is held.

lock escalation. The promotion of a lock from a row, page, or LOB lock to a table space lock because the number of page locks that are concurrently held on a given resource exceeds a preset limit.

locking. The process by which the integrity of data is ensured. Locking prevents concurrent users from accessing inconsistent data.

lock mode. A representation for the type of access that concurrently running programs can have to a resource that a DB2 lock is holding.

lock object. The resource that is controlled by a DB2 lock.

lock promotion. The process of changing the size or mode of a DB2 lock to a higher, more restrictive level.

lock size. The amount of data that is controlled by a DB2 lock on table data; the value can be a row, a page, a LOB, a partition, a table, or a table space.

lock structure. A coupling facility data structure that is composed of a series of lock entries to support shared and exclusive locking for logical resources.

log. A collection of records that describe the events that occur during DB2 execution and that indicate their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

log control interval definition. A suffix of the physical log record that tells how record segments are placed in the physical control interval.

logical claim. A claim on a logical partition of a nonpartitioning index.

logical data modeling. The process of documenting the comprehensive business information requirements in an accurate and consistent format. Data modeling is the first task of designing a database.

logical drain. A drain on a logical partition of a nonpartitioning index.

logical index partition. The set of all keys that reference the same data partition.

logical lock (L-lock). The lock type that transactions use to control intra- and inter-DB2 data concurrency between transactions. Contrast with *physical lock (P-lock)*.

logically complete. A state in which the concurrent copy process is finished with the initialization of the target objects that are being copied. The target objects are available for update.

logical page list (LPL). A list of pages that are in error and that cannot be referenced by applications until the pages are recovered. The page is in *logical error* because the actual media (coupling facility or disk) might not contain any errors. Usually a connection to the media has been lost.

logical partition. A set of key or RID pairs in a nonpartitioning index that are associated with a particular partition.

logical recovery pending (LRECP). The state in which the data and the index keys that reference the data are inconsistent.

logical unit (LU). An access point through which an application program accesses the SNA network in order to communicate with another application program.

logical unit of work (LUW). The processing that a program performs between synchronization points.

logical unit of work identifier (LUWID). A name that uniquely identifies a thread within a network. This name consists of a fully-qualified LU network name, an LUW instance number, and an LUW sequence number.

log initialization. The first phase of restart processing during which DB2 attempts to locate the current end of the log.

log record header (LRH). A prefix, in every logical record, that contains control information.

log record sequence number (LRSN). A unique identifier for a log record that is associated with a data sharing member. DB2 uses the LRSN for recovery in the data sharing environment.

log truncation. A process by which an explicit starting RBA is established. This RBA is the point at which the next byte of log data is to be written.

LPL. Logical page list.

LRECP. Logical recovery pending.

LRH. Log record header.

LRSN. Log record sequence number.

LU. Logical unit.

LU name. Logical unit name, which is the name by which VTAM refers to a node in a network. Contrast with *location name*.

LUW. Logical unit of work.

LUWID. Logical unit of work identifier.

M

mapping table. A table that the REORG utility uses to map the associations of the RIDs of data records in the original copy and in the shadow copy. This table is created by the user.

mass delete. The deletion of all rows of a table.

master terminal. The IMS logical terminal that has complete control of IMS resources during online operations.

master terminal operator (MTO). See *master terminal.*

materialize. (1) The process of putting rows from a view or nested table expression into a work file for additional processing by a query.

(2) The placement of a LOB value into contiguous storage. Because LOB values can be very large, DB2 avoids materializing LOB data until doing so becomes absolutely necessary.

| **materialized query table.** A table that is used to contain information that is derived and can be summarized from one or more source tables.

MB. Megabyte (1 048 576 bytes).

MBCS. Multibyte character set. UTF-8 is an example of an MBCS. Characters in UTF-8 can range from 1 to 4 bytes in DB2.

member name. The z/OS XCF identifier for a particular DB2 subsystem in a data sharing group.

menu. A displayed list of available functions for selection by the operator. A menu is sometimes called a *menu panel*.

| **metalanguage.** A language that is used to create other specialized languages.

migration. The process of converting a subsystem with a previous release of DB2 to an updated or current release. In this process, you can acquire the functions of the updated or current release without losing the data that you created on the previous release.

mixed data string. A character string that can contain both single-byte and double-byte characters.

MLPA. Modified link pack area.

MODEENT. A VTAM macro instruction that associates a logon mode name with a set of parameters representing session protocols. A set of MODEENT macro instructions defines a logon mode table.

modeling database. A DB2 database that you create on your workstation that you use to model a DB2 UDB for z/OS subsystem, which can then be evaluated by the Index Advisor.

mode name. A VTAM name for the collection of physical and logical characteristics and attributes of a session.

modify locks. An L-lock or P-lock with a MODIFY attribute. A list of these active locks is kept at all times in the coupling facility lock structure. If the requesting DB2 subsystem fails, that DB2 subsystem's modify locks are converted to retained locks.

MPP. Message processing program (in IMS).

MTO. Master terminal operator.

multibyte character set (MBCS). A character set that represents single characters with more than a single byte. Contrast with *single-byte character set* and *double-byte character set*. See also *Unicode*.

multidimensional analysis. The process of assessing and evaluating an enterprise on more than one level.

Multiple Virtual Storage. An element of the z/OS operating system. This element is also called the Base Control Program (BCP).

multisite update. Distributed relational database processing in which data is updated in more than one location within a single unit of work.

multithreading. Multiple TCBs that are executing one copy of DB2 ODBC code concurrently (sharing a processor) or in parallel (on separate central processors).

must-complete. A state during DB2 processing in which the entire operation must be completed to maintain data integrity.

mutex. Pthread mutual exclusion; a lock. A Pthread mutex variable is used as a locking mechanism to allow serialization of critical sections of code by temporarily blocking the execution of all but one thread.

| **MVS.** See *Multiple Virtual Storage*.

N

negotiable lock. A lock whose mode can be downgraded, by agreement among contending users, to be compatible to all. A physical lock is an example of a negotiable lock.

nested table expression. A fullselect in a FROM clause (surrounded by parentheses).

network identifier (NID). The network ID that is assigned by IMS or CICS, or if the connection type is RRSAF, the RRS unit of recovery ID (URID).

NID. Network identifier.

nonleaf page. A page that contains keys and page numbers of other pages in the index (either leaf or nonleaf pages). Nonleaf pages never point to actual data.

| **nonpartitioned index.** An index that is not physically
| partitioned. Both partitioning indexes and secondary
| indexes can be nonpartitioned.

nonscrollable cursor. A cursor that can be moved only in a forward direction. Nonscrollable cursors are sometimes called forward-only cursors or serial cursors.

normalization. A key step in the task of building a logical relational database design. Normalization helps you avoid redundancies and inconsistencies in your data. An entity is normalized if it meets a set of constraints for a particular normal form (first normal form, second normal form, and so on). Contrast with *denormalization*.

nondeterministic function. A user-defined function whose result is not solely dependent on the values of the input arguments. That is, successive invocations with the same argument values can produce a different answer. this type of function is sometimes called a *variant* function. Contrast this with a *deterministic function* (sometimes called a *not-variant function*), which always produces the same result for the same inputs.

not-variant function. See *deterministic function*.

| **NPSI.** See *nonpartitioned secondary index*.

NRE. Network recovery element.

NUL. The null character ('\0'), which is represented by the value X'00'. In C, this character denotes the end of a string.

null. A special value that indicates the absence of information.

NULLIF. A scalar function that evaluates two passed expressions, returning either NULL if the arguments are equal or the value of the first argument if they are not.

null-terminated host variable. A varying-length host variable in which the end of the data is indicated by a null terminator.

null terminator. In C, the value that indicates the end of a string. For EBCDIC, ASCII, and Unicode UTF-8 strings, the null terminator is a single-byte value (X'00'). For Unicode UCS-2 (wide) strings, the null terminator is a double-byte value (X'0000').

O

OASN (origin application schedule number). In IMS, a 4-byte number that is assigned sequentially to each IMS schedule since the last cold start of IMS. The OASN is used as an identifier for a unit of work. In an 8-byte format, the first 4 bytes contain the schedule number and the last 4 bytes contain the number of IMS sync points (*commit points*) during the current schedule. The OASN is part of the NID for an IMS connection.

ODBC. Open Database Connectivity.

ODBC driver. A dynamically-linked library (DLL) that implements ODBC function calls and interacts with a data source.

OBID. Data object identifier.

Open Database Connectivity (ODBC). A Microsoft® database application programming interface (API) for C that allows access to database management systems by using callable SQL. ODBC does not require the use of an SQL preprocessor. In addition, ODBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time. This means that applications no longer need to be directly linked to the modules of all the database management systems that are supported.

ordinary identifier. An uppercase letter followed by zero or more characters, each of which is an uppercase letter, a digit, or the underscore character. An ordinary identifier must not be a reserved word.

ordinary token. A numeric constant, an ordinary identifier, a host identifier, or a keyword.

originating task. In a parallel group, the primary agent that receives data from other execution units (referred to as *parallel tasks*) that are executing portions of the query in parallel.

OS/390. Operating System/390®.

outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves some or all of the unmatched rows of the tables that are being joined. See also *join*.

overloaded function. A function name for which multiple function instances exist.

