IBM DB2 Universal Database

Administration Getting Started

Version 5

S10J-8154-00

IBM DB2 Universal Database

Administration Getting Started

Version 5

S10J-8154-00

Before using this information and the product it supports, be sure to read the general information under Appendix G, “Notices” on page 123.

Order publications through your IBM representative or the IBM branch office serving your locality or by calling 1-800-879-2755 in U.S. or 1-800-IBM-4YOU in Canada.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Welcome

This book will teach you:

- How to use the DB2 administration tools to administer a single, local database. You access the tools from the Control Center. For information about multi-node databases, see the *Administration Guide*.
- Basic DB2 administration concepts
- To perform basic DB2 administration tasks covering the areas of:
  - Database implementation
  - Database management
  - Database configuration and tuning to improve performance

The *Administration Guide* covers all administration topics at a more advanced level.

Another interface you can use to perform administration tasks, which is not described in this book, is the application programming interface (API). It allows you to execute DB2 utility functions within an application program using SQL statements or DB2 APIs. For more information about using the application programming interface, see the *Embedded SQL Programming Guide* and the *API Reference* book. You can also use the command line processor (accessible from its own icon in the DB2 product folder) to execute SQL statements and DB2 commands. It is described in the *SQL Reference*.

Before you perform any of the actions described in this book, you should:

- Install and configure the server and install the SAMPLE database as outlined in the *Quick Beginnings* book for your platform. It is recommended that you do not put your own data into the DB2 SAMPLE database.
- Create the user ID following the instructions in the *Quick Beginnings* for your platform. That user ID should have SYSADM authority.

Use the table below to help you quickly find the part of the book you are interested in.

Part 1, “Tools and Basic Concepts” on page 1	Provides an overview of the tools to use to manage a DB2 system, shows you how to create a basic database and work with objects, and describes the components of a DB2 database.
Part 2, “Preparing a Database For Use” on page 25	Explains how to: 1. Add a table space 2. Create a table 3. Move data into a database 4. Enforce rules for the data 5. Collect statistics 6. Back up a database 7. Support applications 8. Control access to the data
Part 3, “Database Operation” on page 59	Explains more about how to protect data, how to recover data, and some basics of DB2 performance.

Conventions

This book uses these highlighting conventions:

- **Boldface type** indicates commands or graphical user interface (GUI) controls such as names of fields, folders, icons, or menu choices.
- `Monospace type` indicates examples of text you enter exactly as shown.
- *Italic type* indicates variables that you should replace with a value. It is used also to indicate book titles and to emphasize words.

Part 1. Tools and Basic Concepts

Chapter 1. DB2 Administration Tools and Basic Concepts

This chapter presents:

- The basics of how to use the Control Center
- A description of the components of the Control Center interface
- An overview of some Control Center objects and their relationships with each other

The **Server Administration** folder contains the Control Center from which you can access tools to help you administer DB2 servers. The tools are installed by default on:

- Servers running the Windows NT or OS/2 operating systems
- Workstations running the Windows NT, Windows 95, or OS/2 operating systems

The tools are optionally installable on any client system running the:

- Windows NT operating system
- Windows 95 operating system
- OS/2 operating system

If you wish to have a dedicated database administrator's (DBA) system, which allows you to administer remote DB2 databases, you can install the tools on one of the client systems.

In addition to this book, you will find extensive online help accompanying the tools for all the administration tasks. For details on all other administration functions available in DB2, see the *Administration Guide*.

Knowledge of basic SQL is useful but not required. See the *SQL Getting Started* book if you need to get started using SQL.

Using the Control Center

The Control Center contains tools for performing common database administration tasks. It provides seamless integration of the DB2 administration tools, gives you a clear view of all managed systems, lets you manage databases remotely, and provides step-by-step assistance for some tasks.

This section covers the following topics:

- Opening the Control Center
- Displaying all the systems to which yours is connected

- Creating a basic database
- Creating and working with objects below the database level

Opening the Control Center

To open the Control Center:

1. From the desktop, open the DB2 product folder. Depending on the system you are running, for example the Windows NT operating system, it could be called **DB2 for Windows NT**. On systems running the Windows NT 4.0 and Windows 95 operating systems, it is found in the **Start** menu.
2. Open the **Server Administration** folder. (You do not perform this step on systems running the Windows NT 3.51 operating system.)
3. Double-click on the **Control Center** icon. The Control Center opens.

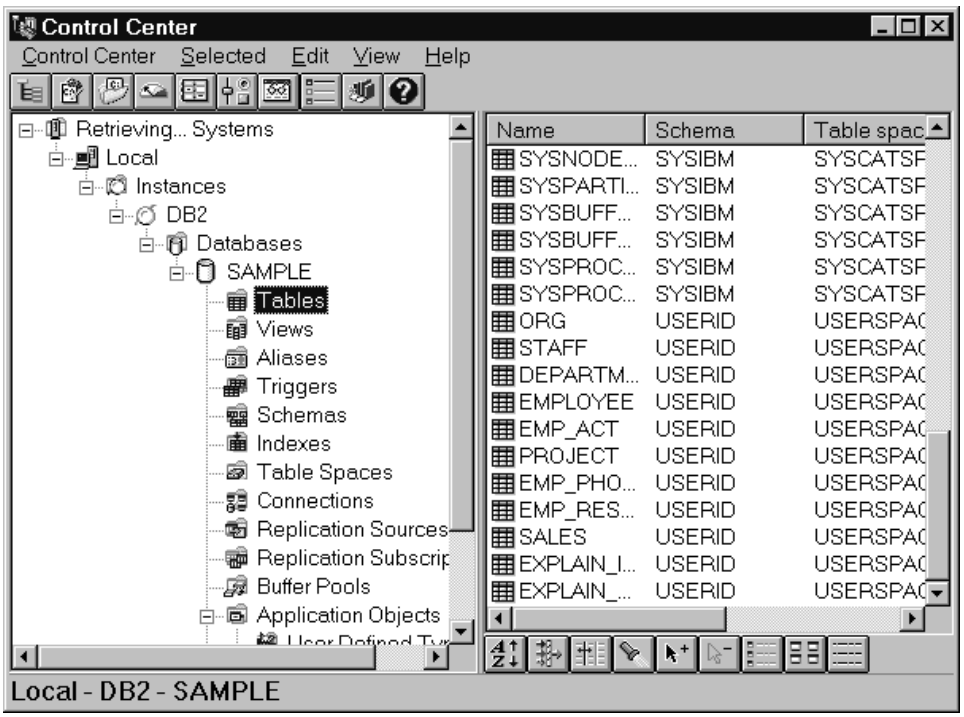

Figure 1. DB2 Control Center

The interface of the Control Center contains the following features:

- The left side of the window is called the object tree. It displays icons representing servers and database objects such as tables and views and their relationships. You can expand and collapse the object tree.

- The right side of the window is called the contents pane. When you open an object folder in the object tree, specific objects are displayed here.
- The menu bar along the top of the window lets you access Control Center functions and online help.

Figure 2. Control Center Menu Bar

Menu items under **Control Center** include actions that affect the entire Control Center. (The **Control Center** menu changes dynamically if you open a new Control Center.) Menu items under **Selected** dynamically change to include only those actions that apply to selected objects in the contents pane. Menu items under **Edit** include actions that let you work with objects in the contents pane. Menu items under **View** include those that let you customize the display of objects in the contents pane, and invoke some windows such as the Journal and the Alert Center. Menu items under **Help** include actions to display online help information and the features of the online help information.

- The toolbar located above the object tree contains icons that let you access Control Center functions such as creating and working with scripts, executing commands, viewing job status, logs, and messages, and setting preferences and alerts. These functions can also be selected in the **View** menu. Hover help provides a short description for each icon. (When you move your mouse pointer over an area of an icon, and pause, the brief description that appears is called hover help.)

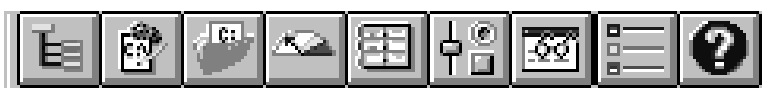

Figure 3. Control Center Toolbar

- The toolbar located below the contents pane contains icons that let you tailor the view of objects and information in the contents pane to suit your needs. These toolbar functions can also be selected from the **View** menu. A toolbar and menu items similar to the **View** menu are seen throughout the DB2 administration tools when you work with lists. Hover help provides a short description for each icon.

Figure 4. Control Center Contents Pane Toolbar

At any time, you can set preferences for the DB2 administration tools by clicking on the **Tools Settings** icon in the toolbar or the menu choice in the **View** menu to open the **Tools Settings** notebook.

Displaying Systems

To display all of the systems that your system is connected to and which have DB2 installed:

1. Expand the object tree by clicking on the plus sign (+) beside **Systems**. Icons representing the actual local machine and any remote machines are displayed.

   (Your local system is represented by the icon labelled **Local**. It appears only if the local machine is a DB2 server. If you click with mouse button 2 on the **Local** icon, one of the options in the pop-up menu is called **Connect to administration server**. The Administration Server lets you take advantage of functions such as performance monitoring and scheduling. It is used as a service by the DB2 Administration Tools to satisfy operating system requests and it is automatically created and started for you (its default name is DB2DAS00).)

2. Expand the **Local** icon. The instance of DB2 on the local machine is displayed in a tree structure.

   Each copy of the database manager code can be thought of as a separate *instance*, and is stored in a directory on your machine. The database manager is DB2 code that manages data. A default local instance is created when you install DB2. In this book you will create and work with a single database under the default instance.

   (You can have several instances on a single system. You can use these instances to separate the development environment from the production environment, or to restrict sensitive information to a particular group of people. You can also tune an instance for a particular environment.)

3. Expand the **Instances** icon. For each database that exists, an icon and the name are displayed.

   A *relational database* is a collection of data that is stored in tables. Database objects such as views and indexes are provided to help manage the data. For each database, an entry exists in the Database Directory file.

Figure 5 on page 7 illustrates the concepts described in this section.

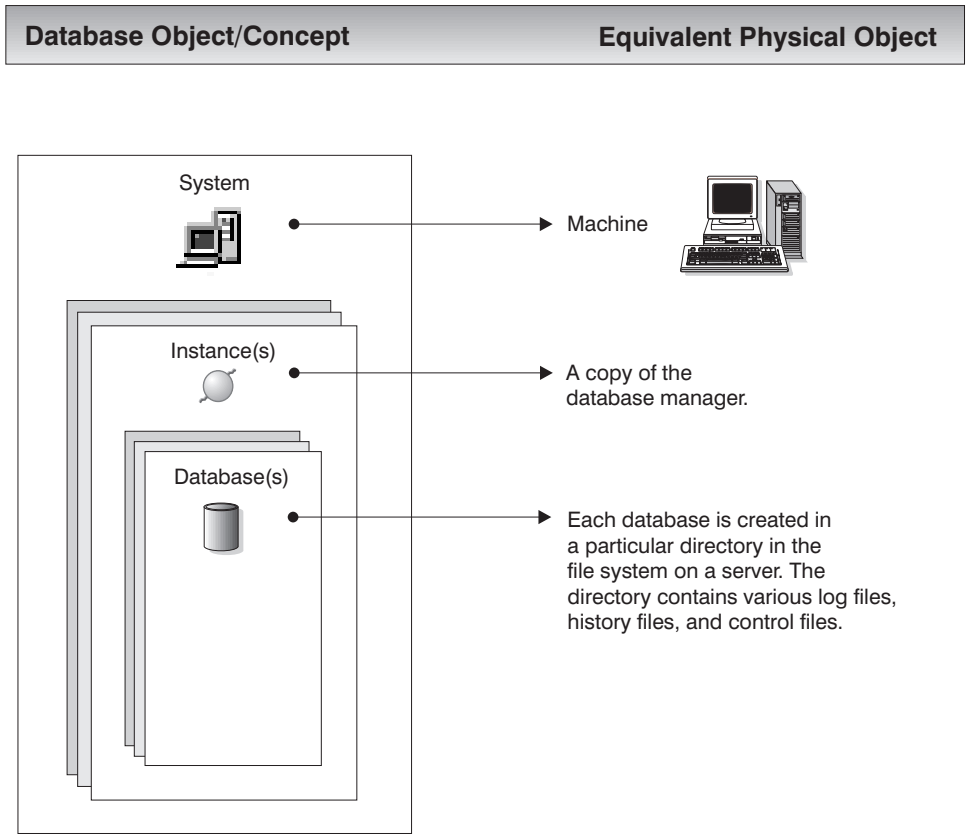

Figure 5. Database Logical View and Equivalent Physical View

Creating a Basic Database

To quickly create a basic database:

1. Click mouse button 2 on the **Databases** icon and select **Create -> New** from the pop-up menu. (Mouse button 2 is the right mouse button on a right-handed mouse.) The Create Database SmartGuide opens.

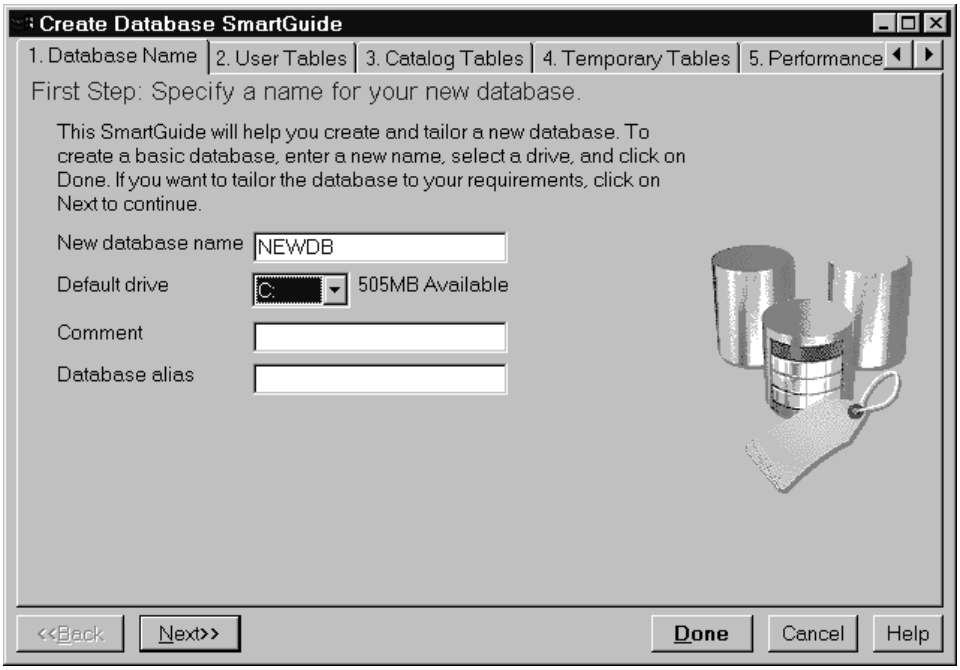

Figure 6. Create Database SmartGuide

2. Enter a name for the database, select a drive, and click on **Done**. If you do not enter an alias (local name), the database name will be used as the alias.