P

package. An object containing a set of SQL statements that have been statically bound and that is available for processing. A package is sometimes also called an *application package*.

package list. An ordered list of package names that may be used to extend an application plan.

package name. The name of an object that is created by a BIND PACKAGE or REBIND PACKAGE command. The object is a bound version of a database request module (DBRM). The name consists of a location name, a collection ID, a package ID, and a version ID.

page. A unit of storage within a table space (4 KB, 8 KB, 16 KB, or 32 KB) or index space (4 KB). In a table space, a page contains one or more rows of a table. In a LOB table space, a LOB value can span more than one page, but no more than one LOB value is stored on a page.

page set. Another way to refer to a table space or index space. Each page set consists of a collection of VSAM data sets.

page set recovery pending (PSRCP). A restrictive state of an index space. In this case, the entire page set must be recovered. Recovery of a logical part is prohibited.

panel. A predefined display image that defines the locations and characteristics of display fields on a display surface (for example, a *menu panel*).

parallel complex. A cluster of machines that work together to handle multiple transactions and applications.

parallel group. A set of consecutive operations that execute in parallel and that have the same number of parallel tasks.

parallel I/O processing. A form of I/O processing in which DB2 initiates multiple concurrent requests for a single user query and performs I/O processing concurrently (in *parallel*) on multiple data partitions.

parallelism assistant. In Sysplex query parallelism, a DB2 subsystem that helps to process parts of a parallel query that originates on another DB2 subsystem in the data sharing group.

parallelism coordinator. In Sysplex query parallelism, the DB2 subsystem from which the parallel query originates.

Parallel Sysplex. A set of z/OS systems that communicate and cooperate with each other through certain multisystem hardware components and software services to process customer workloads.

parallel task. The execution unit that is dynamically created to process a query in parallel. A parallel task is implemented by a z/OS service request block.

parameter marker. A question mark (?) that appears in a statement string of a dynamic SQL statement. The question mark can appear where a host variable could appear if the statement string were a static SQL statement.

| **parameter-name.** An SQL identifier that designates a
| parameter in an SQL procedure or an SQL function.

parent key. A primary key or unique key in the parent table of a referential constraint. The values of a parent key determine the valid values of the foreign key in the referential constraint.

| **parent lock.** For explicit hierarchical locking, a lock
| that is held on a resource that might have child locks
| that are lower in the hierarchy. A parent lock is usually
| the table space lock or the partition intent lock. See also
| *child lock*.

parent row. A row whose primary key value is the foreign key value of a dependent row.

parent table. A table whose primary key is referenced by the foreign key of a dependent table.

parent table space. A table space that contains a parent table. A table space containing a dependent of that table is a dependent table space.

participant. An entity other than the commit coordinator that takes part in the commit process. The term participant is synonymous with *agent* in SNA.

partition. A portion of a page set. Each partition corresponds to a single, independently extendable data set. Partitions can be extended to a maximum size of 1, 2, or 4 GB, depending on 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 disk storage that is divided into partitions, which are called members. Each partition can contain a program, part of a program, or data. The term partitioned data set is synonymous with program library.

| **partitioned index.** An index that is physically
| partitioned. Both partitioning indexes and secondary
| indexes can be partitioned.

partitioned page set. A partitioned table space or an index space. Header pages, space map pages, data pages, and index pages reference data only within the scope of the partition.

partitioned table space. A table space that is subdivided into parts (based on index key range), each of which can be processed independently by utilities.

| **partitioning index.** An index in which the leftmost columns are the partitioning columns of the table. The index can be partitioned or nonpartitioned.

| **partition pruning.** The removal from consideration of inapplicable partitions through setting up predicates in a query on a partitioned table to access only certain partitions to satisfy the query.

partner logical unit. An access point in the SNA network that is connected to the local DB2 subsystem by way of a VTAM conversation.

path. See *SQL path*.

PCT. Program control table (in CICS).

PDS. Partitioned data set.

piece. A data set of a nonpartitioned page set.

physical claim. A claim on an entire nonpartitioning index.

physical consistency. The state of a page that is not in a partially changed state.

physical drain. A drain on an entire nonpartitioning index.

physical lock (P-lock). A type of lock that DB2 acquires to provide consistency of data that is cached in different DB2 subsystems. Physical locks are used only in data sharing environments. Contrast with *logical lock (L-lock)*.

physical lock contention. Conflicting states of the requesters for a physical lock. See also *negotiable lock*.

physically complete. The state in which the concurrent copy process is completed and the output data set has been created.

plan. See *application plan*.

plan allocation. The process of allocating DB2 resources to a plan in preparation for execution.

plan member. The bound copy of a DBRM that is identified in the member clause.

plan name. The name of an application plan.

plan segmentation. The dividing of each plan into sections. When a section is needed, it is independently brought into the EDM pool.

P-lock. Physical lock.

PLT. Program list table (in CICS).

point of consistency. A time when all recoverable data that an application accesses is consistent with other data. The term point of consistency is synonymous with *sync point* or *commit point*.

policy. See *CFRM policy*.

Portable Operating System Interface (POSIX). The IEEE operating system interface standard, which defines the Pthread standard of threading. See also *Pthread*.

POSIX. Portable Operating System Interface.

postponed abort UR. A unit of recovery that was inflight or in-abort, was interrupted by system failure or cancellation, and did not complete backout during restart.

PPT. (1) Processing program table (in CICS). (2) Program properties table (in z/OS).

precision. In SQL, the total number of digits in a decimal number (called the *size* in the C language). In the C language, the number of digits to the right of the decimal point (called the *scale* in SQL). The DB2 library uses the SQL terms.

precompilation. A processing of application programs containing SQL statements that takes place before compilation. SQL statements are replaced with statements that are recognized by the host language compiler. Output from this precompilation includes source code that can be submitted to the compiler and the database request module (DBRM) that is input to the bind process.

predicate. An element of a search condition that expresses or implies a comparison operation.

prefix. A code at the beginning of a message or record.

preformat. The process of preparing a VSAM ESDS for DB2 use, by writing specific data patterns.

prepare. The first phase of a two-phase commit process in which all participants are requested to prepare for commit.

prepared SQL statement. A named object that is the executable form of an SQL statement that has been processed by the PREPARE statement.

presumed-abort. An optimization of the presumed-nothing two-phase commit protocol that reduces the number of recovery log records, the duration of state maintenance, and the number of messages between coordinator and participant. The optimization also modifies the indoubt resolution responsibility.

presumed-nothing. The standard two-phase commit protocol that defines coordinator and participant responsibilities, relative to logical unit of work states, recovery logging, and indoubt resolution.

primary authorization ID. The authorization ID that is used to identify the application process to DB2.

primary group buffer pool. For a duplexed group buffer pool, the structure that is used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The z/OS equivalent is *old* structure. Compare with *secondary group buffer pool*.

primary index. An index that enforces the uniqueness of a primary key.

primary key. In a relational database, a unique, nonnull key that is part of the definition of a table. A table cannot be defined as a parent unless it has a unique key or primary key.

principal. An entity that can communicate securely with another entity. In Kerberos, principals are represented as entries in the Kerberos registry database and include users, servers, computers, and others.

principal name. The name by which a principal is known to the DCE security services.

private connection. A communications connection that is specific to DB2.

private protocol access. A method of accessing distributed data by which you can direct a query to another DB2 system. Contrast with *DRDA access*.

private protocol connection. A DB2 private connection of the application process. See also *private connection*.

privilege. The capability of performing a specific function, sometimes on a specific object. The types of privileges are:

- **explicit privileges**, which have names and are held as the result of SQL GRANT and REVOKE statements. For example, the SELECT privilege.
- **implicit privileges**, which accompany the ownership of an object, such as the privilege to drop a synonym that one owns, or the holding of an authority, such as the privilege of SYSADM authority to terminate any utility job.

privilege set. For the installation SYSADM ID, the set of all possible privileges. For any other authorization ID, the set of all privileges that are recorded for that ID in the DB2 catalog.

process. In DB2, the unit to which DB2 allocates resources and locks. Sometimes called an *application process*, a process involves the execution of one or more programs. The execution of an SQL statement is always associated with some process. The means of initiating and terminating a process are dependent on the environment.

program. A single, compilable collection of executable statements in a programming language.

program temporary fix (PTF). A solution or bypass of a problem that is diagnosed as a result of a defect in a current unaltered release of a licensed program. An authorized program analysis report (APAR) fix is corrective service for an existing problem. A PTF is preventive service for problems that might be encountered by other users of the product. A PTF is *temporary*, because a permanent fix is usually not incorporated into the product until its next release.

protected conversation. A VTAM conversation that supports two-phase commit flows.

PSRCP. Page set recovery pending.

PTF. Program temporary fix.

Pthread. The POSIX threading standard model for splitting an application into subtasks. The Pthread standard includes functions for creating threads, terminating threads, synchronizing threads through locking, and other thread control facilities.

Q

QMF™. Query Management Facility.

QSAM. Queued sequential access method.

query. A component of certain SQL statements that specifies a result table.

query block. The part of a query that is represented by one of the FROM clauses. Each FROM clause can have multiple query blocks, depending on DB2's internal processing of the query.

query CP parallelism. Parallel execution of a single query, which is accomplished by using multiple tasks. See also *Sysplex query parallelism*.

query I/O parallelism. Parallel access of data, which is accomplished by triggering multiple I/O requests within a single query.

queued sequential access method (QSAM). An extended version of the basic sequential access method (BSAM). When this method is used, a queue of data blocks is formed. Input data blocks await processing, and output data blocks await transfer to auxiliary storage or to an output device.

quiesce point. A point at which data is consistent as a result of running the DB2 QUIESCE utility.

quiesced member state. A state of a member of a data sharing group. An active member becomes quiesced when a STOP DB2 command takes effect without a failure. If the member's task, address space, or z/OS system fails before the command takes effect, the member state is failed.

R

| **RACF.** Resource Access Control Facility, which is a
| component of the z/OS Security Server.

RAMAC®. IBM family of enterprise disk storage system products.

RBA. Relative byte address.

RCT. Resource control table (in CICS attachment facility).

RDB. Relational database.

RDBMS. Relational database management system.

RDBNAM. Relational database name.

RDF. Record definition field.

read stability (RS). An isolation level that is similar to repeatable read but does not completely isolate an application process from all other concurrently executing application processes. Under level RS, an application that issues the same query more than once might read additional rows that were inserted and committed by a concurrently executing application process.

rebind. The creation of a new application plan for an application program that has been bound previously. If, for example, you have added an index for a table that your application accesses, you must rebind the application in order to take advantage of that index.

rebuild. The process of reallocating a coupling facility structure. For the shared communications area (SCA) and lock structure, the structure is repopulated; for the group buffer pool, changed pages are usually cast out to disk, and the new structure is populated only with changed pages that were not successfully cast out.

RECFM. Record format.

record. The storage representation of a row or other data.

record identifier (RID). A unique identifier that DB2 uses internally to identify a row of data in a table. Compare with *row ID*.

| **record identifier (RID) pool.** An area of main storage
| that is used for sorting record identifiers during
| list-prefetch processing.

record length. The sum of the length of all the columns in a table, which is the length of the data as it is physically stored in the database. Records can be fixed length or varying length, depending on how the columns are defined. If all columns are fixed-length columns, the record is a fixed-length record. If one or more columns are varying-length columns, the record is a varying-length column.

Recoverable Resource Manager Services attachment facility (RRSAF). A DB2 subcomponent that uses Resource Recovery Services to coordinate resource commitment between DB2 and all other resource managers that also use RRS in a z/OS system.

recovery. The process of rebuilding databases after a system failure.

recovery log. A collection of records that describes the events that occur during DB2 execution and indicates their sequence. The recorded information is used for recovery in the event of a failure during DB2 execution.

recovery manager. (1) A subcomponent that supplies coordination services that control the interaction of DB2 resource managers during commit, abort, checkpoint, and restart processes. The recovery manager also supports the recovery mechanisms of other subsystems (for example, IMS) by acting as a participant in the other subsystem's process for protecting data that has reached a point of consistency. (2) A coordinator or a participant (or both), in the execution of a two-phase commit, that can access a recovery log that maintains the state of the logical unit of work and names the immediate upstream coordinator and downstream participants.

recovery pending (RECP). A condition that prevents SQL access to a table space that needs to be recovered.

recovery token. An identifier for an element that is used in recovery (for example, NID or URID).

RECP. Recovery pending.

redo. A state of a unit of recovery that indicates that changes are to be reapplied to the disk media to ensure data integrity.

reentrant. Executable code that can reside in storage as one shared copy for all threads. Reentrant code is not self-modifying and provides separate storage areas for each thread. Reentrancy is a compiler and operating system concept, and reentrancy alone is not enough to guarantee logically consistent results when multithreading. See also *threadsafe*.

referential constraint. The requirement that nonnull values of a designated foreign key are valid only if they equal values of the primary key of a designated table.

referential integrity. The state of a database in which all values of all foreign keys are valid. Maintaining referential integrity requires the enforcement of referential constraints on all operations that change the data in a table on which the referential constraints are defined.

referential structure. A set of tables and relationships that includes at least one table and, for every table in the set, all the relationships in which that table participates and all the tables to which it is related.

| **refresh age.** The time duration between the current time and the time during which a materialized query table was last refreshed.

registry. See *registry database*.

registry database. A database of security information about principals, groups, organizations, accounts, and security policies.

relational database (RDB). A database that can be perceived as a set of tables and manipulated in accordance with the relational model of data.

relational database management system (RDBMS). A collection of hardware and software that organizes and provides access to a relational database.

relational database name (RDBNAM). A unique identifier for an RDBMS within a network. In DB2, this must be the value in the LOCATION column of table SYSIBM.LOCATIONS in the CDB. DB2 publications refer to the name of another RDBMS as a LOCATION value or a location name.

relationship. A defined connection between the rows of a table or the rows of two tables. A relationship is the internal representation of a referential constraint.

relative byte address (RBA). The offset of a data record or control interval from the beginning of the storage space that is allocated to the data set or file to which it belongs.

remigration. The process of returning to a current release of DB2 following a fallback to a previous release. This procedure constitutes another migration process.

remote. Any object that is maintained by a remote DB2 subsystem (that is, by a DB2 subsystem other than the local one). A *remote view*, for example, is a view that is maintained by a remote DB2 subsystem. Contrast with *local*.

remote attach request. A request by a remote location to attach to the local DB2 subsystem. Specifically, the request that is sent is an SNA Function Management Header 5.

remote subsystem. Any relational DBMS, except the *local subsystem*, with which the user or application can communicate. The subsystem need not be remote in any physical sense, and might even operate on the same processor under the same z/OS system.

reoptimization. The DB2 process of reconsidering the access path of an SQL statement at run time; during reoptimization, DB2 uses the values of host variables, parameter markers, or special registers.

REORG pending (REORP). A condition that restricts SQL access and most utility access to an object that must be reorganized.

REORP. REORG pending.

repeatable read (RR). The isolation level that provides maximum protection from other executing application programs. When an application program executes with repeatable read protection, rows that the program references cannot be changed by other programs until the program reaches a commit point.

repeating group. A situation in which an entity includes multiple attributes that are inherently the same. The presence of a repeating group violates the requirement of first normal form. In an entity that satisfies the requirement of first normal form, each attribute is independent and unique in its meaning and its name. See also *normalization*.

replay detection mechanism. A method that allows a principal to detect whether a request is a valid request from a source that can be trusted or whether an untrustworthy entity has captured information from a previous exchange and is replaying the information exchange to gain access to the principal.

request commit. The vote that is submitted to the prepare phase if the participant has modified data and is prepared to commit or roll back.

requester. The source of a request to access data at a remote server. In the DB2 environment, the requester function is provided by the distributed data facility.

resource. The object of a lock or claim, which could be a table space, an index space, a data partition, an index partition, or a logical partition.

resource allocation. The part of plan allocation that deals specifically with the database resources.

resource control table (RCT). A construct of the CICS attachment facility, created by site-provided macro parameters, that defines authorization and access attributes for transactions or transaction groups.

resource definition online. A CICS feature that you use to define CICS resources online without assembling tables.

resource limit facility (RLF). A portion of DB2 code that prevents dynamic manipulative SQL statements from exceeding specified time limits. The resource limit facility is sometimes called the governor.

resource limit specification table (RLST). A site-defined table that specifies the limits to be enforced by the resource limit facility.

resource manager. (1) A function that is responsible for managing a particular resource and that guarantees the consistency of all updates made to recoverable resources within a logical unit of work. The resource that is being managed can be physical (for example, disk or main storage) or logical (for example, a particular type of system service). (2) A participant, in the execution of a two-phase commit, that has recoverable resources that could have been modified. The resource manager has access to a recovery log so that it can commit or roll back the effects of the logical unit of work to the recoverable resources.

restart pending (RESTP). A restrictive state of a page set or partition that indicates that restart (backout) work needs to be performed on the object. All access to the page set or partition is denied except for access by the:
- RECOVER POSTPONED command
- Automatic online backout (which DB2 invokes after restart if the system parameter LBACKOUT=AUTO)

RESTP. Restart pending.

result set. The set of rows that a stored procedure returns to a client application.

result set locator. A 4-byte value that DB2 uses to uniquely identify a query result set that a stored procedure returns.

result table. The set of rows that are specified by a SELECT statement.

retained lock. A MODIFY lock that a DB2 subsystem was holding at the time of a subsystem failure. The lock is retained in the coupling facility lock structure across a DB2 failure.

RID. Record identifier.

RID pool. Record identifier pool.

right outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of the second join operand. See also *join*.

RLF. Resource limit facility.

RLST. Resource limit specification table.

RMID. Resource manager identifier.

RO. Read-only access.

rollback. The process of restoring data that was changed by SQL statements to the state at its last commit point. All locks are freed. Contrast with *commit*.

root page. The index page that is at the highest level (or the beginning point) in an index.

routine. A term that refers to either a user-defined function or a stored procedure.

row. The horizontal component of a table. A row consists of a sequence of values, one for each column of the table.

ROWID. Row identifier.

row identifier (ROWID). A value that uniquely identifies a row. This value is stored with the row and never changes.

row lock. A lock on a single row of data.

| **rowset.** A set of rows for which a cursor position is
| established.

| **rowset cursor.** A cursor that is defined so that one or
| more rows can be returned as a rowset for a single
| FETCH statement, and the cursor is positioned on the
| set of rows that is fetched.

| **rowset-positioned access.** The ability to retrieve
| multiple rows from a single FETCH statement.

| **row-positioned access.** The ability to retrieve a single
| row from a single FETCH statement.

row trigger. A trigger that is defined with the trigger granularity FOR EACH ROW.

RRE. Residual recovery entry (in IMS).

RRSAF. Recoverable Resource Manager Services attachment facility.

RS. Read stability.

RTT. Resource translation table.

RURE. Restart URE.

S

savepoint. A named entity that represents the state of data and schemas at a particular point in time within a unit of work. SQL statements exist to set a savepoint, release a savepoint, and restore data and schemas to the state that the savepoint represents. The restoration of data and schemas to a savepoint is usually referred to as *rolling back to a savepoint*.

SBCS. Single-byte character set.