Later you can use the Create Database SmartGuide again to create other databases where you can:

1. Allocate space for the database
2. Specify basic storage performance characteristics
3. Specify the locale for the correct selection of code pages for sorting and code page conversion

More options exist for creating a database. For now, a simple database is sufficient to illustrate how to use the Control Center.

To bring up a pop-up menu that shows all the actions you can perform on a database, click on it with mouse button 2. Many of the actions are described in more detail in this book.

Creating and Working with Objects

You can create and work with objects below the database level such as tables, views, and indexes.

To create new objects:

1. Expand the database icon of the database you just created. Object types are displayed as folder icons.
2. Click mouse button 2 on a folder icon, such as **Tables**. The pop-up menu is displayed.

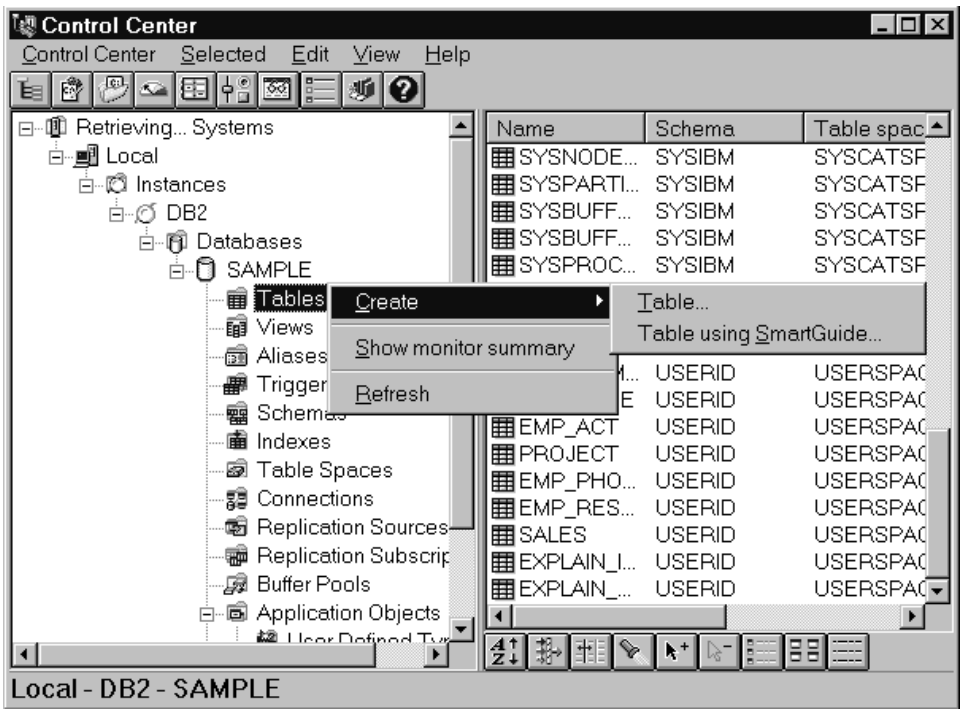

Figure 7. Pop-up Menu for Tables

3. Select **Create**. A Create window or Create SmartGuide opens, depending on the create action you choose.

From the object tree, you can click on a folder icon such as **Tables**. If any tables exist, they are displayed in the contents pane. You can then select a table in the contents pane and invoke an action for it.

For more information about using the Control Center, go to its online help (available from its **Help** menu or by pressing F1 anywhere in the Control Center). For example, you will find instructions on:

- Opening a new Control Center
- Finding an object in the contents pane

Using the SmartGuides

The DB2 SmartGuides are part of the DB2 administration tools and step you through the administration tasks. The following SmartGuides are available:

- Create Database: Helps you create a database, assign storage, and select basic performance options. To invoke it, click mouse button 2 on the **Databases** icon in the Control Center, and select **Create -> New**.
- Create Table Space: Helps you create a new table space and set basic storage performance options. To invoke it, click mouse button 2 on the **Table spaces** icon in the Control Center, and select **Create -> Table space using SmartGuide**.
- Create Table: Helps you design columns (using pre-defined column templates if you wish), create a primary key for the table, and assign the table to one or more table spaces. To invoke it, click mouse button 2 on the **Tables** icon in the Control Center, and select **Create -> Table using SmartGuide**.
- Backup Database: Asks you basic questions about the data in the database, the database's availability, downtime restrictions, and recoverability requirements. It then suggests a backup plan, creates the job script, and schedules it. To invoke it, click mouse button 2 on an icon representing one of your databases in the Control Center, and select **Backup -> Database using SmartGuide**.
- Restore Database: Walks you through the process of recovering a database. To invoke it, click mouse button 2 on an icon representing one of your databases in the Control Center, and select **Restore -> Database using SmartGuide**.
- Performance Configuration: Asks you questions about the database, its data, and the purpose of the system and then suggests new configuration parameters for the database and automatically applies them to the database if you wish. To invoke it, click mouse button 2 on an icon representing one of your databases in the Control Center, then select **Configure performance**.

Overview of Basic Database Objects

This section provides an overview of the key database objects that are displayed in the Control Center and that you need to understand to perform the tasks in this book. In this book you will create a:

- Table
- View
- Index

The following objects also are displayed in the Control Center and are described in other documentation that is shipped with the DB2 product (that is, they are not described in this book):

- Triggers
- User-defined types (UDTs)
- User-defined functions (UDFs)
- Packages
- Aliases
- Replication objects
- Users and groups

See the *Road Map to DB2 Programming* for an overview of triggers, UDTs, UDFs, and packages. See the *Administration Guide* for an overview of aliases. See the *Replication Guide and Reference* for an overview of replication objects. See the *Quick Beginnings* book for your platform and the *Administration Guide* for an overview of users and groups.

Tables

A relational database presents data as a collection of tables. A *table* consists of data logically arranged in columns and rows. The data in the table is logically related, and relationships can be defined between tables. Data can be viewed and manipulated based on mathematical principles and operations called *relations*. Table data is accessed via SQL, a standardized language for defining and manipulating data in a relational database.

Views

A *view* is an efficient way of representing data without needing to maintain it. A view is not an actual table and requires no permanent storage. A “virtual table” is created and used.

A view can include all or some of the columns or rows contained in the tables on which it is defined. For example, you can join a department table and an employee table in a view so that you can list all employees in a particular department.

Figure 8 shows the relationship among tables and views.

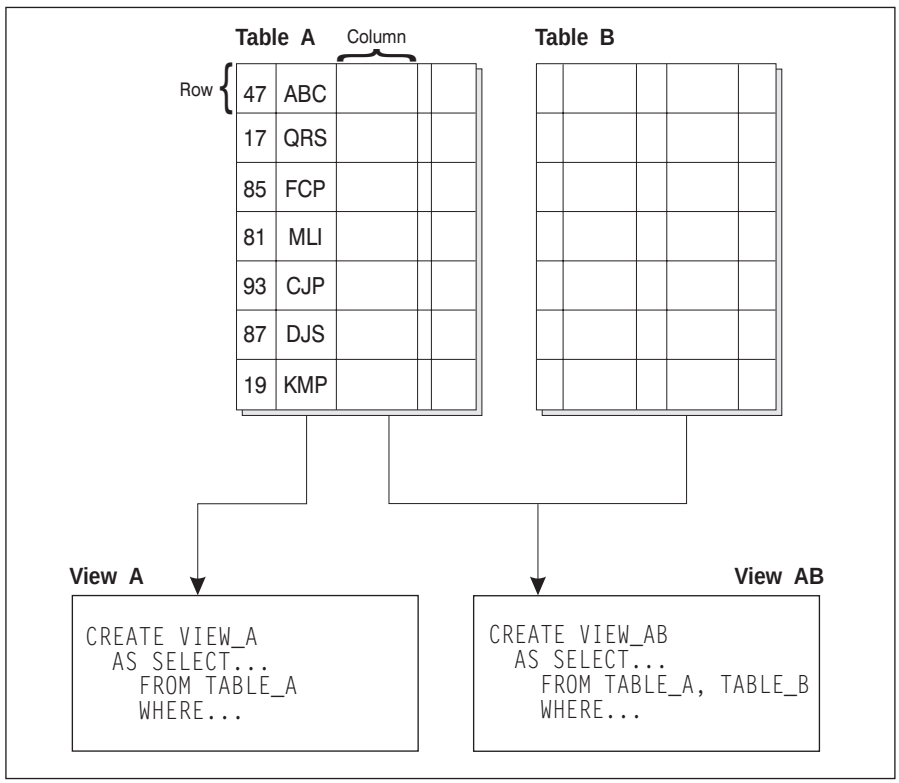

Figure 8. Relationship Among Tables and Views

Indexes

An *index* is a set of keys, each pointing to rows in a table. For example, table A in Figure 9 on page 13 has an index based on the employee numbers in the table. This key value provides a pointer to the rows in the table: employee number 19 points to employee KMP. An index allows more efficient access to rows in a table by creating a direct path to the data through these pointers.

The SQL *optimizer* automatically chooses the most efficient way to access data in tables. The optimizer takes indexes into consideration when determining the fastest access path to data. See the *Administration Guide* for more information.

Unique indexes can be created to ensure uniqueness of the index key. An *index key* is a column or an ordered collection of columns on which an index is defined. Using a unique index will ensure that the value of each index key in the indexed column or columns is unique. “Step 3. Enforcing Business Rules for Data” on page 39 describes keys and indexes in more detail.

Figure 9 shows the relationship between an index and a table.

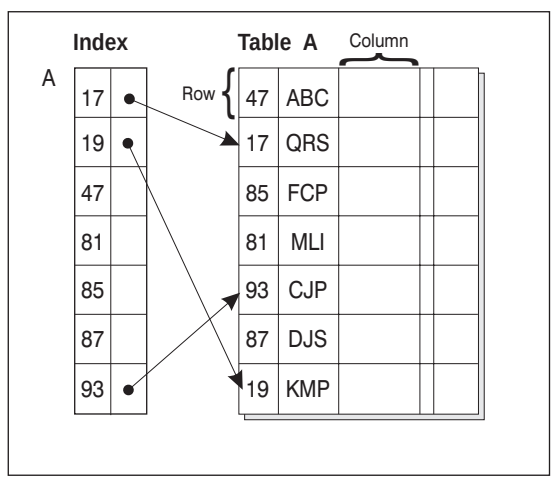

Figure 9. Relationship Between an Index and a Table

System Catalog Tables

Each database includes a set of system catalog tables, which describe the logical and physical structure of the data. DB2 creates and maintains an extensive set of system catalog tables for each database. These tables contain information about the definitions of the database objects such as user tables, views, and indexes, as well as security information about the authority that users have for these objects. They are created when the database is created, and are updated in the course of normal operation. You cannot explicitly create or drop them, but you can query and view their contents using the catalog views. For more information, see the *Administration Guide* and the *SQL Reference*.

Overview of Recovery Objects

Log files and the recovery history file are created automatically when a database is created. Figure 10 illustrates the relationship. You cannot modify a log file or the recovery history file; however they are important to you should you need to use your database backup to recover data that is lost or damaged.

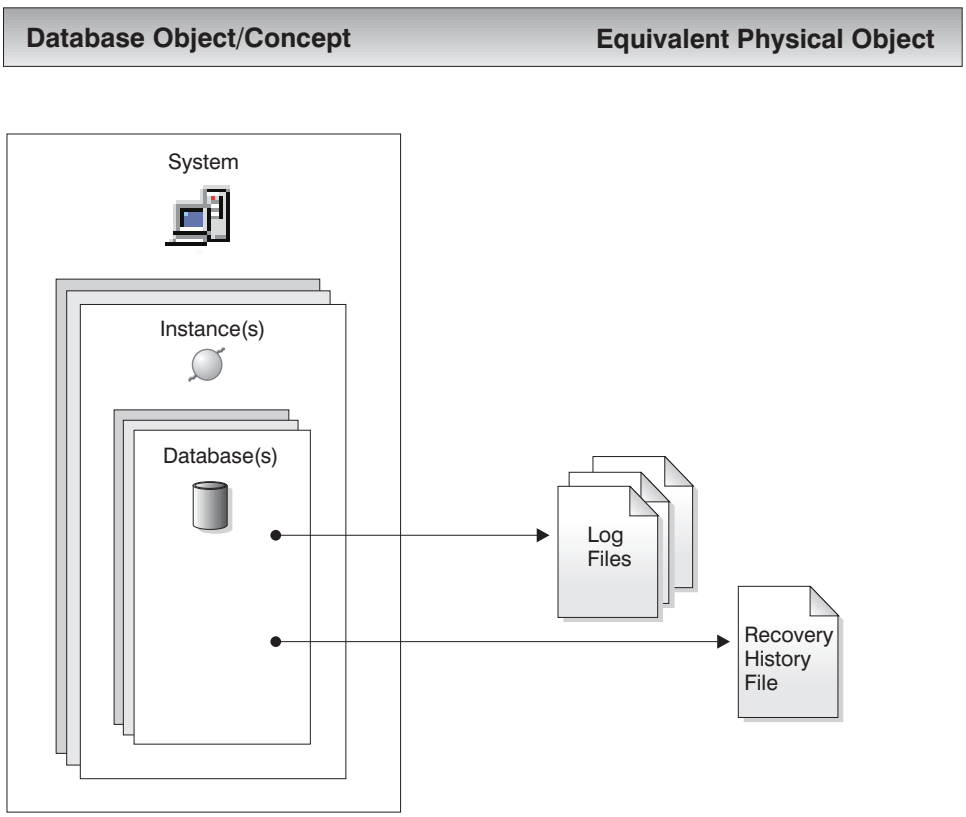

Figure 10. Log Files and Recovery History File

Log Files

Each database includes recovery logs which are used to recover from application or system errors. In combination with the database backups, they are used to recover the consistency of the database right up to the point in time when the error occurred.

Database backups and logs are described in more detail in “About the Logs that DB2 Keeps” on page 63. Recovery, discussed in “Facilities for Recovering Data” on page 68, uses logs in an additional way to allow a database to be rebuilt to a specified point-in-time.

Recovery History File

The *recovery history file* contains a summary of the backup information that can be used in case all or part of the database must be recovered to a given point in time. It is used to track recovery-related events such as backups, restores, and loads. The procedure for backing up a database is described in “Step 5. Backing Up a Database for the First Time” on page 46. The procedure for restoring a database is described in “How Do I Perform a Full Database Restore?” on page 69. An overview of the load utility is provided in “Using the Load Utility to Load Data Quickly” on page 38.

A recovery history file is created with each database and is automatically updated when certain actions are performed, such as:

- Database (or table space) backup
- Database (or table space) restore

See “Viewing the Recovery History File” on page 48 for the steps for viewing the contents of the file. For a more detailed discussion, see the *Administration Guide*.

Overview of Storage Objects

In addition to the Control Center objects already described, the following database objects are important because they let you define how you will store the data on your system and how you can improve performance related to accessing the data:

- Table space
- Container
- Buffer pool

You are not required to create a table space, container, or buffer pool to be able to add data to tables in a database. You can accept the defaults for table space, container, and buffer pool when you create a database and a table. However, if you are interested in tuning one of them for your environment, you may find their descriptions in this section useful.

Table Spaces

A database is organized into parts called *table spaces*. Essentially, a table space is a place to store tables. A table space can be either a system managed space (SMS), or a database managed space (DMS). For an SMS table space, each container is a directory in the file space of the operating system, and the operating system's file manager controls the storage space. Containers are described in the next section. For a DMS table space, each container is either a fixed size pre-allocated file, or a physical device such as a disk, and the database manager controls the storage space.

Figure 11 illustrates the relationship of tables to table spaces, and the two types of space: SMS and DMS. It also illustrates that tables, indexes, and long data are stored in table spaces.

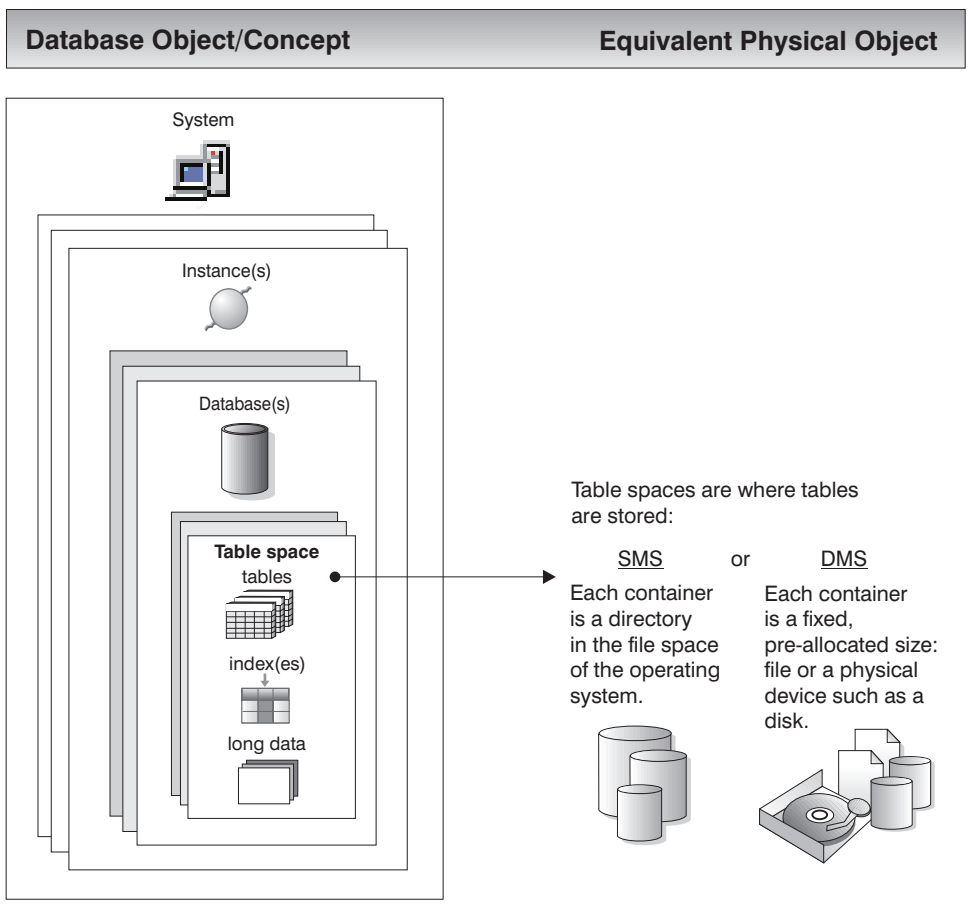

Figure 11. Table Spaces and Tables

Figure 12 on page 17 shows the three table space types: *regular*, *temporary*, and *long*.

Tables containing user data exist in the regular table space(s). By default, a table space called USERSPACE1 is created. Indexes are also stored in regular table space(s). The system catalog tables exist in regular table space(s) as well. The default system catalog table space is called SYSCATSPACE.

Tables containing long field data or long object data, such as multi-media objects, exist in the long table space(s).

The temporary table space is used during SQL operations for things like sorting or reorganizing tables, creating indexes, and joining tables. A database should have one temporary table space. (You can create more, although for most situations only one is required.) By default, a table space called TEMPSPACE1 is created.

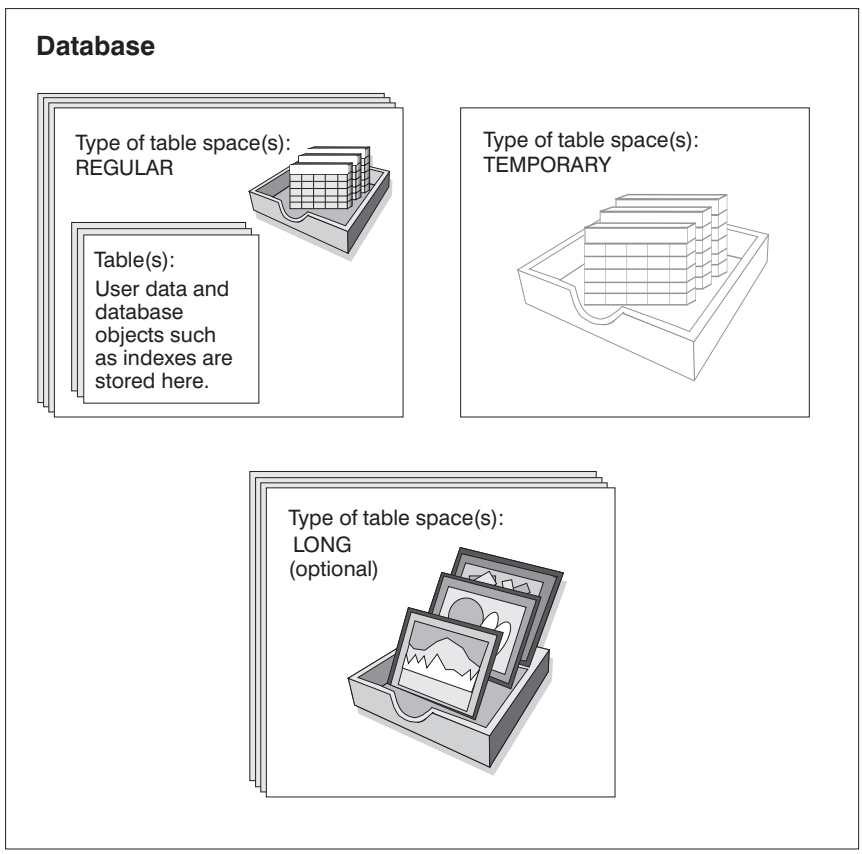

Figure 12. Three Table Space Types

Containers

A *container* is a physical storage device. It can be identified by a directory name, a device name, or a file name.

A container is assigned to a table space. All database and table data is assigned to table spaces. A table space's definitions and attributes are recorded in the database system catalog. Once a table space is created, you can then create tables within this table space.

A container is not explicitly shown in the Control Center but it is listed when you view table spaces.

A single table space can span many containers, but each container can belong to only one table space.

Figure 13 shows an example of the relationship between tables and a table space within a database, and the associated containers and disks.

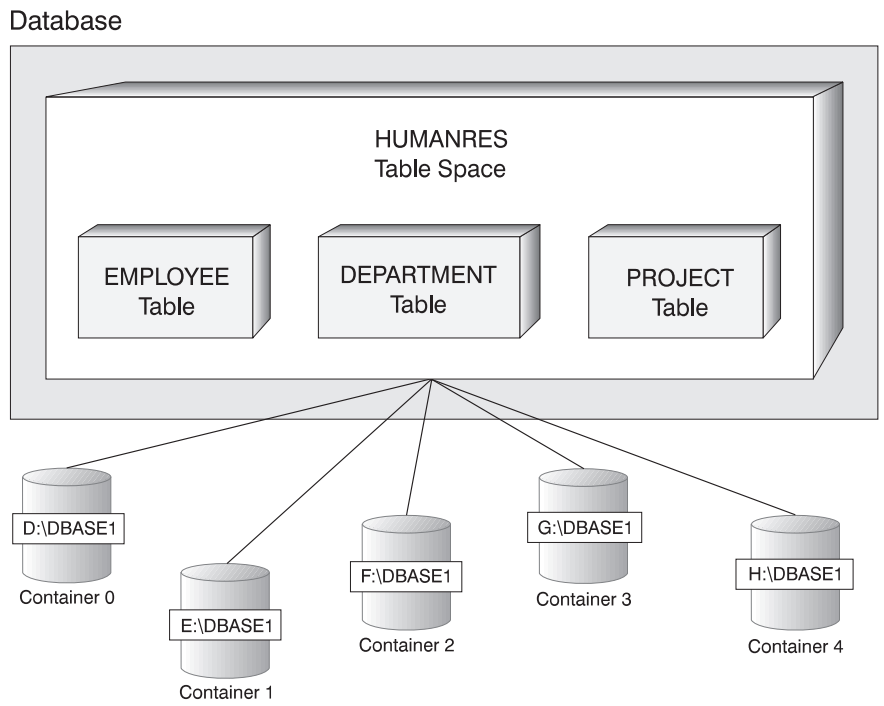

Figure 13. Table Spaces and Tables Within a Database

The EMPLOYEE, DEPARTMENT, and PROJECT tables are in the HUMANRES table space which spans containers 0, 1, 2, 3, and 4. This example shows each container existing on a separate disk.

Data for any table will be stored on all containers in a table space in a round-robin fashion. This balances the data across the containers that belong to a given table space. The number of pages that the database manager writes to one container before using a different one is called the *extent size*.

Buffer Pool

A *buffer pool* is an allocation of main memory allocated to cache table and index data pages as they are being read from disk or being modified. The purpose of the buffer pool is to improve database system performance. Data can be accessed much faster from memory than from a disk; therefore, the fewer times the database manager needs to read from or write to a disk, the better the performance. (You can create more than one buffer pool, although for most situations only one is required.)

The configuration of the buffer pool is the single most important tuning area, since you can reduce the delay caused by slow physical I/O.

Figure 14 on page 20 illustrates the relationship of a buffer pool and containers to a system, a database, and a table space.

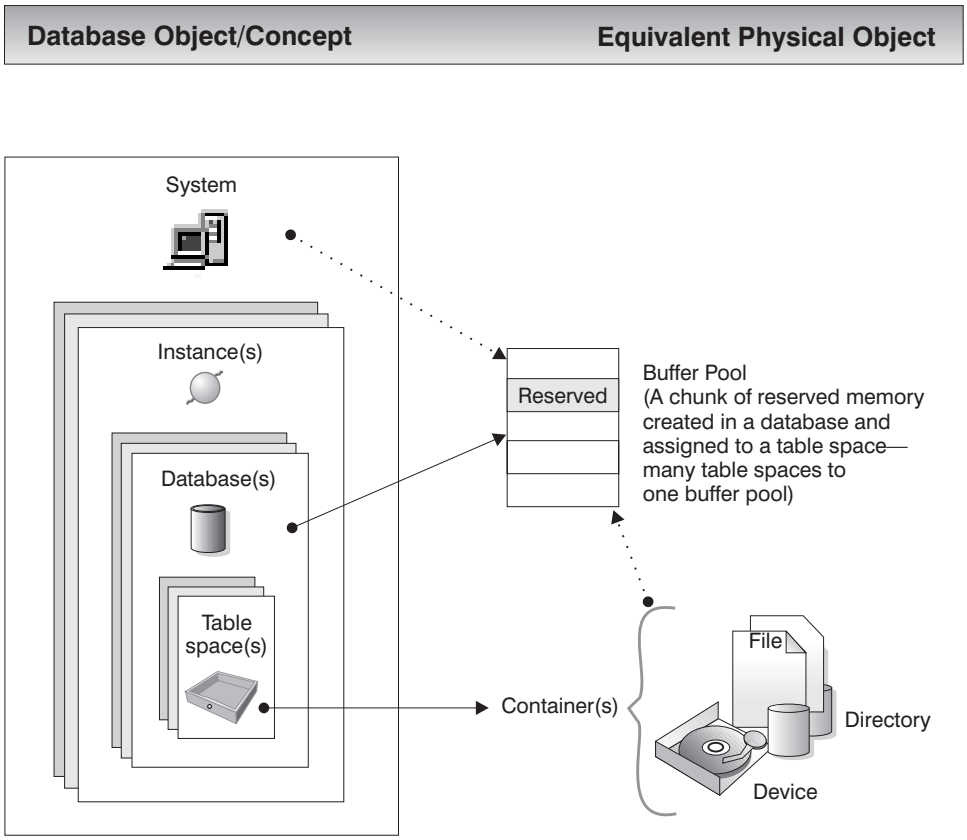

Figure 14. Buffer Pool and Containers

Overview of System Objects

When a database instance or a database is created, a corresponding configuration file is created with default parameter values. You can modify the parameter values to improve performance.

Configuration Parameters

Configuration files contain parameters that define values such as the resources allocated to the DB2 products and to individual databases, and the diagnostic level. There are two types of configuration files: the database manager configuration file for each DB2 instance and the database configuration file for each individual database.