SCA. Shared communications area.

scalar function. An SQL operation that produces a
single value from another value and is expressed as a
function name, followed by a list of arguments that are
enclosed in parentheses. Contrast with *aggregate*
function.

scale. In SQL, the number of digits to the right of the decimal point (called the *precision* in the C language). The DB2 library uses the SQL definition.

schema. (1) The organization or structure of a database. (2) A logical grouping for user-defined functions, distinct types, triggers, and stored procedures. When an object of one of these types is created, it is assigned to one schema, which is determined by the name of the object. For example, the following statement creates a distinct type T in schema C:

```
CREATE DISTINCT TYPE C.T ...
```

scrollability. The ability to use a cursor to fetch in either a forward or backward direction. The FETCH statement supports multiple fetch orientations to indicate the new position of the cursor. See also *fetch orientation*.

scrollable cursor. A cursor that can be moved in both a forward and a backward direction.

SDWA. System diagnostic work area.

search condition. A criterion for selecting rows from a table. A search condition consists of one or more predicates.

secondary authorization ID. An authorization ID that has been associated with a primary authorization ID by an authorization exit routine.

secondary group buffer pool. For a duplexed group buffer pool, the structure that is used to back up changed pages that are written to the primary group buffer pool. No page registration or cross-invalidation occurs using the secondary group buffer pool. The z/OS equivalent is *new* structure.

secondary index. A nonpartitioning index on a partitioned table.

section. The segment of a plan or package that contains the executable structures for a single SQL statement. For most SQL statements, one section in the plan exists for each SQL statement in the source program. However, for cursor-related statements, the DECLARE, OPEN, FETCH, and CLOSE statements reference the same section because they each refer to the SELECT statement that is named in the DECLARE CURSOR statement. SQL statements such as COMMIT, ROLLBACK, and some SET statements do not use a section.

segment. A group of pages that holds rows of a single table. See also *segmented table space*.

segmented table space. A table space that is divided into equal-sized groups of pages called segments. Segments are assigned to tables so that rows of different tables are never stored in the same segment.

self-referencing constraint. A referential constraint that defines a relationship in which a table is a dependent of itself.

self-referencing table. A table with a self-referencing constraint.

sensitive cursor. A cursor that is sensitive to changes that are made to the database after the result table has been materialized.

sequence. A user-defined object that generates a sequence of numeric values according to user specifications.

sequential data set. A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 database utilities require sequential data sets.

sequential prefetch. A mechanism that triggers consecutive asynchronous I/O operations. Pages are fetched before they are required, and several pages are read with a single I/O operation.

serial cursor. A cursor that can be moved only in a forward direction.

serialized profile. A Java object that contains SQL statements and descriptions of host variables. The SQLJ translator produces a serialized profile for each connection context.

server. The target of a request from a remote requester. In the DB2 environment, the server function is provided by the distributed data facility, which is used to access DB2 data from remote applications.

server-side programming. A method for adding DB2 data into dynamic Web pages.

service class. An eight-character identifier that is used by the z/OS Workload Manager to associate user performance goals with a particular DDF thread or stored procedure. A service class is also used to classify work on parallelism assistants.

service request block. A unit of work that is scheduled to execute in another address space.

session. A link between two nodes in a VTAM network.

session protocols. The available set of SNA communication requests and responses.

shared communications area (SCA). A coupling facility list structure that a DB2 data sharing group uses for inter-DB2 communication.

share lock. A lock that prevents concurrently executing application processes from changing data, but not from reading data. Contrast with *exclusive lock*.

shift-in character. A special control character (X'0F') that is used in EBCDIC systems to denote that the subsequent bytes represent SBCS characters. See also *shift-out character*.

shift-out character. A special control character (X'0E') that is used in EBCDIC systems to denote that the subsequent bytes, up to the next shift-in control character, represent DBCS characters. See also *shift-in character*.

sign-on. A request that is made on behalf of an individual CICS or IMS application process by an attachment facility to enable DB2 to verify that it is authorized to use DB2 resources.

simple page set. A nonpartitioned page set. A simple page set initially consists of a single data set (page set piece). If and when that data set is extended to 2 GB, another data set is created, and so on, up to a total of 32 data sets. DB2 considers the data sets to be a single contiguous linear address space containing a maximum of 64 GB. Data is stored in the next available location within this address space without regard to any partitioning scheme.

simple table space. A table space that is neither partitioned nor segmented.

single-byte character set (SBCS). A set of characters in which each character is represented by a single byte. Contrast with *double-byte character set* or *multibyte character set*.

single-precision floating point number. A 32-bit approximate representation of a real number.

size. In the C language, the total number of digits in a decimal number (called the *precision* in SQL). The DB2 library uses the SQL term.

SMF. System Management Facilities.

SMP/E. System Modification Program/Extended.

SMS. Storage Management Subsystem.

SNA. Systems Network Architecture.

SNA network. The part of a network that conforms to the formats and protocols of Systems Network Architecture (SNA).

socket. A callable TCP/IP programming interface that TCP/IP network applications use to communicate with remote TCP/IP partners.

sourced function. A function that is implemented by another built-in or user-defined function that is already known to the database manager. This function can be a scalar function or a column (aggregating) function; it returns a single value from a set of values (for example, MAX or AVG). Contrast with *built-in function, external function,* and *SQL function*.

source program. A set of host language statements and SQL statements that is processed by an SQL precompiler.

| **source table.** A table that can be a base table, a view, a
| table expression, or a user-defined table function.

source type. An existing type that DB2 uses to internally represent a distinct type.

space. A sequence of one or more blank characters.

special register. A storage area that DB2 defines for an application process to use for storing information that can be referenced in SQL statements. Examples of special registers are USER and CURRENT DATE.

specific function name. A particular user-defined function that is known to the database manager by its specific name. Many specific user-defined functions can have the same function name. When a user-defined function is defined to the database, every function is assigned a specific name that is unique within its schema. Either the user can provide this name, or a default name is used.

SPUFI. SQL Processor Using File Input.

SQL. Structured Query Language.

SQL authorization ID (SQL ID). The authorization ID that is used for checking dynamic SQL statements in some situations.

SQLCA. SQL communication area.

SQL communication area (SQLCA). A structure that is used to provide an application program with information about the execution of its SQL statements.

SQL connection. An association between an application process and a local or remote application server or database server.

SQLDA. SQL descriptor area.

SQL descriptor area (SQLDA). A structure that describes input variables, output variables, or the columns of a result table.

SQL escape character. The symbol that is used to enclose an SQL delimited identifier. This symbol is the double quotation mark ("). See also *escape character*.

SQL function. A user-defined function in which the CREATE FUNCTION statement contains the source code. The source code is a single SQL expression that evaluates to a single value. The SQL user-defined function can return only one parameter.

SQL ID. SQL authorization ID.

SQLJ. Structured Query Language (SQL) that is embedded in the Java programming language.

SQL path. An ordered list of schema names that are used in the resolution of unqualified references to user-defined functions, distinct types, and stored procedures. In dynamic SQL, the current path is found in the CURRENT PATH special register. In static SQL, it is defined in the PATH bind option.

SQL procedure. A user-written program that can be invoked with the SQL CALL statement. Contrast with *external procedure*.

SQL processing conversation. Any conversation that requires access of DB2 data, either through an application or by dynamic query requests.

SQL Processor Using File Input (SPUFI). A facility of the TSO attachment subcomponent that enables the DB2I user to execute SQL statements without embedding them in an application program.

SQL return code. Either SQLCODE or SQLSTATE.

SQL routine. A user-defined function or stored procedure that is based on code that is written in SQL.

SQL statement coprocessor. An alternative to the DB2 precompiler that lets the user process SQL statements at compile time. The user invokes an SQL statement coprocessor by specifying a compiler option.

SQL string delimiter. A symbol that is used to enclose an SQL string constant. The SQL string delimiter is the apostrophe ('), except in COBOL applications, where the user assigns the symbol, which is either an apostrophe or a double quotation mark (").

SRB. Service request block.

SSI. Subsystem interface (in z/OS).

SSM. Subsystem member (in IMS).

stand-alone. An attribute of a program that means that it is capable of executing separately from DB2, without using DB2 services.

star join. A method of joining a dimension column of a fact table to the key column of the corresponding dimension table. See also *join*, *dimension*, and *star schema*.

star schema. The combination of a fact table (which contains most of the data) and a number of dimension tables. See also *star join*, *dimension*, and *dimension table*.

statement handle. In DB2 ODBC, the data object that contains information about an SQL statement that is managed by DB2 ODBC. This includes information such as dynamic arguments, bindings for dynamic arguments and columns, cursor information, result values, and status information. Each statement handle is associated with the connection handle.

statement string. For a dynamic SQL statement, the character string form of the statement.

statement trigger. A trigger that is defined with the trigger granularity FOR EACH STATEMENT.

| **static cursor.** A named control structure that does not
| change the size of the result table or the order of its
| rows after an application opens the cursor. Contrast
| with *dynamic cursor*.

static SQL. SQL statements, embedded within a program, that are prepared during the program preparation process (before the program is executed). After being prepared, the SQL statement does not change (although values of host variables that are specified by the statement might change).

storage group. A named set of disks on which DB2 data can be stored.

stored procedure. A user-written application program that can be invoked through the use of the SQL CALL statement.

string. See *character string* or *graphic string*.

strong typing. A process that guarantees that only user-defined functions and operations that are defined on a distinct type can be applied to that type. For example, you cannot directly compare two currency types, such as Canadian dollars and U.S. dollars. But you can provide a user-defined function to convert one currency to the other and then do the comparison.

structure. (1) A name that refers collectively to different types of DB2 objects, such as tables, databases, views, indexes, and table spaces. (2) A construct that uses z/OS to map and manage storage on a coupling facility. See also *cache structure*, *list structure*, or *lock structure*.

Structured Query Language (SQL). A standardized language for defining and manipulating data in a relational database.

structure owner. In relation to group buffer pools, the DB2 member that is responsible for the following activities:

- Coordinating rebuild, checkpoint, and damage assessment processing
- Monitoring the group buffer pool threshold and notifying castout owners when the threshold has been reached

subcomponent. A group of closely related DB2 modules that work together to provide a general function.

subject table. The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

subpage. The unit into which a physical index page can be divided.

subquery. A SELECT statement within the WHERE or HAVING clause of another SQL statement; a nested SQL statement.

subselect. That form of a query that does not include an ORDER BY clause, an UPDATE clause, or UNION operators.

substitution character. A unique character that is substituted during character conversion for any characters in the source program that do not have a match in the target coding representation.

subsystem. A distinct instance of a relational database management system (RDBMS).

surrogate pair. A coded representation for a single character that consists of a sequence of two 16-bit code units, in which the first value of the pair is a high-surrogate code unit in the range U+D800 through U+DBFF, and the second value is a low-surrogate code unit in the range U+DC00 through U+DFFF. Surrogate pairs provide an extension mechanism for encoding 917 476 characters without requiring the use of 32-bit characters.

SVC dump. A dump that is issued when a z/OS or a DB2 functional recovery routine detects an error.

sync point. See *commit point*.

syncpoint tree. The tree of recovery managers and resource managers that are involved in a logical unit of work, starting with the recovery manager, that make the final commit decision.

synonym. In SQL, an alternative name for a table or view. Synonyms can be used to refer only to objects at the subsystem in which the synonym is defined.

syntactic character set. A set of 81 graphic characters that are registered in the IBM registry as character set 00640. This set was originally recommended to the programming language community to be used for syntactic purposes toward maximizing portability and interchangeability across systems and country boundaries. It is contained in most of the primary registered character sets, with a few exceptions. See also *invariant character set*.

Sysplex. See *Parallel Sysplex*.

Sysplex query parallelism. Parallel execution of a single query that is accomplished by using multiple tasks on more than one DB2 subsystem. See also *query CP parallelism*.

system administrator. The person at a computer installation who designs, controls, and manages the use of the computer system.

system agent. A work request that DB2 creates internally such as prefetch processing, deferred writes, and service tasks.

system conversation. The conversation that two DB2 subsystems must establish to process system messages before any distributed processing can begin.

system diagnostic work area (SDWA). The data that is recorded in a SYS1.LOGREC entry that describes a program or hardware error.

system-directed connection. A connection that a relational DBMS manages by processing SQL statements with three-part names.

System Modification Program/Extended (SMP/E). A z/OS tool for making software changes in programming systems (such as DB2) and for controlling those changes.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information through and controlling the configuration and operation of networks.

SYS1.DUMPxx data set. A data set that contains a system dump (in z/OS).

SYS1.LOGREC. A service aid that contains important information about program and hardware errors (in z/OS).

T

table. A named data object consisting of a specific number of columns and some number of unordered rows. See also *base table* or *temporary table*.

| **table-controlled partitioning.** A type of partitioning in
| which partition boundaries for a partitioned table are
| controlled by values that are defined in the CREATE
| TABLE statement. Partition limits are saved in the
| LIMITKEY_INTERNAL column of the
| SYSIBM.SYSTABLEPART catalog table.

table function. A function that receives a set of arguments and returns a table to the SQL statement that references the function. A table function can be referenced only in the FROM clause of a subselect.

table locator. A mechanism that allows access to trigger transition tables in the FROM clause of SELECT statements, in the subselect of INSERT statements, or from within user-defined functions. A table locator is a fullword integer value that represents a transition table.

table space. A page set that is used to store the records in one or more tables.

table space set. A set of table spaces and partitions that should be recovered together for one of these reasons:
- Each of them contains a table that is a parent or descendent of a table in one of the others.
- The set contains a base table and associated auxiliary tables.

A table space set can contain both types of relationships.

task control block (TCB). A z/OS control block that is used to communicate information about tasks within an address space that are connected to DB2. See also *address space connection*.

TB. Terabyte (1 099 511 627 776 bytes).

TCB. Task control block (in z/OS).

TCP/IP. A network communication protocol that computer systems use to exchange information across telecommunication links.

TCP/IP port. A 2-byte value that identifies an end user or a TCP/IP network application within a TCP/IP host.

template. A DB2 utilities output data set descriptor that is used for dynamic allocation. A template is defined by the TEMPLATE utility control statement.

temporary table. A table that holds temporary data. Temporary tables are useful for holding or sorting intermediate results from queries that contain a large number of rows. The two types of temporary table, which are created by different SQL statements, are the created temporary table and the declared temporary table. Contrast with *result table*. See also *created temporary table* and *declared temporary table*.

Terminal Monitor Program (TMP). A program that provides an interface between terminal users and command processors and has access to many system services (in z/OS).

thread. The DB2 structure that describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services. Most DB2 functions execute under a thread structure. See also *allied thread* and *database access thread*.

threadsafe. A characteristic of code that allows multithreading both by providing private storage areas for each thread, and by properly serializing shared (global) storage areas.

three-part name. The full name of a table, view, or alias. It consists of a location name, authorization ID, and an object name, separated by a period.

time. A three-part value that designates a time of day in hours, minutes, and seconds.

time duration. A decimal integer that represents a number of hours, minutes, and seconds.

timeout. Abnormal termination of either the DB2 subsystem or of an application because of the unavailability of resources. Installation specifications are set to determine both the amount of time DB2 is to wait for IRLM services after starting, and the amount of time IRLM is to wait if a resource that an application requests is unavailable. If either of these time specifications is exceeded, a timeout is declared.

Time-Sharing Option (TSO). An option in MVS that provides interactive time sharing from remote terminals.

timestamp. A seven-part value that consists of a date and time. The timestamp is expressed in years, months, days, hours, minutes, seconds, and microseconds.

TMP. Terminal Monitor Program.

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 disk media or backed out, as determined by the commit coordinator.

trace. A DB2 facility that provides the ability to monitor and collect DB2 monitoring, auditing, performance, accounting, statistics, and serviceability (global) data.

transaction lock. A lock that is used to control concurrent execution of SQL statements.

transaction program name. In SNA LU 6.2 conversations, the name of the program at the remote logical unit that is to be the other half of the conversation.

| **transient XML data type.** A data type for XML values
| that exists only during query processing.

transition table. A temporary table that contains all the affected rows of the subject table in their state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the table of changed rows in the old state or the new state.

transition variable. A variable that contains a column value of the affected row of the subject table in its state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the set of old values or the set of new values.

| **tree structure.** A data structure that represents entities
| in nodes, with a most one parent node for each node,
| and with only one root node.

trigger. A set of SQL statements that are stored in a DB2 database and executed when a certain event occurs in a DB2 table.

trigger activation. The process that occurs when the trigger event that is defined in a trigger definition is executed. Trigger activation consists of the evaluation of the triggered action condition and conditional execution of the triggered SQL statements.

trigger activation time. An indication in the trigger definition of whether the trigger should be activated before or after the triggered event.

trigger body. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. A trigger body is also called *triggered SQL statements*.

trigger cascading. The process that occurs when the triggered action of a trigger causes the activation of another trigger.

triggered action. The SQL logic that is performed when a trigger is activated. The triggered action consists of an optional triggered action condition and a set of triggered SQL statements that are executed only if the condition evaluates to true.

triggered action condition. An optional part of the triggered action. This Boolean condition appears as a WHEN clause and specifies a condition that DB2 evaluates to determine if the triggered SQL statements should be executed.

triggered SQL statements. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. Triggered SQL statements are also called the *trigger body*.

trigger granularity. A characteristic of a trigger, which determines whether the trigger is activated:
- Only once for the triggering SQL statement
- Once for each row that the SQL statement modifies

triggering event. The specified operation in a trigger definition that causes the activation of that trigger. The triggering event is comprised of a triggering operation (INSERT, UPDATE, or DELETE) and a subject table on which the operation is performed.

triggering SQL operation. The SQL operation that causes a trigger to be activated when performed on the subject table.

trigger package. A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

TSO. Time-Sharing Option.

TSO attachment facility. A DB2 facility consisting of the DSN command processor and DB2I. Applications that are not written for the CICS or IMS environments can run under the TSO attachment facility.

typed parameter marker. A parameter marker that is specified along with its target data type. It has the general form:

```
CAST(? AS data-type)
```

type 1 indexes. Indexes that were created by a release of DB2 before DB2 Version 4 or that are specified as type 1 indexes in Version 4. Contrast with *type 2 indexes*. As of Version 8, type 1 indexes are no longer supported.

type 2 indexes. Indexes that are created on a release of DB2 after Version 7 or that are specified as type 2 indexes in Version 4 or later.

U

UCS-2. Universal Character Set, coded in 2 octets, which means that characters are represented in 16-bits per character.

UDF. User-defined function.

UDT. User-defined data type. In DB2 UDB for z/OS, the term *distinct type* is used instead of user-defined data type. See *distinct type*.

uncommitted read (UR). The isolation level that allows an application to read uncommitted data.

underlying view. The view on which another view is directly or indirectly defined.

undo. A state of a unit of recovery that indicates that the changes that the unit of recovery made to recoverable DB2 resources must be backed out.

Unicode. A standard that parallels the ISO-10646 standard. Several implementations of the Unicode standard exist, all of which have the ability to represent a large percentage of the characters that are contained in the many scripts that are used throughout the world.

uniform resource locator (URL). A Web address, which offers a way of naming and locating specific items on the Web.

union. An SQL operation that combines the results of two SELECT statements. Unions are often used to merge lists of values that are obtained from several tables.

unique constraint. An SQL rule that no two values in a primary key, or in the key of a unique index, can be the same.

unique index. An index that ensures that no identical key values are stored in a column or a set of columns in a table.

unit of recovery. A recoverable sequence of operations within a single resource manager, such as an instance of DB2. Contrast with *unit of work*.

unit of recovery identifier (URID). The LOGRBA of the first log record for a unit of recovery. The URID also appears in all subsequent log records for that unit of recovery.

unit of work. A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations. In a *multisite update* operation, a single unit of work can include several *units of recovery*. Contrast with *unit of recovery*.