The *database manager configuration file* is created when an instance of DB2 is created. The parameters it contains affect system resources at the instance level, independent of any one database that is part of that instance. Many of these parameters can be changed from the system default values to improve performance or increase capacity, depending on your system's configuration.

There is one database manager configuration file for each installation of a client as well. This file contains information about the client enabler for a specific workstation. A subset of the parameters available for a server are applicable for the client.

A *database configuration file* is created when a database is created, and resides where that database physically resides. There is one configuration file per database. Its parameters specify, among other things, the amount of resources to be allocated to that database. Many of the parameters can be changed to improve performance or increase capacity. Different changes may be required depending on the type of activity in that specific database.

Figure 15 on page 22 illustrates the relationship of the two configuration files with an instance and a database. As well, one or more configuration files exist which are unique to the operating system.

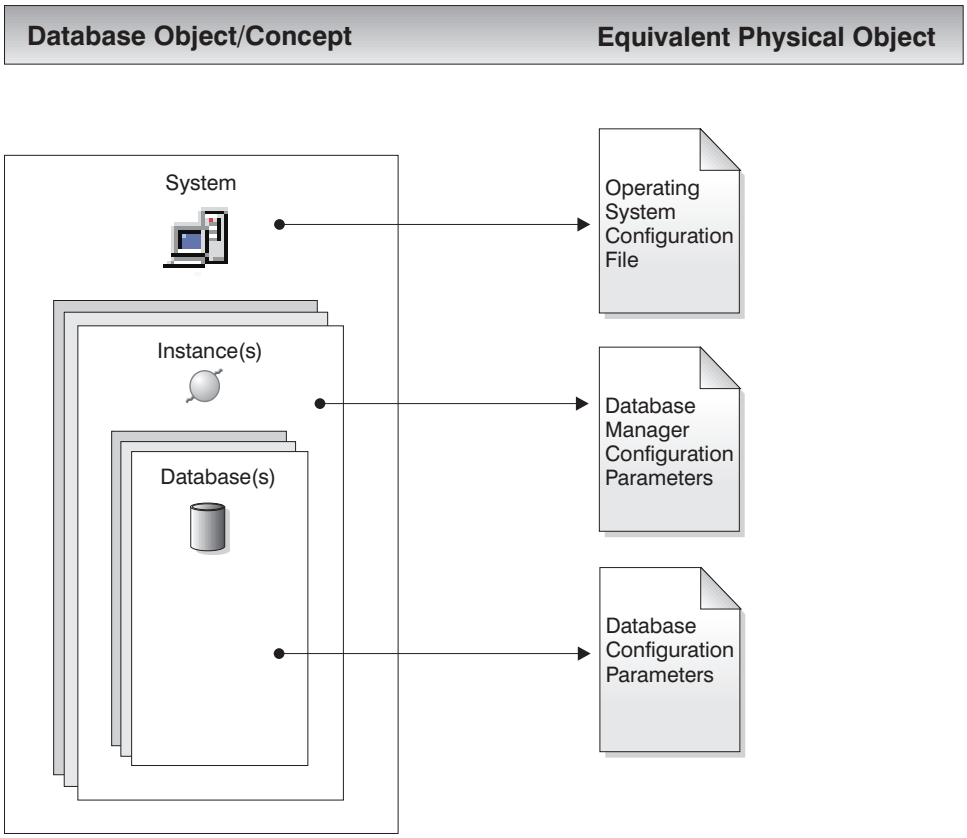

Figure 15. Configuration Parameter Files

Modifying Configuration Parameters

To modify or view a database configuration parameter, click mouse button 2 on the database for which you want to modify configuration parameters and select **Configure** from the pop-up menu. The Configure Database window opens.

The window displays all configuration parameters grouped into categories. You can browse all the parameters. For each parameter, the following can be viewed:

- The default value
- The current value
- Hints on how to set the parameter

The push buttons let you set all values for a parameter to their defaults, or to reset them to their previous values.

To modify a database manager configuration parameter, select the same action from the instance of the database manager you want to modify. In order for the new values to take effect, you must stop and then start the instance.

Part 2. Preparing a Database For Use

Chapter 2. More About Creating a Database

This chapter presents:

- A list of the events that occur automatically when you create a database
- An overview of the options the Create Database SmartGuide provides in order for you to tune a database to your requirements

It is recommended that you do some basic logical and physical database design if you plan to use a database to store a large amount of data or if you know that the data in the database will continue to grow. Basic logical design considerations include deciding what data to store in the database and deciding what tables to create. Basic physical design considerations include determining the physical storage to use to store data and estimating space requirements for tables. (It is assumed that you know the physical characteristics of your system.)

The *Administration Guide* provides information about these topics. In addition, there are many retail books available on this subject.

Events Associated with Creating a Database

When you create a database, the following events automatically occur:

- All the system catalog tables are created
- The database recovery log is allocated
- The database configuration file is created and its default values set
- The database is cataloged in the system database directory file
- The default table spaces are created
- Finally, the DB2 utilities are bound to the database. This makes the utilities available for use. If you wish to use these utilities from a client, and the client is running a different operating system than the server or is at a different service level, you must bind them explicitly. Binding a utility creates a *package*, which is an object that includes all the information needed to process specific SQL statements from a single source file. See the *Quick Beginnings* for your platform for more information about binding and packages.

Unless you override the defaults, the user tables, system tables, and temporary tables will be stored on the same drive you selected in the SmartGuide. The database configuration file and the database recovery log will also be stored on the same drive.

SmartGuide Options for Tuning a Database

To tune a database to your requirements, use the **Next** push button in the Create Database SmartGuide to progress through the pages of the SmartGuide. You can specify:

- Space allocation for:
  - A user space (for the tables that will contain data)
  - The system catalog tables (described in “System Catalog Tables” on page 13)
  - Temporary space (required for things like sorting or reorganizing tables, creating indexes, and joining tables)
- Basic storage performance characteristics which alter how the database reads and writes data to the disk and optimize how the database uses its available hard drives
- The locale (territory and code page set) which determines the set of characters your database will use

Adding an Additional Table Space

The default user table space, USERSPACE1, is fine to use for simple tables that you create for experimental purposes. However, you should create your own table space for “real” tables.

To create a new table space:

1. From the Control Center, click mouse button 2 on the **Table Spaces** folder icon, and select **Create -> Table space using SmartGuide**. The Create Table Space SmartGuide opens.

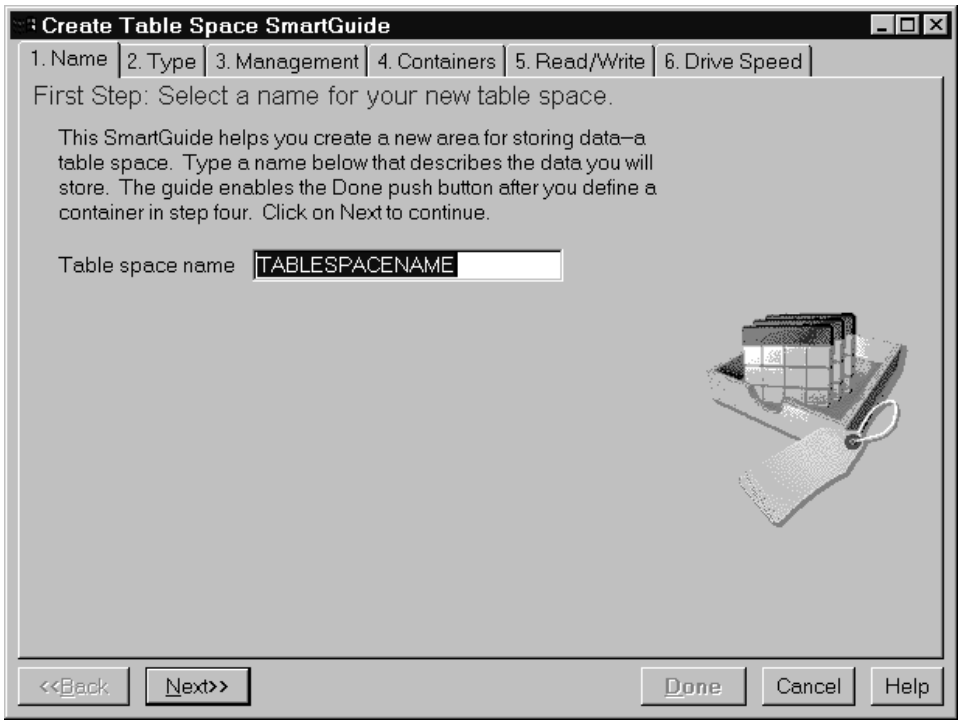

Figure 16. Create Table Space SmartGuide

2. If you just need a basic table space:
   a. Enter a new name
   b. Define a container for the table space on the **Containers** page of the SmartGuide.
   c. Click on the **Done** push button
3. Optional: If you want to specify more details about the table space, click on the **Next** or **Back** push buttons. For example, you can choose:
   - The type of data you will store in the table space on the **Type** page of the SmartGuide
   - The type of space management system on the **Management** page of the SmartGuide

   The SmartGuide will step you through panels that prompt you for information. See the online help for explanations of all the choices.

From this point on, if you want a table to belong to a particular table space, you must provide the name of that table space when you create the table. For each table space, a row exists in the SYSCAT.TABLESPACES catalog view. If you do not specify a table

space, the table will be placed in the first table space that was created (that is, the first one listed in SYSCAT.TABLESPACES). If you have not created any table spaces, the table will be placed in the default table space (IBMDEFAULTGROUP, or a table space created by the same user ID that is creating the table, or USERSPACE1, in that order). If none of the default table spaces exist, table creation will fail.

The *Administration Guide* describes more advanced table space design considerations:

- Mapping tables to table spaces
- Choosing an extent size
- Deciding between an SMS or DMS table space

Suggestions for Allocating Storage for a Dedicated Server

We suggest the following for setting up your disks, containers, table spaces, and tables, and their relationships among each other. Note that the suggestions below reflect only one sample implementation which allows you to add more containers later as your data grows. Each disk is used as a raw device (unpartitioned drive). This suggested implementation may not apply to a large database configuration.

- After you have installed the operating system, DB2, and any other software, identify the unused disks (for example , three disks) that you want to allocate to the database.
- For best I/O performance, make each container a separate device. Define no more than 1 container per disk. For example, 5 containers on 5 separate disks will allow for efficient scanning of data.
- Create one table space for tables containing user data and their indexes.

  By default, the user, temporary, and system catalog tables belong to separate table spaces. They are created on your default drive.
- Assign at least 3 containers to the table space for tables containing user data. Assign at least 1 container to the table space for the temporary tables, and at least 1 container to the table space for the system catalog tables.
- Create all user tables in one table space.

Creating a Container

This section describes how to access the Change Container window from which you can create a container.

1. From the Control Center, click mouse button 2 on the **Table Spaces** folder icon, and select **Create ->Table space using SmartGuide**. The Create Table Space SmartGuide opens.
2. Click on the **Management** tab.

3. Select **Low maintenance** to create an SMS container. Select **High performance** to create a DMS container (file or raw device).

   An SMS container uses the file system's services to perform I/O. The attributes of this method are:

   - Easy management
   - Space pre-allocation is not required
   - Table space growth is possible by growing the data files up to the limit of the file system

   A DMS container bypasses the file system's services and accesses the file or disk directly (this is often referred to as using raw I/O). The attributes of this method are:

   - High performance
   - Space pre-allocation is required
   - Table space growth is possible by adding new containers

   In general, small personal databases are easiest to manage with SMS table spaces. If you choose to use DMS table spaces with device containers, you must be prepared to tune and administer your environment.

4. Click on the **Next** push button so that you move to the **Containers** page.
5. Click on the **Add** push button. The Add Container window opens. If you require assistance while creating a container, invoke the online help.

Chapter 3. Preparing a Database for Use

This chapter presents the simplest way to prepare a database for use. Alternate methods are described in the *Administration Guide*.

This chapter presents the following:

1. Creating a table
2. Moving data into a table
3. Enforcing business rules for the data
4. Collecting statistics
5. Backing up a database for the first time

You may perform these tasks in a different order; however, the order shown here is the recommended one.

Step 1. Creating a Table

Before you can create a table, you must ensure that a table space exists and that it has sufficient free space. See “Adding an Additional Table Space” on page 28.

The following example shows you how to use the Create Table SmartGuide. (Or, you can use the Create Table window instead.) To create a table:

1. From the Control Center, click mouse button 2 on the **Tables** folder, and select **Create -> Table using SmartGuide** from the pop-up menu. The Create Table SmartGuide opens.

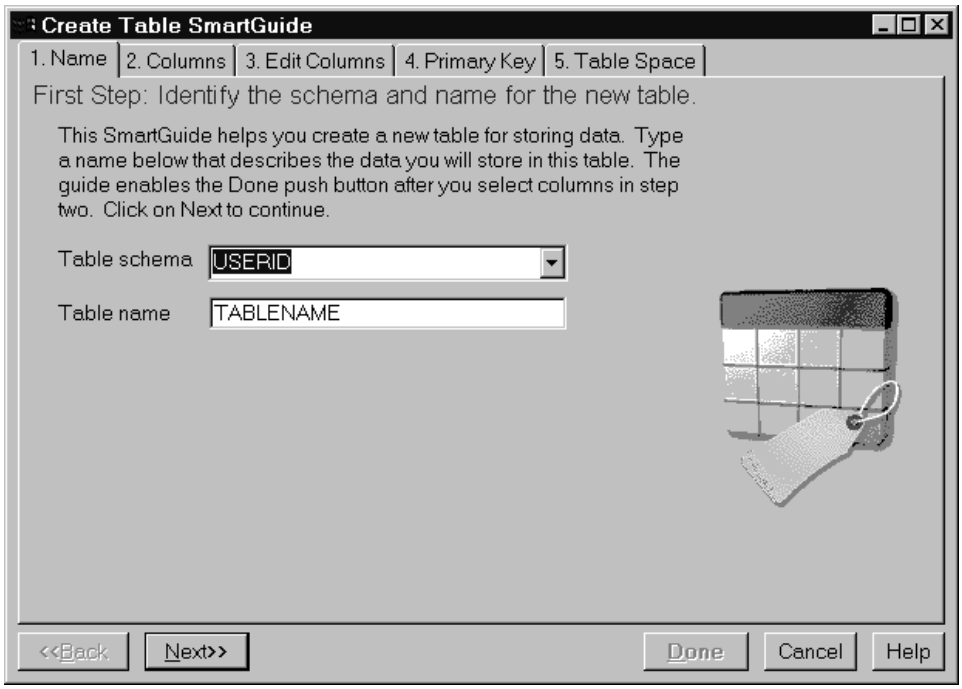

Figure 17. Create Table SmartGuide

The SmartGuide steps you through the task of creating a table, including naming it, selecting, ordering, and editing columns, selecting table spaces, and selecting a primary key. It provides you with templates for columns which you can use as is or modify. The basic steps are described below. Click on **Help** at any time if you are unsure how to complete a step.