Universal Unique Identifier (UUID). An identifier that is immutable and unique across time and space (in z/OS).

unlock. The act of releasing an object or system resource that was previously locked and returning it to general availability within DB2.

untyped parameter marker. A parameter marker that is specified without its target data type. It has the form of a single question mark (?).

updatability. The ability of a cursor to perform positioned updates and deletes. The updatability of a cursor can be influenced by the SELECT statement and the cursor sensitivity option that is specified on the DECLARE CURSOR statement.

update hole. The location on which a cursor is positioned when a row in a result table is fetched again and the new values no longer satisfy the search condition. DB2 marks a row in the result table as an update hole when an update to the corresponding row in the database causes that row to no longer qualify for the result table.

update trigger. A trigger that is defined with the triggering SQL operation UPDATE.

upstream. The node in the syncpoint tree that is responsible, in addition to other recovery or resource managers, for coordinating the execution of a two-phase commit.

UR. Uncommitted read.

URE. Unit of recovery element.

URID . Unit of recovery identifier.

URL. Uniform resource locator.

user-defined data type (UDT). See *distinct type*.

user-defined function (UDF). A function that is defined to DB2 by using the CREATE FUNCTION statement and that can be referenced thereafter in SQL statements. A user-defined function can be an *external function*, a *sourced function*, or an *SQL function*. Contrast with *built-in function*.

user view. In logical data modeling, a model or representation of critical information that the business requires.

UTF-8. Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems. The CCSID value for data in UTF-8 format is 1208. DB2 UDB for z/OS supports UTF-8 in mixed data fields.

UTF-16. Unicode Transformation Format, 16-bit encoding form, which is designed to provide code values for over a million characters and a superset of UCS-2. The CCSID value for data in UTF-16 format is 1200. DB2 UDB for z/OS supports UTF-16 in graphic data fields.

UUID. Universal Unique Identifier.

V

value. The smallest unit of data that is manipulated in SQL.

variable. A data element that specifies a value that can be changed. A COBOL elementary data item is an example of a variable. Contrast with *constant*.

variant function. See *nondeterministic function*.

varying-length string. A character or graphic string whose length varies within set limits. Contrast with *fixed-length string*.

version. A member of a set of similar programs, DBRMs, packages, or LOBs.

A version of a program is the source code that is produced by precompiling the program. The program version is identified by the program name and a timestamp (consistency token).

A version of a DBRM is the DBRM that is produced by precompiling a program. The DBRM version is identified by the same program name and timestamp as a corresponding program version.

A version of a package is the result of binding a DBRM within a particular database system. The package version is identified by the same program name and consistency token as the DBRM.

A version of a LOB is a copy of a LOB value at a point in time. The version number for a LOB is stored in the auxiliary index entry for the LOB.

view. An alternative representation of data from one or more tables. A view can include all or some of the columns that are contained in tables on which it is defined.

view check option. An option that specifies whether every row that is inserted or updated through a view must conform to the definition of that view. A view check option can be specified with the WITH CASCADED CHECK OPTION, WITH CHECK OPTION, or WITH LOCAL CHECK OPTION clauses of the CREATE VIEW statement.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed- and varying-length records on disk devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by relative-record number (in z/OS).

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network (in z/OS).

| **volatile table.** A table for which SQL operations
| choose index access whenever possible.

VSAM. Virtual Storage Access Method.

VTAM. Virtual Telecommunication Access Method (in z/OS).

W

warm start. The normal DB2 restart process, which involves reading and processing log records so that data that is under the control of DB2 is consistent. Contrast with *cold start*.

WLM application environment. A z/OS Workload Manager attribute that is associated with one or more stored procedures. The WLM application environment determines the address space in which a given DB2 stored procedure runs.

write to operator (WTO). An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that might need to be corrected (in z/OS).

WTO. Write to operator.

WTOR. Write to operator (WTO) with reply.

X

XCF. See *cross-system coupling facility*.

XES. See *cross-system extended services*.

| **XML.** See *Extensible Markup Language*.

| **XML attribute.** A name-value pair within a tagged
| XML element that modifies certain features of the
| element.

XML element. A logical structure in an XML
document that is delimited by a start and an end tag.
Anything between the start tag and the end tag is the
content of the element.

| **XML node.** The smallest unit of valid, complete
| structure in a document. For example, a node can
| represent an element, an attribute, or a text string.

| **XML publishing functions.** Functions that return
| XML values from SQL values.

X/Open. An independent, worldwide open systems organization that is supported by most of the world's largest information systems suppliers, user organizations, and software companies. X/Open's goal is to increase the portability of applications by combining existing and emerging standards.

XRF. Extended recovery facility.

Z

| **z/OS.** An operating system for the eServer™ product
| line that supports 64-bit real and virtual storage.

z/OS Distributed Computing Environment (z/OS DCE). A set of technologies that are provided by the Open Software Foundation to implement distributed computing.

Bibliography

DB2 Universal Database for z/OS Version 8 product information:

- *DB2 Administration Guide*, SC18-7413
- *DB2 Application Programming and SQL Guide*, SC18-7415
- *DB2 Application Programming Guide and Reference for Java*, SC18-7414
- *DB2 Codes*, GC18-9603
- *DB2 Command Reference*, SC18-7416
- *DB2 Common Criteria Guide*, SC18-9672
- *DB2 Data Sharing: Planning and Administration*, SC18-7417
- *DB2 Diagnosis Guide and Reference*, LY37-3201
- *DB2 Diagnostic Quick Reference Card*, LY37-3202
- *DB2 Image, Audio, and Video Extenders Administration and Programming*, SC18-7429
- *DB2 Installation Guide*, GC18-7418
- *DB2 Licensed Program Specifications*, GC18-7420
- *DB2 Management Clients Package Program Directory*, GI10-8567
- *DB2 Messages*, GC18-9602
- *DB2 ODBC Guide and Reference*, SC18-7423
- *The Official Introduction to DB2 UDB for z/OS*
- *DB2 Program Directory*, GI10-8566
- *DB2 RACF Access Control Module Guide*, SC18-7433
- *DB2 Reference for Remote DRDA Requesters and Servers*, SC18-7424
- *DB2 Reference Summary*, SX26-3853
- *DB2 Release Planning Guide*, SC18-7425
- *DB2 SQL Reference*, SC18-7426
- *DB2 Text Extender Administration and Programming*, SC18-7430
- *DB2 Utility Guide and Reference*, SC18-7427
- *DB2 What's New?*, GC18-7428
- *DB2 XML Extender for z/OS Administration and Programming*, SC18-7431

Books and resources about related products:

APL2®

- *APL2 Programming Guide*, SH21-1072
- *APL2 Programming: Language Reference*, SH21-1061
- *APL2 Programming: Using Structured Query Language (SQL)*, SH21-1057

BookManager READ/MVS

- *BookManager READ/MVS V1R3: Installation Planning & Customization*, SC38-2035

C language: IBM C/C++ for z/OS

- *z/OS C/C++ Programming Guide*, SC09-4765
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821

Character Data Representation Architecture

- *Character Data Representation Architecture Overview*, GC09-2207
- *Character Data Representation Architecture Reference and Registry*, SC09-2190

CICS Transaction Server for z/OS

The publication order numbers below are for Version 2 Release 2 and Version 2 Release 3 (with the release 2 number listed first).

- *CICS Transaction Server for z/OS Information Center*, SK3T-6903 or SK3T-6957.
- *CICS Transaction Server for z/OS Application Programming Guide*, SC34-5993 or SC34-6231
- *CICS Transaction Server for z/OS Application Programming Reference*, SC34-5994 or SC34-6232
- *CICS Transaction Server for z/OS CICS-RACF Security Guide*, SC34-6011 or SC34-6249
- *CICS Transaction Server for z/OS CICS Supplied Transactions*, SC34-5992 or SC34-6230
- *CICS Transaction Server for z/OS Customization Guide*, SC34-5989 or SC34-6227
- *CICS Transaction Server for z/OS Data Areas*, LY33-6100 or LY33-6103
- *CICS Transaction Server for z/OS DB2 Guide*, SC34-6014 or SC34-6252
- *CICS Transaction Server for z/OS External Interfaces Guide*, SC34-6006 or SC34-6244
- *CICS Transaction Server for z/OS Installation Guide*, GC34-5985 or GC34-6224
- *CICS Transaction Server for z/OS Intercommunication Guide*, SC34-6005 or SC34-6243
- *CICS Transaction Server for z/OS Messages and Codes*, GC34-6003 or GC34-6241
- *CICS Transaction Server for z/OS Operations and Utilities Guide*, SC34-5991 or SC34-6229

- *CICS Transaction Server for z/OS Performance Guide*, SC34-6009 or SC34-6247
- *CICS Transaction Server for z/OS Problem Determination Guide*, SC34-6002 or SC34-6239
- *CICS Transaction Server for z/OS Release Guide*, GC34-5983 or GC34-6218
- *CICS Transaction Server for z/OS Resource Definition Guide*, SC34-5990 or SC34-6228
- *CICS Transaction Server for z/OS System Definition Guide*, SC34-5988 or SC34–6226
- *CICS Transaction Server for z/OS System Programming Reference*, SC34-5595 or SC34–6233

CICS Transaction Server for OS/390
- *CICS Transaction Server for OS/390 Application Programming Guide*, SC33-1687
- *CICS Transaction Server for OS/390 DB2 Guide*, SC33-1939
- *CICS Transaction Server for OS/390 External Interfaces Guide*, SC33-1944
- *CICS Transaction Server for OS/390 Resource Definition Guide*, SC33-1684

COBOL:
- *IBM COBOL Language Reference*, SC27-1408
- *Enterprise COBOL for z/OS Programming Guide*, SC27-1412

Database Design
- *DB2 for z/OS and OS/390 Development for Performance Volume I* by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-605-2
- *DB2 for z/OS and OS/390 Development for Performance Volume II* by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-606-0
- *Handbook of Relational Database Design* by C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8

DB2 Administration Tool
- *DB2 Administration Tool for z/OS User's Guide and Reference*, available on the Web at www.ibm.com/software/data/db2imstools/library.html

DB2 Buffer Pool Analyzer for z/OS
- *DB2 Buffer Pool Tool for z/OS User's Guide and Reference,* available on the Web at www.ibm.com/software/data/db2imstools/library.html

DB2 Connect™
- *IBM DB2 Connect Quick Beginnings for DB2 Connect Enterprise Edition*, GC09-4833
- *IBM DB2 Connect Quick Beginnings for DB2 Connect Personal Edition*, GC09-4834
- *IBM DB2 Connect User's Guide*, SC09-4835

DB2 DataPropagator
- *DB2 Universal Database Replication Guide and Reference*, SC27-1121

DB2 Performance Expert for z/OS, Version 1

The following books are part of the DB2 Performance Expert library. Some of these books include information about the following tools: IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS; and DB2 Buffer Pool Analyzer for z/OS.
- *OMEGAMON Buffer Pool Analyzer User's Guide*, SC18-7972
- *OMEGAMON Configuration and Customization*, SC18-7973
- *OMEGAMON Messages*, SC18-7974
- *OMEGAMON Monitoring Performance from ISPF*, SC18-7975
- *OMEGAMON Monitoring Performance from Performance Expert Client*, SC18-7976
- *OMEGAMON Program Directory*, GI10-8549
- *OMEGAMON Report Command Reference*, SC18-7977
- *OMEGAMON Report Reference*, SC18-7978
- *Using IBM Tivoli OMEGAMON XE on z/OS*, SC18-7979

DB2 Query Management Facility (QMF) Version 8.1
- *DB2 Query Management Facility: DB2 QMF High Performance Option User's Guide for TSO/CICS*, SC18-7450
- *DB2 Query Management Facility: DB2 QMF Messages and Codes*, GC18-7447
- *DB2 Query Management Facility: DB2 QMF Reference*, SC18-7446
- *DB2 Query Management Facility: Developing DB2 QMF Applications*, SC18-7651
- *DB2 Query Management Facility: Getting Started with DB2 QMF for Windows and DB2 QMF for WebSphere*, SC18-7449
- *DB2 Query Management Facility: Getting Started with DB2 QMF Query Miner*, GC18-7451
- *DB2 Query Management Facility: Installing and Managing DB2 QMF for TSO/CICS*, GC18-7444
- *DB2 Query Management Facility: Installing and Managing DB2 QMF for Windows and DB2 QMF for WebSphere*, GC18-7448

- *DB2 Query Management Facility: Introducing DB2 QMF*, GC18-7443
- *DB2 Query Management Facility: Using DB2 QMF*, SC18-7445
- *DB2 Query Management Facility: DB2 QMF Visionary Developer's Guide*, SC18-9093
- *DB2 Query Management Facility: DB2 QMF Visionary Getting Started Guide*, GC18-9092

DB2 Redbooks™

For access to all IBM Redbooks about DB2, see the IBM Redbooks Web page at www.ibm.com/redbooks

DB2 Server for VSE & VM
- *DB2 Server for VM: DBS Utility*, SC09-2983

DB2 Universal Database Cross-Platform information
- *IBM DB2 Universal Database SQL Reference for Cross-Platform Development*, available at www.ibm.com/software/data/developer/cpsqlref/

DB2 Universal Database for iSeries

The following books are available at www.ibm.com/iseries/infocenter
- *DB2 Universal Database for iSeries Performance and Query Optimization*
- *DB2 Universal Database for iSeries Database Programming*
- *DB2 Universal Database for iSeries SQL Programming Concepts*
- *DB2 Universal Database for iSeries SQL Programming with Host Languages*
- *DB2 Universal Database for iSeries SQL Reference*
- *DB2 Universal Database for iSeries Distributed Data Management*
- *DB2 Universal Database for iSeries Distributed Database Programming*

DB2 Universal Database for Linux, UNIX, and Windows:
- *DB2 Universal Database Administration Guide: Planning*, SC09-4822
- *DB2 Universal Database Administration Guide: Implementation*, SC09-4820
- *DB2 Universal Database Administration Guide: Performance*, SC09-4821
- *DB2 Universal Database Administrative API Reference*, SC09-4824
- *DB2 Universal Database Application Development Guide: Building and Running Applications*, SC09-4825
- *DB2 Universal Database Call Level Interface Guide and Reference, Volumes 1 and 2*, SC09-4849 and SC09-4850
- *DB2 Universal Database Command Reference*, SC09-4828
- *DB2 Universal Database SQL Reference Volume 1*, SC09-4844
- *DB2 Universal Database SQL Reference Volume 2*, SC09-4845

Device Support Facilities
- *Device Support Facilities User's Guide and Reference*, GC35-0033

DFSMS

These books provide information about a variety of components of DFSMS, including z/OS DFSMS, z/OS DFSMSdfp™, z/OS DFSMSdss, z/OS DFSMShsm, and z/OS DFP.
- *z/OS DFSMS Access Method Services for Catalogs*, SC26-7394
- *z/OS DFSMSdss Storage Administration Guide*, SC35-0423
- *z/OS DFSMSdss Storage Administration Reference*, SC35-0424
- *z/OS DFSMShsm Managing Your Own Data*, SC35-0420
- *z/OS DFSMSdfp: Using DFSMSdfp in the z/OS Environment*, SC26-7473
- *z/OS DFSMSdfp Diagnosis Reference*, GY27-7618
- *z/OS DFSMS: Implementing System-Managed Storage*, SC27-7407
- *z/OS DFSMS: Macro Instructions for Data Sets*, SC26-7408
- *z/OS DFSMS: Managing Catalogs*, SC26-7409
- *z/OS MVS: Program Management User's Guide and Reference*, SA22-7643
- *z/OS MVS Program Management: Advanced Facilities*, SA22-7644
- *z/OS DFSMSdfp Storage Administration Reference*, SC26-7402
- *z/OS DFSMS: Using Data Sets*, SC26-7410
- *DFSMS/MVS: Using Advanced Services* , SC26-7400
- *DFSMS/MVS: Utilities*, SC26-7414

DFSORT™
- *DFSORT Application Programming: Guide*, SC33-4035
- *DFSORT Installation and Customization*, SC33-4034

Distributed Relational Database Architecture

- *Open Group Technical Standard*; the Open Group presently makes the following DRDA books available through its Web site at www.opengroup.org
  - *Open Group Technical Standard, DRDA Version 3 Vol. 1: Distributed Relational Database Architecture*
  - *Open Group Technical Standard, DRDA Version 3 Vol. 2: Formatted Data Object Content Architecture*
  - *Open Group Technical Standard, DRDA Version 3 Vol. 3: Distributed Data Management Architecture*

Domain Name System
- *DNS and BIND, Third Edition, Paul Albitz and Cricket Liu, O'Reilly*, ISBN 0-59600-158-4

Education
- Information about IBM educational offerings is available on the Web at http://www.ibm.com/software/sw-training/
- A collection of glossaries of IBM terms is available on the IBM Terminology Web site at www.ibm.com/ibm/terminology/index.html

eServer zSeries®
- *IBM eServer zSeries Processor Resource/System Manager Planning Guide*, SB10-7033

Fortran: VS Fortran
- *VS Fortran Version 2: Language and Library Reference*, SC26-4221
- *VS Fortran Version 2: Programming Guide for CMS and MVS*, SC26-4222

High Level Assembler
- *High Level Assembler for MVS and VM and VSE Language Reference*, SC26-4940
- *High Level Assembler for MVS and VM and VSE Programmer's Guide*, SC26-4941

ICSF
- *z/OS ICSF Overview*, SA22-7519
- *Integrated Cryptographic Service Facility Administrator's Guide*, SA22-7521

IMS Version 8

IMS product information is available on the IMS Library Web page, which you can find at www.ibm.com/ims
- *IMS Administration Guide: System*, SC27-1284
- *IMS Administration Guide: Transaction Manager*, SC27-1285
- *IMS Application Programming: Database Manager*, SC27-1286
- *IMS Application Programming: Design Guide*, SC27-1287
- *IMS Application Programming: Transaction Manager*, SC27-1289
- *IMS Command Reference*, SC27-1291
- *IMS Customization Guide*, SC27-1294
- *IMS Install Volume 1: Installation Verification*, GC27-1297
- *IMS Install Volume 2: System Definition and Tailoring*, GC27-1298
- *IMS Messages and Codes Volumes 1 and 2*, GC27-1301 and GC27-1302
- *IMS Open Transaction Manager Access Guide and Reference*, SC18-7829
- *IMS Utilities Reference: System*, SC27-1309

General information about IMS Batch Terminal Simulator for z/OS is available on the Web at www.ibm.com/software/data/db2imstools/library.html

IMS DataPropagator
- *IMS DataPropagator for z/OS Administrator's Guide for Log*, SC27-1216
- *IMS DataPropagator: An Introduction*, GC27-1211
- *IMS DataPropagator for z/OS Reference*, SC27-1210