2. Identify the schema and a name for the new table.

   A schema provides a logical classification of objects in a database. When a table is created, it is assigned to a schema. See “Creating a Schema” on page 45.

   Before creating your own objects, you need to consider whether you want to assign them all to the default schema (identified by your user ID) or to a separate schema that logically groups them. In many cases, using a different schema name is very beneficial.

3. Select columns for your table. The SmartGuide provides sample column definitions.

   a. Select a column list to see the available columns. Use the **Edit Lists** push buttons to change column defaults to add your own columns to the lists.

   b. Use the > push button to move an available column to the **Columns to create** list.

You can now either select **Done**, or continue on to edit the details of your selected columns with the **Edit lists** push button.

4. Optional: Edit column definitions for the new table.

   Choose the order in which you want the columns to appear when the data in the table is listed.

5. Optional: Define a primary key for the new table.

   Although this step is optional, it is recommended that you define a primary key because table access is quicker if each row can be uniquely identified. A primary key is necessary in order to support updates from many ODBC applications. When you choose a column as the primary key, the database checks each new row for a unique value in that column, rejecting any duplicates. Primary keys are described in “Step 3. Enforcing Business Rules for Data” on page 39. (To create a foreign key, use the Create Table or Alter Table window. You can invoke these windows by clicking with mouse button 2 on a table icon in the Control Center's object tree and selecting the appropriate action from the pop-up menu. Foreign keys are described in “Step 3. Enforcing Business Rules for Data” on page 39.)

6. Optional: Choose space for storing the table data.

   - Table space:

     The easiest option here is to accept the default table space and continue. A common starting point is to store all your tables in one table space. But if this will be a large table, you may want to assign it a separate table space. If necessary, create a new one by clicking on **Create** to open the Create Table Space SmartGuide.

     It is recommended that you create one table space for tables containing user data and their indexes. Create one table space for the temporary tables and one for the system catalog tables.

   - Separate index space:

     This option is not available until you assign a primary table space. Using a separate space for indexes can speed up access to the table. If you have a fast hard disk for which you want to create an index space, you can create and assign the space here. This separation may help make I/O more efficient.

   - Separate long space:

     This option is not available until you assign a primary table space. If this table will have a large amount of long or large object (LOB) data, you can increase the size limit from 64 gigabytes (GB) to 2 terabytes (TB) with a long table space here. There are performance benefits to separating long or LOB data from regular data.

     The SmartGuide will also allow you to backup the long data separately. For example, if the long data is read-only or not accessed frequently, you might choose to back it up on a separate schedule or not at all.

7. Click on the **Done** push button to create the table.

Step 2. Moving Data Into a Table

DB2 provides the import and load utilities to help you move data into a table from existing sources. Another method of moving data into a database is replication, described briefly in “Replicating Data” on page 38.

Manual entry of data into your database can be done either with SQL commands, described in the *SQL Reference*; Lotus Approach, described in the *Using Lotus Approach with DB2* booklet (provided in the product box with DB2); or an application.

The import utility takes data from an input file and inserts it into a table or view. This input file would contain data that was extracted from an existing source of data, such as a Lotus 1–2–3 file or an ASCII file. DB2 imports file formats generated from the supported sources listed in “Importing Data into a Table or View.” However, the procedure for generating each of these files will vary with the original source. For details on generating the appropriate input file format, refer to the documentation accompanying the source product.

You can also use the import utility to re-create a table or view that was saved by using the export utility. See the *Administration Guide* for more information on import and export.

This section provides the steps for importing data. A brief overview of the other two methods is provided.

Importing Data into a Table or View

Once you have an input file available in a supported format, use the **Import** notebook to insert data from the file into an existing table. If this table already contains data, you can either replace or append to the existing data with the data in the file.

You can also use the Import notebook to create a new table that is populated by an input file, or delete existing rows in the selected table and repopulate it using data from the input file.

The online help provides details on how to complete all the optional and required fields.

To import a file into an existing table:

1. From the Control Center, expand the object tree until you find the **Tables** folder.
2. Click on the **Tables** folder. Any existing tables are displayed in the contents pane.
3. Click mouse button 2 on the table you want in the contents pane and select **Import** from the pop-up menu. The **Import** notebook opens with the **File** page displayed.
4. Specify file options.

Use the **File** page of the **Import** notebook to specify the file and type of file you want to import:

a. In the **Import file** field, enter the name of the file that contains the data you want to import.

b. Specify the type of file to import by selecting one of the following radio buttons:

- Non-delimited ASCII format (ASC)

  Non-delimited ASCII data is data that is aligned in columns.

- Delimited ASCII format (DEL)

  Delimited ASCII data is a commonly used way of storing data that separates column values with a user-defined delimiting character such as a comma.

- Worksheet format (WSF)
- Integrated exchange format (IXF)

  PC/IXF is a structured description of a database table or view. Data that was exported in PC/IXF format can be imported or loaded into another DB2 Universal Database product database.

See the online help for the specific products and releases that are supported.

c. Optional: Specify file type modifiers by clicking on the corresponding **Options** push button. The Options window for that format opens.

d. Select the **Import mode** you want to use. The available import modes vary depending on the file type you selected.

e. Optional: In the **Commit records** field, enter the number of records to import before the changes are committed.

f. Optional: In the **Restart** field, enter the number of records in the file to skip before beginning the import.

g. Optional: In the **Compound** field, type a number to specify how many SQL statements will be executed (in an executable block).

h. Optional: Select the **Insert an implied decimal point on decimal data (IMPLIEDDECIMALPOINT)** check box.

i. In the **Message file** field, type the name of the file that will contain warning and error messages that occur during import.

5. Optional: Retrieve large objects from separate files.

   Use the **Large Objects** page of the **Import** notebook to retrieve large objects (LOBs) from the path or paths that store the LOB files:

   a. Click on the **Retrieve large objects (LOBs) in separate files (LOBSINFILE)** check box to enable the options on the **Large Objects** page.

   b. Specify the location of separate LOB files in the LOB paths list box by clicking on the **Add** push button. These paths are searched (in the order in which they

appear in the **LOB paths** list box) for the LOB files specified in the LOB column of the input file.

c. Click on **OK** to accept the defaults on the other notebook pages and begin the import process.

6. Optional: Specify column import options.

   Use the **Columns** page of the **Import** notebook to specify column import options:

   a. Click on one of the radio buttons in the **Include columns by** box to specify the column method that will be used to import data file columns into the table. The available methods vary depending on the file type and mode you selected on the **File** page.

   b. Optional: Specify or change the import file column attributes by clicking on the **Change** push button.

      This option is not available if you selected the **Default (method D)** radio button.

7. Click on **OK** to accept the parameters specified in this notebook and begin the import process.

Using the Load Utility to Load Data Quickly

The load utility loads data directly into a DB2 table from one or more files, tapes and other devices, or named pipes. For example, you might be required to move in data from databases other than DB2. For a description of the functional differences between the load utility and the import utility, see the *Administration Guide*.

During load processing, indexes can be built, primary keys, unique keys, and unique indexes validated, and many statistics generated, all without requiring secondary passes through the data. The load utility is faster than the import utility, and supports the loading of all data types (excluding worksheet format). It is intended for either bulk loading of new tables or appending large amounts of data to existing tables.

Replicating Data

Replication is the process of taking changes stored in the database log at the source server and applying them to the target server. You can use replication to define, synchronize, automate, and manage copy operations for data across your enterprise. You can automatically deliver the data from a host system to target sites. For example, you can copy data and applications to branch offices, retail outlets, and even sales representatives' laptops.

The two operational components in replication are Capture and Apply. The Capture component captures changes made to data in source tables which have been defined for replication by reading the database log. The Apply component reads the changed

data previously captured and stored in a change data table and applies it to the target tables.

Using the Control Center you can do the setup required for replication using the **Define as replication source** and **Define subscription** actions. The replication components Capture and Apply run outside the DB2 administration tools.

Replication administrators can perform the following from the Control Center:

- Define replication sources
- Define replication subscriptions
- Create control tables and target tables
- Specify SQL to enhance data during the apply process

The high-level steps for replicating data are as follows. See the *Replication Guide and Reference* for details.

1. Design a replication scenario (map the source and target tables)
2. Define a replication source (this relates to the capture action):
    a. Specify source columns to capture
    b. Choose replication options
3. Define a replication subscription (this relates to the apply action):
    a. Name the subscription set
    b. Specify the database and target table
    c. Specify the target columns
    d. Specify the row selection
    e. Specify SQL for run-time processing
    f. Set the subscription timing
4. Alter source table with Data Capture Changes option
5. Start Capture to read and store data changes
6. Start Apply to replicate changes to target tables

Step 3. Enforcing Business Rules for Data

Within any business, data often must adhere to certain restrictions or rules. Restrictions may apply to single pieces of information, such as an employee number; association of pieces of data, such as employee number and department; or analysis of the impact of changing data, such as salary increases.

DB2 provides *constraints* as a way to enforce those rules using the database system. A constraint is a mechanism that ensures that certain conditions relating columns and

tables are maintained. For example, an employee number must be unique from person to person and, as such, is under some constraint.

DB2 provides different kinds of constraints which are described below. In this section you will create three of these constraints:

- Unique constraint
- Primary key constraint
- Check constraint

NOT NULL constraints

NOT NULL constraints prevent null values from being entered into a column. You can implement a NOT NULL constraint by selecting **Nullable** on the Add Column window. You access the Add Column window from the Create Table window.

unique constraints

Unique constraints ensure that the values in a set of columns are unique and not null for all rows in the table. For example, a typical unique constraint in a DEPARTMENT table might be that the department number is unique and not null. The steps for creating a unique constraint are described in “Adding a Unique Constraint” on page 43.

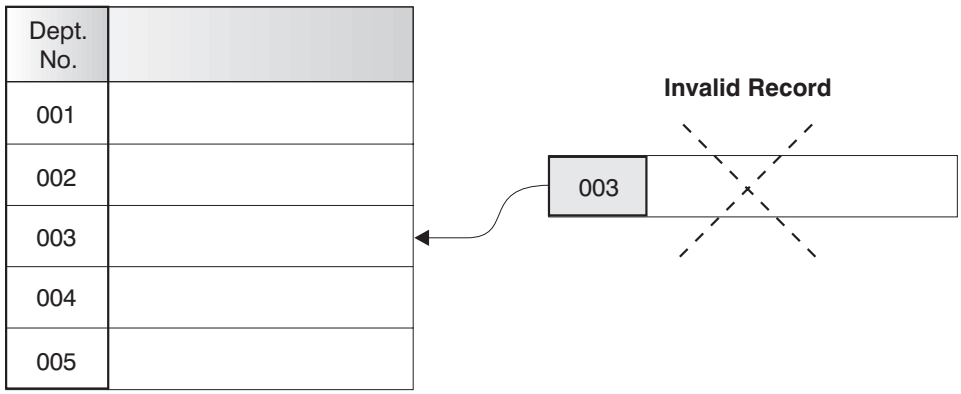

Figure 18. Unique Constraints Prevent Duplicate Data

The database manager enforces the constraint during insert and update operations, ensuring data integrity.

primary key constraint

Each table can have one primary key. A primary key is a column or combination of columns that has the same properties as a unique constraint. You can use a primary key and foreign key constraints to define relationships between tables.

Because the primary key is used to identify a row in a table, it should be unique and have very few additions or deletions. A table cannot have more

than one primary key, but it can have multiple unique keys. Primary keys are optional, and can be defined when a table is created or altered. They are also beneficial in that they order the data when data is exported or reorganized.

Instructions for creating a primary key are described in “Defining a Primary Key on an Existing Table” on page 44.

In the following tables, DEPTNO and EMPNO are the primary keys of the DEPARTMENT and EMPLOYEE tables.

Table 1. DEPARTMENT Table

DEPTNO (Primary Key)	DEPTNAME	MGRNO
A00	Spiffy Computer Service Division	000010
B01	Planning	000020
C01	Information Center	000030
D11	Manufacturing Systems	000060

Table 2. EMPLOYEE Table

EMPNO (Primary Key)	FIRSTNAME	LASTNAME	WORKDEPT (Foreign Key)	PHONENO
000010	Christine	Haas	A00	3978
000030	Sally	Kwan	C01	4738
000060	Irving	Stern	D11	6423
000120	Sean	O'Connell	A00	2167
000140	Heather	Nicholls	C01	1793
000170	Masatoshi	Yoshimura	D11	2890

foreign key constraints

Foreign key constraints (also known as referential integrity constraints) enable you to define required relationships between and within tables.

For example, a typical foreign key constraint might state that every employee in the EMPLOYEE table must be a member of an existing department as defined in the DEPARTMENT table.

To establish this relationship, you would define the department number in the EMPLOYEE table as the foreign key, and the department number in the DEPARTMENT table as the primary key.

You can create a foreign key from the **Foreign Keys** page of the Create Table notebook.