ISPF
- *z/OS ISPF Dialog Developer's Guide*, SC23-4821
- *z/OS ISPF Messages and Codes*, SC34-4815
- *z/OS ISPF Planning and Customizing*, GC34-4814
- *z/OS ISPF User's Guide Volumes 1 and 2*, SC34-4822 and SC34-4823

Language Environment
- *Debug Tool User's Guide and Reference*, SC18-7171
- *Debug Tool for z/OS and OS/390 Reference and Messages*, SC18-7172
- *z/OS Language Environment Concepts Guide*, SA22-7567
- *z/OS Language Environment Customization*, SA22-7564
- *z/OS Language Environment Debugging Guide*, GA22-7560
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Programming Reference*, SA22-7562

MQSeries®
- *MQSeries Application Messaging Interface*, SC34-5604

- *MQSeries for OS/390 Concepts and Planning Guide*, GC34-5650
- *MQSeries for OS/390 System Setup Guide*, SC34-5651

National Language Support
- *National Language Design Guide Volume 1*, SE09-8001
- *IBM National Language Support Reference Manual Volume 2*, SE09-8002

NetView
- *Tivoli NetView for z/OS Installation: Getting Started*, SC31-8872
- *Tivoli NetView for z/OS User's Guide*, GC31-8849

Microsoft ODBC

Information about Microsoft ODBC is available at http://msdn.microsoft.com/library/

Parallel Sysplex Library
- *System/390 9672 Parallel Transaction Server, 9672 Parallel Enterprise Server, 9674 Coupling Facility System Overview For R1/R2/R3 Based Models*, SB10-7033
- *z/OS Parallel Sysplex Application Migration*, SA22-7662
- *z/OS Parallel Sysplex Overview: An Introduction to Data Sharing and Parallelism*, SA22-7661
- *z/OS Parallel Sysplex Test Report*, SA22-7663

The *Parallel Sysplex Configuration Assistant* is available at www.ibm.com/s390/pso/psotool

PL/I: Enterprise PL/I for z/OS
- *IBM Enterprise PL/I for z/OS Language Reference*, SC27-1460
- *IBM Enterprise PL/I for z/OS Programming Guide*, SC27-1457

PL/I: PL/I for MVS & VM
- *PL/I for MVS & VM Programming Guide*, SC26-3113

SMP/E
- *SMP/E for z/OS and OS/390 Reference*, SA22-7772
- *SMP/E for z/OS and OS/390 User's Guide*, SA22-7773

Storage Management
- *z/OS DFSMS: Implementing System-Managed Storage*, SC26-7407
- *MVS/ESA Storage Management Library: Managing Data*, SC26-7397
- *MVS/ESA Storage Management Library: Managing Storage Groups*, SC35-0421
- *MVS Storage Management Library: Storage Management Subsystem Migration Planning Guide*, GC26-7398

System Network Architecture (SNA)
- *SNA Formats*, GA27-3136
- *SNA LU 6.2 Peer Protocols Reference*, SC31-6808
- *SNA Transaction Programmer's Reference Manual for LU Type 6.2*, GC30-3084
- *SNA/Management Services Alert Implementation Guide*, GC31-6809

TCP/IP
- *IBM TCP/IP for MVS: Customization & Administration Guide*, SC31-7134
- *IBM TCP/IP for MVS: Diagnosis Guide*, LY43-0105
- *IBM TCP/IP for MVS: Messages and Codes*, SC31-7132
- *IBM TCP/IP for MVS: Planning and Migration Guide*, SC31-7189

TotalStorage™ Enterprise Storage Server
- *RAMAC Virtual Array: Implementing Peer-to-Peer Remote Copy*, SG24-5680
- *Enterprise Storage Server Introduction and Planning*, GC26-7444
- *IBM RAMAC Virtual Array*, SG24-6424

Unicode
- *z/OS Support for Unicode: Using Conversion Services*, SA22-7649

Information about Unicode, the Unicode consortium, the Unicode standard, and standards conformance requirements is available at www.unicode.org

VTAM
- *Planning for NetView, NCP, and VTAM*, SC31-8063
- *VTAM for MVS/ESA Diagnosis*, LY43-0078
- *VTAM for MVS/ESA Messages and Codes*, GC31-8369
- *VTAM for MVS/ESA Network Implementation Guide*, SC31-8370
- *VTAM for MVS/ESA Operation*, SC31-8372
- *z/OS Communications Server SNA Programming*, SC31-8829
- *z/OS Communicatons Server SNA Programmer's LU 6.2 Reference*, SC31-8810
- *VTAM for MVS/ESA Resource Definition Reference*, SC31-8377

WebSphere family
- *WebSphere MQ Integrator Broker: Administration Guide*, SC34-6171
- *WebSphere MQ Integrator Broker for z/OS: Customization and Administration Guide*, SC34-6175
- *WebSphere MQ Integrator Broker: Introduction and Planning*, GC34-5599
- *WebSphere MQ Integrator Broker: Using the Control Center*, SC34-6168

z/Architecture
- *z/Architecture Principles of Operation*, SA22-7832

z/OS
- *z/OS C/C++ Programming Guide*, SC09-4765
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821
- *z/OS C/C++ User's Guide*, SC09-4767
- *z/OS Communications Server: IP Configuration Guide*, SC31-8875
- *z/OS DCE Administration Guide*, SC24-5904
- *z/OS DCE Introduction*, GC24-5911
- *z/OS DCE Messages and Codes*, SC24-5912
- *z/OS Information Roadmap*, SA22-7500
- *z/OS Introduction and Release Guide*, GA22-7502
- *z/OS JES2 Initialization and Tuning Guide*, SA22-7532
- *z/OS JES3 Initialization and Tuning Guide*, SA22-7549
- *z/OS Language Environment Concepts Guide*, SA22-7567
- *z/OS Language Environment Customization*, SA22-7564
- *z/OS Language Environment Debugging Guide*, GA22-7560
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Programming Reference*, SA22-7562
- *z/OS Managed System Infrastructure for Setup User's Guide*, SC33-7985
- *z/OS MVS Diagnosis: Procedures*, GA22-7587
- *z/OS MVS Diagnosis: Reference*, GA22-7588
- *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589
- *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS MVS Installation Exits*, SA22-7593
- *z/OS MVS JCL Reference*, SA22-7597
- *z/OS MVS JCL User's Guide*, SA22-7598
- *z/OS MVS Planning: Global Resource Serialization*, SA22-7600
- *z/OS MVS Planning: Operations*, SA22-7601
- *z/OS MVS Planning: Workload Management*, SA22-7602
- *z/OS MVS Programming: Assembler Services Guide*, SA22-7605
- *z/OS MVS Programming: Assembler Services Reference, Volumes 1 and 2*, SA22-7606 and SA22-7607
- *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608
- *z/OS MVS Programming: Authorized Assembler Services Reference Volumes 1-4*, SA22-7609, SA22-7610, SA22-7611, and SA22-7612
- *z/OS MVS Programming: Callable Services for High-Level Languages*, SA22-7613
- *z/OS MVS Programming: Extended Addressability Guide*, SA22-7614
- *z/OS MVS Programming: Sysplex Services Guide*, SA22-7617
- *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618
- *z/OS MVS Programming: Workload Management Services*, SA22-7619
- *z/OS MVS Recovery and Reconfiguration Guide*, SA22-7623
- *z/OS MVS Routing and Descriptor Codes*, SA22-7624
- *z/OS MVS Setting Up a Sysplex*, SA22-7625
- *z/OS MVS System Codes* SA22-7626
- *z/OS MVS System Commands*, SA22-7627
- *z/OS MVS System Messages Volumes 1-10*, SA22-7631, SA22-7632, SA22-7633, SA22-7634, SA22-7635, SA22-7636, SA22-7637, SA22-7638, SA22-7639, and SA22-7640
- *z/OS MVS Using the Subsystem Interface*, SA22-7642
- *z/OS Planning for Multilevel Security and the Common Criteria*, SA22-7509
- *z/OS RMF User's Guide*, SC33-7990
- *z/OS Security Server Network Authentication Server Administration*, SC24-5926
- *z/OS Security Server RACF Auditor's Guide*, SA22-7684
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Security Server RACF Macros and Interfaces*, SA22-7682
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS Security Server RACF System Programmer's Guide*, SA22-7681
- *z/OS Security Server RACROUTE Macro Reference*, SA22-7692
- *z/OS Support for Unicode: Using Conversion Services*, SA22-7649
- *z/OS TSO/E CLISTs*, SA22-7781
- *z/OS TSO/E Command Reference*, SA22-7782

- *z/OS TSO/E Customization*, SA22-7783
- *z/OS TSO/E Messages*, SA22-7786
- *z/OS TSO/E Programming Guide*, SA22-7788
- *z/OS TSO/E Programming Services*, SA22-7789
- *z/OS TSO/E REXX Reference*, SA22-7790
- *z/OS TSO/E User's Guide*, SA22-7794
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS UNIX System Services Messages and Codes*, SA22-7807
- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803
- *z/OS UNIX System Services User's Guide*, SA22-7801

Index

Readers' Comments — We'd Like to Hear from You

DB2 Universal Database for z/OS
Installation Guide
Version 8

Publication No. GC18-7418-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name	Address
Company or Organization	
Phone No.	

Readers' Comments — We'd Like to Hear from You
GC18-7418-04

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines
Corporation
Reader Comments
DTX/E269
555 Bailey Avenue
San Jose, CA 95141-9989
U. S. A.

Fold and Tape **Please do not staple** Fold and Tape

GC18-7418-04

Cut or Fold
Along Line

Program Number: 5625-DB2

Printed in USA

GC18-7418-04