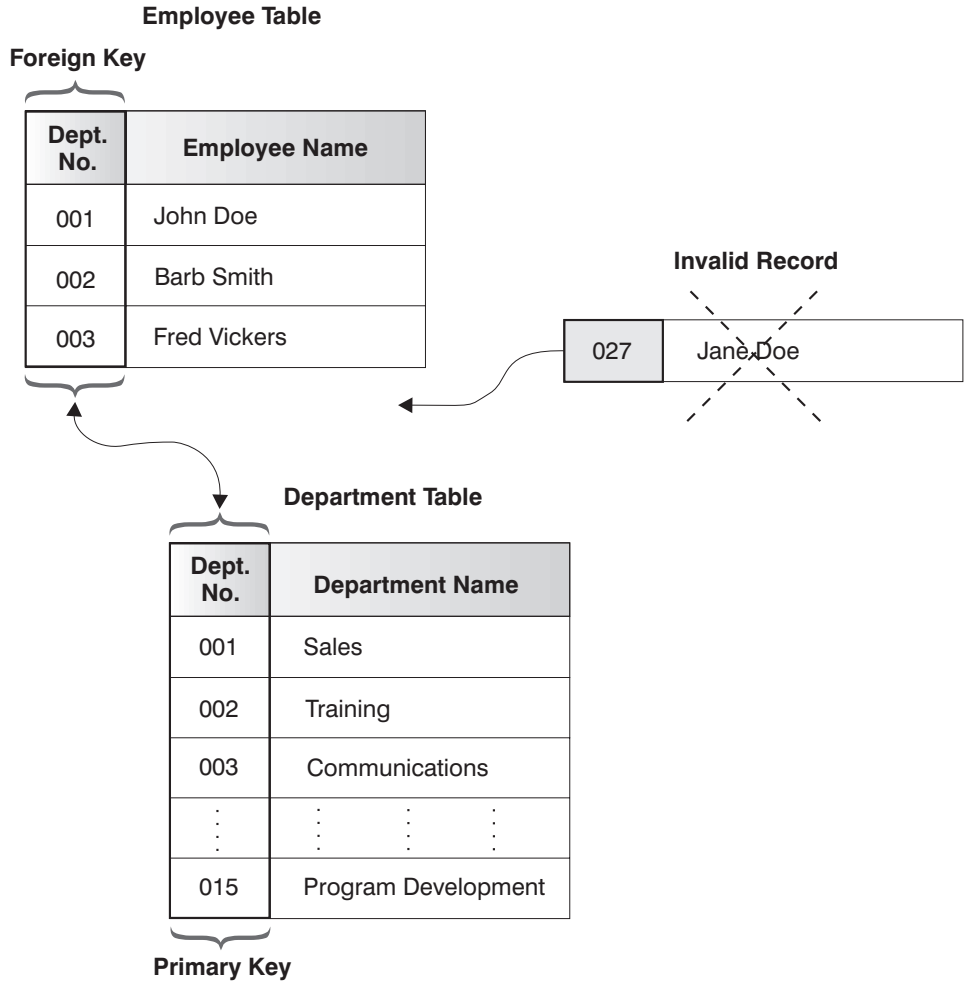

Figure 19. Foreign and Primary Key Constraints Define Relationships and Protect Data

check constraints

A check constraint is a rule in the database that specifies the values allowed in one or more columns of every row of a table.

For example, in an EMPLOYEE table, you can define the Type of Job column to be 'Sales', 'Manager', or 'Clerk'. With this constraint, any record with a different value in the Type of Job column is not valid and would be rejected, enforcing rules about the type of data allowed in the table.

The steps for creating a check constraint are described in “Adding a Check Constraint” on page 44.

When you create a unique key, a primary key, or a foreign key an index is created. See “Index Considerations” on page 74 for information on how indexes affect performance. See the *Administration Guide* for more information about constraints.

You can also use *triggers* in your database. Triggers are more complex and potentially more powerful than constraints. They define a set of actions that are executed in conjunction with, or triggered by, an INSERT, UPDATE, or DELETE clause on a specified base table. You can use triggers to support general forms of integrity or business rules. For example, a trigger can check a customer's credit limit before an order is accepted, or be used in a banking application to raise an alert if a withdrawal from an account did not fit a customer's standard withdrawal patterns. See the *Embedded SQL Programming Guide* for more information on triggers.

Adding a Unique Constraint

To create a unique constraint you use the Create Index window and create an index.

1. Open the Create Index window:
   a. From the Control Center expand the object tree until you find the **Indexes** folder.
   b. Click mouse button 2 on the **Indexes** folder and select **Create** from the pop-up menu. The Create Index window opens.
2. Use the **Index schema** list box to specify the schema for the index you are creating.
3. In the **Index name** field, type the name for the index you are creating.
4. Use the **Table schema** box to specify the schema for the table on which the index is to be created.
5. In the **Table name** field, type the lower-order (unqualified) identifier for the table on which the index is to be created.
6. In the **Available columns** list box, select the column or columns that you want to define as part of the index key.
7. Click on the > push button to move the selected column or columns to the **Selected columns** list box. The order in which the columns appear in the **Selected columns** list box is the order in which the index is created.
8. Optional: Specify the order to place the index entries in by clicking on the **Ascending** and **Descending** radio buttons.
9. Optional: Select the **Unique** check box to prevent the table from containing two or more rows with the same value of the index key.

   Select the **Unique** check box to indicate that a table will not contain two or more rows with the same value of the index key. The constraint is enforced when rows of the table are updated or new rows are inserted. The constraint is also checked during the execution of the CREATE INDEX statement. If the table already contains rows with duplicate key values, the index is not created.

When the **Unique** check box is selected, null values are treated as any other values. For example, if the key is a single column that can contain null values, that column can contain no more than one null value.

10. Optional: In the **Comment** field, document the index that you are creating.
11. Click on **Ok** to create the index.

Defining a Primary Key on an Existing Table

It is recommended that you define a primary key at the time you create a table as described in “Step 1. Creating a Table” on page 33. It is also possible to add a primary key to an existing table.

You can choose multiple columns as the primary key. For example, you might want to create a primary key on the “first name” and “last name” columns of an employee's name to ensure uniqueness. A unique index is automatically created for the columns making up the primary key.

To add a primary key to an existing table, use the **Alter** action from the pop-up menu for that table.

Adding a Check Constraint

You can add check constraints to an existing table and change or remove existing ones. Or, you can create them when you first create a table. To add new check constraints to an existing table:

1. Open the **Alter** table notebook.
   a. From the Control Center, expand the object tree until you find the **Tables** folder.
   b. Click on the **Tables** folder. Any existing tables are displayed in the pane on the right side of the window (the contents pane).
   c. Click mouse button 2 on the table you want, and select **Alter** from the pop-up menu. The **Alter Table** notebook opens.
2. Click on the **Check Constraints** tab.
3. Click on the **Add** push button. The **Add Check Constraint** window opens.
4. In the **Check condition** box, specify the check condition for the constraint you are defining.
5. Optional: In the **Constraint name** field, type a name for the check constraint.
6. Optional: In the **Comment** field, type a comment to document the new check constraint.
7. Click on **Add** to add the new check constraint.

8. If you want to add another check constraint, fill in the Add Check Constraint window again, and click on the **Add** push button. If not, click on the **Close** push button.
9. If you do not want to make any additional changes to the table, click on **OK** to alter your table and close the **Alter Table** notebook. Otherwise, you can continue on to one or more of the optional tasks described in the online help.

Creating a Schema

A *schema* is an identifier, such as a user ID, that is used to help group tables and other database objects. A schema can be owned by an individual, and the owner can control access to the data and the objects within it.

A schema is also an object in the database. It may be created automatically when the first object in a schema is created. The object can be any that can be qualified by a schema name such as a table, index, view, package, distinct type, function, or trigger. You must have IMPLICIT_SCHEMA authority to have the schema automatically created. Or, you can create a schema explicitly.

A schema name is used as the first part of a two-part object name. When an object is created you can assign it to a specific schema. If you do not specify a schema, it is assigned to the default schema, whose name is usually the user ID of the person who created the object. The second part of the name is the name of the object. For example, a user named Smith might have a table named SMITH.PAYROLL.

To access the Create Schema window so that you can create a schema explicitly:

1. From the Control Center, expand the object tree until you see the **Schemas** folder under the database in which you want the schema to reside.
2. Click mouse button 2 on the **Schemas** folder, and select **Create** from the pop-up menu. The Create Schema window opens.
3. Click on the **Help** push button if you need help while creating a schema.

Step 4. Collecting Statistics

Statistics describe the physical and logical characteristics of a table and its indexes. You must collect table and index statistics periodically for each table. These statistics are used by DB2 to determine a good way to access the data. If the data has changed significantly, to the extent that the information last collected no longer reflects the actual table data, then performance may begin to deteriorate when users are accessing data.

You should re-bind application programs that use static SQL after collecting statistics, because the SQL optimizer may choose a different access plan given the new statistics. In particular, you should re-bind those programs that reference tables for which new statistics are available. See the *Quick Beginnings* for your platform for instructions on binding application programs.

How Do I Collect Statistics?

To collect statistics for tables and indexes together:

1. Open the Run Statistics window:
   a. From the Control Center, expand the object tree until you find the **Tables** folder.
   b. Click on the **Tables** folder. Any existing tables are displayed on the right side of the window (the contents pane).
   c. Click mouse button 2 on the table you want, and select **Run Statistics** from the pop-up menu. The Run Statistics window opens.
2. Specify the level of statistics you want to gather for the table, by selecting a radio button under **Statistics for the table**.
3. Specify the level of statistics you want to gather for the table's indexes, by selecting a radio button under **Statistics for the indexes**.
4. Use the **Share level** radio buttons to specify the type of access you want other users to have to the table while the statistics are being collected:
   - Change = Read and write access
   - Reference = Read-only access
5. Click on **OK** to begin collecting statistics.

Additional options exist. See the online help accessible from the Run Statistics window.

Step 5. Backing Up a Database for the First Time

Now that you have created a database and tables, and moved data into the tables, you should do an initial backup of the database.

1. From the Control Center, click mouse button 2 on the database you want to back up. From the pop-up menu, select **Backup -> Database using SmartGuide**. The Backup Database SmartGuide opens.

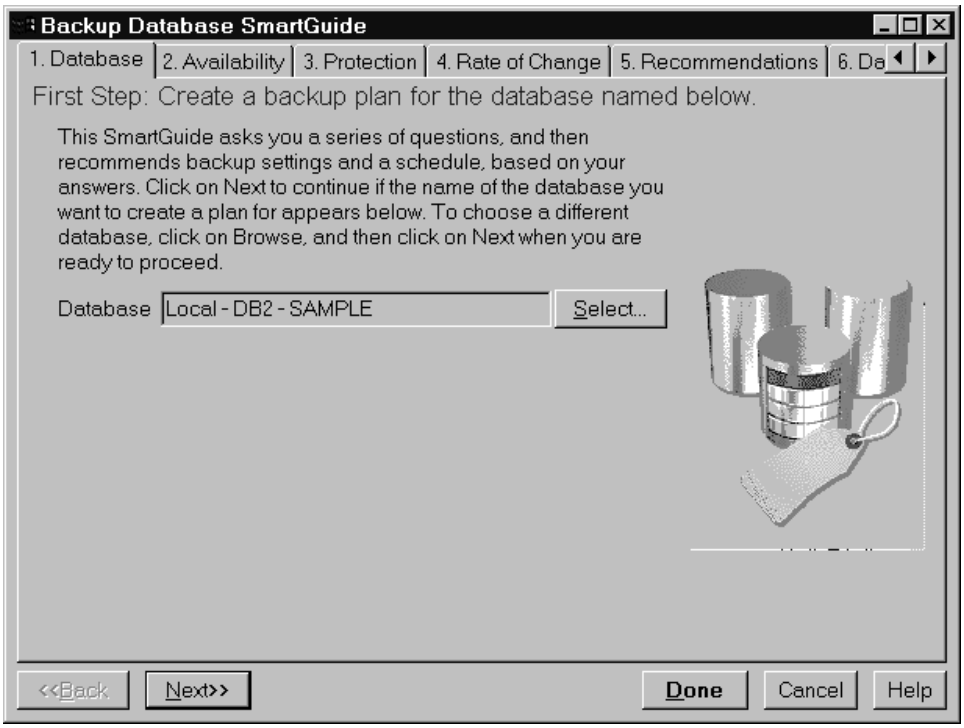

Figure 20. Backup Database SmartGuide

The SmartGuide asks you a series of questions and then recommends settings and schedules the backups. It focuses on basic options. If you want to investigate more elaborate procedures, click with mouse button 2 on a database in the Control Center and select **Backup**.

Select the **Next** push button to proceed.

2. When must your system be available?

   Select the option that best represents when your database needs to be back up and running. This influences how often your database will be backed up and what type of backup will be used.

3. What level of recovery protection do you need DB2 to provide?

   Choose *Complete recovery (Archive Logging)* so that all logs up to the last complete transaction can be replayed. Archive logging sets the LOGRETAIN parameter to *Yes*. You need to complete a full off-line backup after setting on log archiving. An *off-line backup* is a backup of a database that is made when the database is not being accessed by applications. An *on-line backup* is a backup of a database that is made while the database is being accessed by other applications.

   We recommend that you do not make any changes to your database until you have set up archive logging. Setting the *LOGRETAIN* parameter to *Yes* allows for

active and archived logs to be kept and results in the ability for the database to have roll-forward recovery.

See Chapter 5, “More About Protecting the Data” on page 61 for more detail on backup and restore. DB2 logs are described in “About the Logs that DB2 Keeps” on page 63.

4. What percentage of your data is new or changes each day?

   Use the slider to represent how much of the data in your database is new or changes each day. This influences how often the database should be backed up. You can override the suggested value.

5. Confirm the type of recovery plan you want to use. The options will summarize the database recovery plan. Each recommendation comes from the choices you made on the previous panels. You can override the recommendations or change your earlier choices.
6. Select the **Change** push button to open the Scheduler to customize the backup.
7. Specify where to store your database backup files. The SmartGuide supports disk on OS/2 operating systems and UNIX-based systems, and disk and tape on Windows NT and Windows 95 operating systems. If you use the Backup window instead of the SmartGuide, you can also back up data using ADSTAR Distributed Storage Manager (ADSM), a user exit, or a vendor-supplied backup utility (DLL) on Intel platforms. See the *Administration Guide* for more information. For added safety, you may want to store copies of the files at a different location (off-site storage).
8. Click on **Done**.

After completing the SmartGuide, and based on your responses to the questions asked, your database could be backed up:

- Using archive logging
- Off-line
- Automatically for the regular date and time you chose
- To the directory you specify

Viewing the Recovery History File

The recovery history file helps you locate information once you have performed a backup. For example, it helps you determine the location of a backup, or in which backup a DB2 object can be found. If a backup is moved to different media, say from disk to tape, this file can be updated to keep track of the new location of the backup.

To view the contents of the recovery history file:

1. Click on the **Journal** icon from the toolbar in the Control Center. The Journal window opens.
2. Select the **Recovery** page.

Every DB2 backup made contains a copy of this file, and it can be restored from any backup. If you choose to restore it, use caution to avoid overwriting the database's existing history file. Although the recovery history file is an ASCII file, manually editing the file should be done only at your own risk and is not recommended.

Chapter 4. Running Applications and Controlling Access To the Data

This chapter presents the most direct way to get a database ready for others to use. Alternate methods are described in the *Administration Guide*.

This chapter presents the following:

- Enabling applications to run
- Controlling access to the data

You may perform these tasks in a different order; however, the order shown here is the recommended one. Further, it is assumed that you came from the previous chapter and steps to this point in the book.

Step 1. Running Applications

As a database administrator, you might need to set up an application to run against the database. There are certain steps you must perform, depending on the type of application to be run. This section covers the following topics:

- Running DB2 embedded SQL applications
- Running Call Level Interface (CLI) / Open Database Connectivity (ODBC) applications. Non-ODBC applications, such as VX-REXX, come with their own set-up instructions
- Running Java programs

In addition, you can use Lotus Approach with DB2. This is described in the *Using Lotus Approach with DB2* booklet (provided in the product box with DB2).

Running DB2 Embedded SQL Programs

To run a database client application that was developed using the DB2 Software Developer's Kit and ensure it can successfully access a DB2 database, follow the steps below:

1. Be sure that DB2 is started on the database server to which the application program is connecting. If it is not:

a. From the Control Center, click mouse button 2 on the icon representing the DB2 instance you want to start, and select **Start** from the pop-up menu. If the instance is local, it is started automatically. If the instance is remote, the Attach window opens prompting you for a user ID and password.

2. Bind the DB2 utilities and the embedded SQL applications to the database. If you will also run CLI/ODBC applications, you can bind the CLI/ODBC driver at this point as well. See the *Quick Beginnings* book for your server platform.
3. Run the application programs.

Running CLI/ODBC Programs

The DB2 Call Level Interface (CLI) run-time environment and the ODBC driver are included with the DB2 Client Application Enabler. This is contained on the DB2 Client Pack CD-ROM or can be downloaded from the Web page at http://www.software.ibm.com/data/db2.

This support enables applications developed using ODBC and DB2 CLI APIs to work with any DB2 server. DB2 CLI application development support is provided by the DB2 Software Developer's Kit (DB2 SDK) which is part of the separately orderable DB2 Application Development Kit product.

Before DB2 CLI or ODBC applications can access DB2, the DB2 CLI packages must be bound on the server. Although this will occur automatically on the first connection if the user has the required authority to bind the packages, it is recommended that the administrator do this first with each version of the client on each platform that will access the server. See the *Quick Beginnings* manual for specific details.

The following general steps are required on the client system to give DB2 CLI and ODBC applications access to DB2 databases. These instructions assume that you have successfully connected to DB2 using a valid user ID and password. Depending on the platform many of these steps are automatic.

1. Use the Control Center to add the remote system (if you have separate client and server machines) so that its instances and databases can be made known to the Control Center, then add the instances and databases for that system. (Your local system is represented by the icon labelled **Local**.) If you do not have access to this program you can use the **catalog** command in the command line processor.
2. On all platforms other than OS/2 and Windows 3.1, the DB2 CLI/ODBC driver is automatically installed when the DB2 Client Application Enabler is installed, and therefore nothing needs to be done. On OS/2 and Windows 3.1 you must use the **Install ODBC Driver** icon to install both the DB2 CLI/ODBC driver and the ODBC driver manager.
3. To access the DB2 database from ODBC:
   a. The Microsoft, Visigenic, or other ODBC Driver Manager must already be installed (this is done by default during the installation of DB2).

b. The DB2 databases must be registered as ODBC data sources. The ODBC driver manager does not read the DB2 catalog information; instead it references its own list of data sources.

c. If a DB2 table does not have a unique index then many ODBC applications will open it as read-only. Create a unique index for each DB2 table that is to be updated by an ODBC application. Refer to the **CREATE INDEX** statement in the *SQL Reference*. Using the Control Center you would alter the settings of the table, then click on the **Primary Key** tab and move one or more columns from the available columns list over to the primary key columns list. Any column you select as part of the primary key must be defined as NOT NULL.

4. Various CLI/ODBC Configuration Keywords can be set to modify the behavior of DB2 CLI/ODBC and the applications using it.
5. If you followed the above steps to install ODBC support, and added DB2 databases as ODBC data sources, your ODBC applications will now be able to access them.

For specific instructions see the *Quick Beginnings* manual for your platform.

Running Java Programs

You can use DB2 Java Database Connectivity (JDBC) support to run the following types of Java programs:

- Java applications, which rely on the DB2 Client Application Enabler to connect to DB2.
- Java applets, that do not require any other DB2 component code on the client.

Java can also be used on the server to write user-defined functions, stored procedures, and table functions.

For further information on developing Java applications that access DB2 databases refer to the Web page located at http://www.software.ibm.com/data/db2/java/ and the *Road Map to DB2 Programming*.

Configuring the Environment

The following environment variables must be set to run Java programs:

OS/2, Windows 95 and Windows NT

- CLASSPATH includes "." and the file sqllib\java\db2java.zip
- PATH includes the directory sqllib\bin
- LIBPATH includes the directory sqllib\dll (OS/2 only)

UNIX (AIX, Solaris and HP-UX)

In the following settings *instance_name* is the home directory of the DB2 instance owner.

- CLASSPATH includes "." and the file *instance_name*/sqllib/java/db2java.zip
- PATH includes the directory *instance_name*/sqllib/bin
- LD_LIBRARY_PATH includes the directory *instance_name*/sqllib/lib
- SHLIB_PATH include the the directory *instance_name*/sqllib/lib (HP-UX only)

Java Applications

Start your application from the desktop or command line, like any other application. The DB2 JDBC driver handles the JDBC API calls from your application and uses the DB2 Client Application Enabler to communicate the requests to the server and receive the results.

Java Applets

Because Java applets are delivered over the Web, you treat them a bit differently than Java applications. You must install DB2 (server or client) on the same machine as your Web server, then install the Java Development Kit (JDK) Version 1.1 from Sun Microsystems on the server as well (refer to http://www.software.ibm.com/data/db2/java for details).

To run your applet, you need only a Java-enabled Web browser on the client machine. When you load your HTML page, the applet tag downloads the Java applet to your machine, which then downloads the Java class files, including the COM.ibm.db2.java.sql and COM.ibm.db2.jdbc.net classes and DB2's JDBC driver. When your applet calls the JDBC API to connect to DB2, the JDBC driver establishes separate communications with the DB2 database through the JDBC applet server residing on the DB2 server.

To run your applets, do the following:

1. Start the DB2 JDBC applet server on your Web server by entering:

   `db2jstrt portno`

   where *portno* is the number of the unused TCP/IP port that you specified in the DB2Applt.java file.
2. On your client system, start your Web browser and load the HTML file that imbeds your applet.

Step 2. Controlling Access to the Data

As a database administrator, you might need to control the type of access people have to data, or restrict their view of the data.

This section briefly describes two methods for controlling access to the data:

- Managing database authorities and privileges for database objects. An overview of database authorities, privileges on database objects, and system authorities is provided.
- Creating a view

About Authorities and Privileges

Database authorities involve actions on a database as a whole. When a database is created, some authorities are automatically granted to anyone who accesses the database. For example, CONNECT, CREATETAB, BINDADD and IMPLICIT_SCHEMA authorities are granted to all users. *Database privileges* involve actions on specific objects within the database. When a database is created, some privileges are automatically granted to anyone who accesses the database. For example, SELECT privilege is granted on catalog views and EXECUTE and BIND privilege on each successfully bound utility is granted to all users.

In “Welcome” on page vii, we recommend that you use the user ID that was created following the instructions in the *Quick Beginnings* for your platform. That user ID has SYSADM authority. The *Administration Guide* describes all the *system authority* levels. In short they are:

- Administrative authority (SYSADM or DBADM): They give full privileges for a set of objects.
- System authority (SYSCTRL or SYSMAINT): They give full privileges for managing the system, but do not allow access to the data.

Together, privileges and authorities act to control access to an instance and its database objects. Users can access only those objects for which they have the appropriate authorization, that is, the required privilege or authority.

See the section on authorities, privileges, and authorizations, and the section on controlling database access in the *Administration Guide* for a more detailed discussion of authorities and privileges.

Granting and Revoking Authorities and Privileges

You can use the DB2 administration tools to grant and revoke privileges for users and groups for databases, table spaces, tables, views, and schemas. You do this as follows:

1. From the Control Center, click mouse button 2 on the database, table space, table, view, or schema for which you want to grant or revoke privileges, and select **Authorities** or **Privileges** from the pop-up menu. The Authorities window or Privileges window opens.
2. Select the **User** page to work with user authorities or privileges, or the **Group** page to work with group authorities or privileges.
3. Select one or more users from the **User** page, for example. (To add a user to the list, click on the **Add User** push button.)
4. Along the bottom of the window, select **Yes**, **No**, or **Grant** for each individual authority or privilege. (**Grant** is displayed only for objects for which it is a valid option.)
5. When you have finished, click on the **Apply** push button.

See the *Administration Guide* for the details of managing database authorities, privileges for database objects, and system authorities.

Creating a View to Restrict Access

A view is defined in “Views” on page 11. You can define a view so that it includes some or all of the rows in a table. You can create a view, for example, for each individual department manager that allows each manager to look at the salary information for their employees. This is done by specifying the department number for each manager when you define each view.

To create a view:

1. Open the Create View window:
   a. From the Control Center, expand the object tree until you find the **Views** folder.
   b. Click mouse button 2 on the **Views** folder, and select **Create** from the pop-up menu. The Create View window opens.
2. Optional: Use the **View schema** list box to specify the schema for the view that you are creating.
3. In the **View name** field, type the lower-order (unqualified) identifier for the view that you are creating.

4. In the **SQL statement** box, edit the skeleton CREATE VIEW SQL statement to create your own. For example, the following statement creates a view of the non-managers in the STAFF table where salary and commission do not show through from the base table to the view:

   ```
   AS SELECT ID, NAME, DEPT, JOB, YEARS
        FROM STAFF
        WHERE JOB <> 'MGR'
   ```

   Note that the <> operator in the WHERE clause means not equal.

   See the *SQL Reference* for information on the CREATE VIEW statement.

5. Optional: Specify a **Check option** by clicking on either the **Cascaded** or **Local** radio button.

   A **Check option** specifies the constraint that conforms to the definition of the view. A row that does not conform to the definition of the view is a row that does not satisfy the search conditions of the view. The online help describes each of the options.

6. Optional: Type a comment in the **Comment** field.

7. Click on **OK** to begin processing the SQL statement. The view is created and listed in the list of views in the contents pane.

8. To see the rows of data for the view, click with mouse button 2 on the view and select **Sample contents** from the pop-up menu. The view is displayed.

Part 3. Database Operation

Chapter 5. More About Protecting the Data

This chapter provides more detailed information about the type of backup described in “Step 5. Backing Up a Database for the First Time” on page 46 and gives some additional background knowledge in the important area of protecting your data. This chapter presents the following topics:

- Events that can lead to loss of data
- Backing up a database:
  - About the logs DB2 keeps
  - Configuration parameters for database logging
  - Choosing a backup strategy
  - Facilities available for recovering data:
    - Recovering in the event of a power failure
    - Performing a full database restore
    - Bringing the database back to a certain point in time
    - Recovering in the event of a disk failure

Events that Can Lead to Loss of Data

There are several circumstances in which you will need to ensure that adequate protection for your data is in place. In some cases, DB2 takes care of this for you; other cases require careful consideration and planning on your part. The following list outlines situations that require data recovery:

1. System crash or power failure (*DB2 handles this automatically*)

   The operating system crashes, due to either a software or hardware problem, or the system as a whole goes down unexpectedly due to some external power problem before all the changes that are part of a unit of work have been completed and committed.

2. Transaction failures (*DB2 handles this automatically*)

   An application is running but the database manager ends abnormally before the application can either commit or roll back its changes. Roll back refers to restoring changes to the state they were in when they were last committed in the database.

3. Media failure

   A hardware failure (for example, head crash, bad sector) occurs on a disk running your DB2 database.

4. Application errors

Errant program logic or invalid data entry leads to invalid data in the database that DB2 cannot prevent.

5. Disasters

   A wide range of non-trivial damage to more than just a single component of your machine. The system running DB2 becomes partially or completely destroyed, or the facility at which your systems are located suffers damage that impacts your operation.

As a database administrator, you need to evaluate the impact that each of the above would have on your business and implement a backup plan accordingly. The Backup Database SmartGuide (described in “Step 5. Backing Up a Database for the First Time” on page 46) helps you with this.

DB2 handles system crashes and transaction failures automatically. It uses the active logs to recover from these two types of failures, so you should not need to make use of your backup for these two types of failures. Further details on logging facilities are provided in “About the Logs that DB2 Keeps” on page 63.

Media failure is a typical cause of lost data requiring recovery. The options for protecting against this type of failure include database backups.

Point-in-time recovery may be able to assist you in minimizing damage caused by application logic errors.

Disaster recovery is a broad topic. At a minimum, data must be backed up and stored at a location not affected by the damage to the primary site. Beyond that, required procedures that are unique to each case are not discussed here.

Backing Up a Database

The steps required to perform a database backup using the Backup Database SmartGuide are described in “Step 5. Backing Up a Database for the First Time” on page 46. The SmartGuide helps you make basic protection decisions. Here we describe some aspects of database backup and recovery in more detail. Another type of DB2 backup, a table space backup, is covered in the *Administration Guide*.

The concept of a database backup is the same as any other data backup: taking a copy of the data and storing it on a different medium in case of failure or damage to the original. The simplest case of a backup requires shutting down the database to ensure that no further transactions occur, and then simply backing it up.

An aspect that is unique to database backups is the need to consider the database logs. If a database needs to be restored to a point beyond the last full, off-line backup, then logs are required to roll the data forward to the point of failure. Rolling data

forward refers to the log journal information which is applied against a restored database. Log journals contain the changes made to the database since the last backup. After these journals have been applied, the database will be in the same state as it was prior to the failure. In the case of a machine or site disaster in which the logs are destroyed, the database can only be restored to the point of the last full backup and only if the backup media were stored at a location unaffected by the disaster.

About the Logs that DB2 Keeps

All databases have logs associated with them. These logs keep records of database changes. Some logs, called *active logs*, prevent a failure (system power, application error) from leaving a database in an inconsistent state. They restore the state of a database to the point before the change. See the *Administration Guide* for additional information about transaction support.

There are two types of DB2 logging: *circular* and *archive*, with each providing a different level of recovery capability. Your choice between these two depends on your business needs with respect to the recovery of your database in the event of a failure.

Circular Logging

Circular logging is the default when a new database is created. With this type of logging, only full, off-line backups of the database are valid. As the name suggests, circular logging uses a "ring" of online logs in order to provide recovery from transaction failures and system crashes. The logs are used and retained only to the point of ensuring the integrity of current transactions. Circular logging does not allow you to roll forward a database through prior transactions from the last full backup. Recovery from media failures and disasters is done by restoring from a full, off-line backup. All changes since the last backup are lost. The database must be off-line (inaccessible to users) when a full backup is taken. Since this type of restore recovers your data to the specific point in time of the full backup, it is sometimes called *version recovery*.

Figure 21 on page 64 illustrates how the active log uses a ring of log files when circular logging is active.

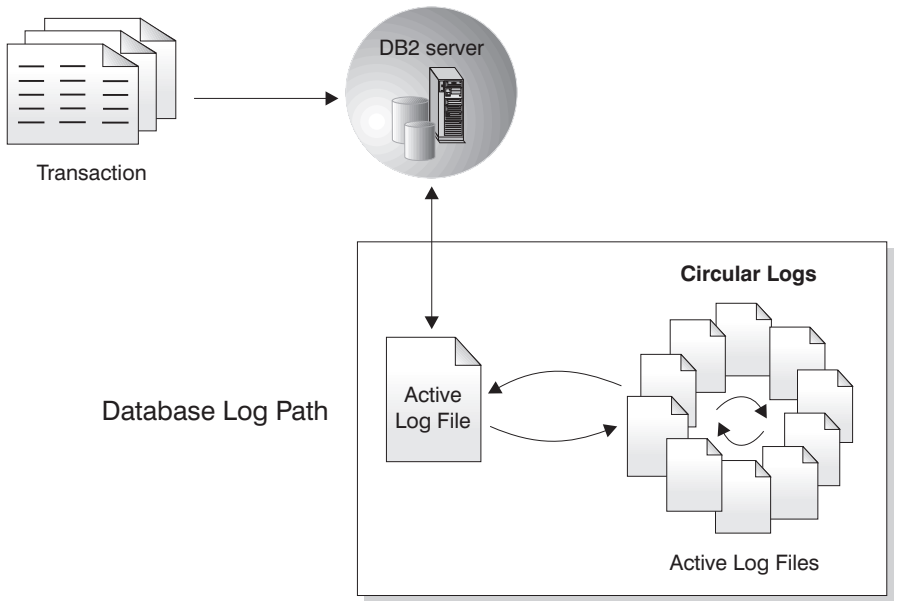

Figure 21. Circular Logging

Active and Archive Logging

Active Logs: Active logs contain transactions which have not yet been committed or rolled back, or whose changes have not yet been written to disk. Active logs are located in the database log path directory.

Archived Logs: Archive logging enables forward recovery using active and archived logs to any point in time before the failure, rather than only to the point in time of a full backup. With archive logging, active logs are retained and become online, archived logs. In addition, archived logs can be moved off-line and still used for roll-forward recovery. Archive logging is activated when either the LOGRETAIN or USEREXIT (or both) parameter is enabled. Once archive logging is activated, a full backup of the database is required. If LOGRETAIN or USEREXIT are turned off, logging reverts to circular, and the online logs are automatically deleted. If it is necessary to back out the changes from an errant application that damaged data, a database can be rolled forward using the logs to any point in time between the full, off-line backup and the last completed transaction.

Figure 22 on page 65 shows a database using active log files and online archived log files.

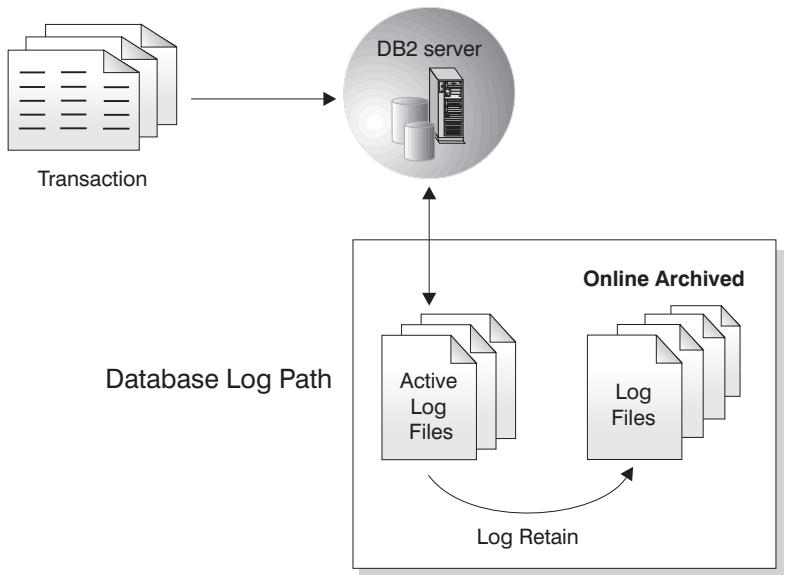

Figure 22. Archive Logging

With archive logging, it becomes necessary to pay more attention to the handling of the logs and to ensure their safety. The ability to perform roll-forward recovery of your database is dependent on the integrity of the logs. Consideration should be given to backing up log data and to keeping it on disk arrays or *mirrored volumes*. A suggestion is provided in “Choosing a Backup Strategy” on page 67. For either type of logging, performance factors such as the location and size of the logs needs to be evaluated for the impact on overall system performance. Finally, in planning for disaster recovery, be sure to remember that log data must be stored off-site, or at least safely away from the disaster, in order to recover your database beyond the point of the last full, off-line backup.

Online Archived Logs: When all changes in the active log are no longer needed for normal processing, the log is closed and becomes an archived log. An archived log is said to be "online" when it is stored in the database log path directory.

Off-line Archived Logs: You also have the ability to store archived logs in a location other than the database log path directory, by using a user exit program. (See the section on user exits for database recovery in the *Administration Guide*.) An archived log is said to be "off-line" when it is not stored in the database log path directory.

The interaction among the active log file and offline archived log files is shown in Figure 23 on page 66.

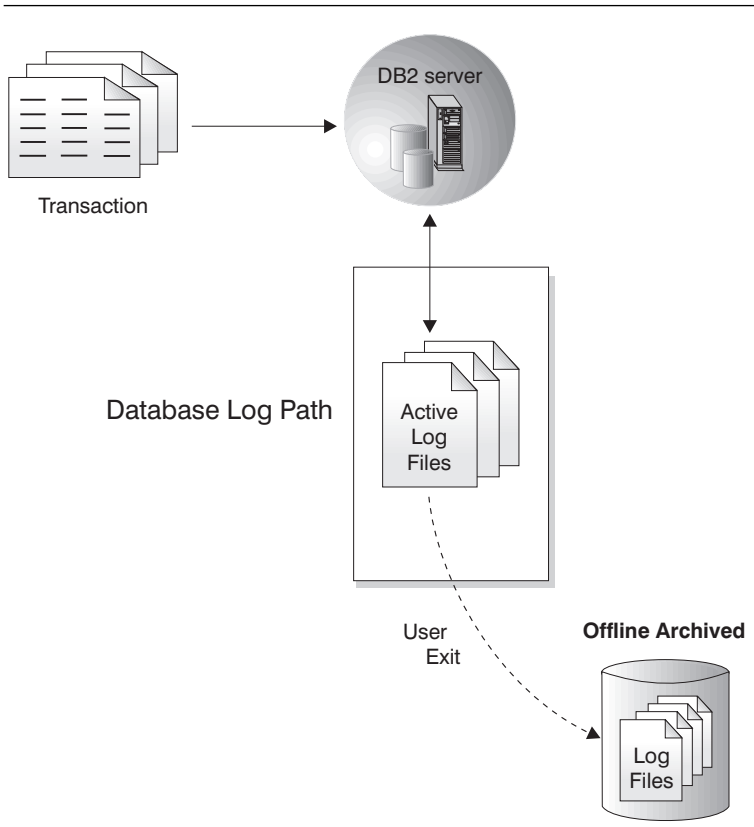

Figure 23. Offline Archived Logs

Configuration Parameters for Database Logging

The database configuration file contains parameters related to roll-forward recovery. The default parameters do not support this recovery, so if you plan to use it, you need to change some of these defaults. In “Step 5. Backing Up a Database for the First Time” on page 46 we recommend that you choose archive logging so that the LOGRETAIN parameter is set to Yes, and that you do not make any changes to your database until this point. This means archived logs will be kept in the database log path directory and that the roll-forward recovery method will be used.

To modify the parameters related to roll-forward recovery:

1. From the Control Center, expand the object tree until you find the database you want to configure.
2. Click mouse button 2 on the database and select **Configure** from the pop-up menu. The Configure Database notebook opens.

3. Select the **Recovery** tab.
4. Modify the parameters you wish. For a description of all the configuration parameters, look for “Database configuration parameters” in the table of contents of the online help.

Choosing a Backup Strategy

Defining a robust backup strategy for your database requires careful consideration of your business requirements, including factors such as the requirements for database availability and the maximum time the database can be down for backups. The Backup Database SmartGuide, described in “Step 5. Backing Up a Database for the First Time” on page 46, helps you with the decision-making. This section describes the considerations and issues in more detail, in order to provide you with a better understanding of this important area.

To begin, you must decide on the types of failure that you are protecting against. The purpose of running a backup is to be able to use it to perform a recovery. Therefore, you must test your processes and procedures to ensure that your backups will be useful in the event of a failure. Backups that cannot be used to restore data are pointless.

The following are general guidelines for planning a recovery strategy:

- The type of data contained in your database is relevant. Databases that contain read-only data do not need to be protected through archive logging if full, off-line backups are run following each new data load activity. The use of circular logging would be sufficient in this case.
- With continuously updated data that is deemed important to your business, you must use archive logging.
- If your database must be continuously available, you must take online backups. This requires the use of archive logs.
- If in the event of a failure your database must be recovered in a short time, you will need to run more frequent backups. In this case, you need to establish how long it would take to recover from a failure (the sum of the time to restore the database from a backup plus the time needed to roll the log forward).

Beyond transaction failures and system crashes, both of which DB2 will recover from, you should consider application errors. This refers to the general case of data being damaged in some manner. Clearly there is no way to prevent an authorized user from altering data inappropriately. The best strategy for dealing with this type of problem is to ensure that you have archive logs to roll forward the database to the point just prior to the corruption of the data. Be sure to factor into your schedule the maximum time the database can be down for backups, and whether these are online or off-line.

Probably the most common type of failure is caused by media problems. This is not limited to disk problems, but can extend to other I/O devices, including disk controllers and tape devices. As a starting point, it is suggested that you do not back up your database to the same disk on which the production version exists: use either a separate disk or external media. The handling of your logs should be similar: consider directing these to a separate physical disk from that of the database. In addition to protecting against a disk failure affecting both, this may also result in performance improvements.

Though unlikely, it is possible that your backup media could suffer a problem just when it is needed to let you recover from a disk failure. Consider the impact of a tape going bad. If your data is absolutely critical, you should consider having duplicate tape media. Another strategy is to minimize the potential for impact caused by a bad disk. This applies to the disks that both the database and logs reside upon. Using disk arrays for your database volumes or logs (or both) is perhaps the best defense against disk media failures. See the *Administration Guide* for information on disk arrays. If you extend redundancy to disk controllers as well, it is highly unlikely that your database will ever be unavailable or that logs will be lost due to a media failure.

Facilities for Recovering Data

Once a database backup is made, there are several ways to restore the database. This section briefly describes the following:

- Recovering in the event of a power failure
- Performing a full database restore
- Bringing the database back to a certain point in time
- Recovering in the event of a disk failure

For complete details, see the *Administration Guide*.

How Do I Recover in the Event of a Power Failure?

DB2 will automatically recover from system crashes, even with the default of circular logging. All committed units of work that are not written to disk will be redone when the system comes back up and the first application connects to a database or when a database is restarted (from the database you want to restart in the object tree, select **Restart** from its pop-up menu). Crash recovery is performed during restart processing.

How Do I Perform a Full Database Restore?

The Restore Database SmartGuide will help you restore a database to the point in time of the last database backup or the last complete transaction. The recovery plan you created for this database determines what is possible.

To restore a database:

1. From the Control Center, click mouse button 2 on the database you want to restore, and select **Restore -> Database using SmartGuide**. The Restore Database SmartGuide opens.
2. Confirm the state of your database. For example, the SmartGuide will indicate if the database is in a roll-forward pending state.

   Confirm that the name of the database to restore is correct. If you want, you can specify the name of a different database.

   To do a basic restore of the specified database (that is, using the most recent backup and rolling forward to the end of the logs), click on the **Done** push button.

   If you wish to modify the restoration parameters, click on the **Next** push button to continue through the remaining steps.
3. Select the database backup image to restore from.
4. Specify how to roll forward the information recorded in the log files.

   You need to roll forward (reapply) the changes to your database since the last time you backed it up. Those changes are stored in a set of log files. If you cannot roll forward all the changes since the last backup, you can roll forward to a specific point in time. If you choose to roll forward to a point in time, you can choose to do more later, or stop and make the database available for use.
5. Click on the **Done** push button to restore the database.

How Do I Bring the Database Back to a Certain Point in Time?

Occasionally, you may need to undo something that happened to your database: for example, if an errant application program has damaged your data, you would want to put the database back to the point just before that application program was run against it. To do this, use the Restore Database SmartGuide to perform a full database restore, but in the last step, specify a date and time to which you wish to roll forward your database using the logs.

How Do I Recover in the Event of a Disk Failure?

In many cases, a disk failure does not involve a loss of data. If it does, you must perform a restore.

Unless the failing disk is part of a disk array, your database will be down and unavailable to users. First, you need to replace the drive, then follow the steps in “How Do I Perform a Full Database Restore?” on page 69 to perform a full database restore, and roll forward the database using the logs.

As an alternative to first waiting for hardware repairs, you could perform a *redirected restore*, to restore the database to some other undamaged disk. You can perform a redirected restore from the Restore Database notebook (from the database you want to restore, select **Restore ->** **Database** from the pop-up menu). See the *Administration Guide* for more information about redirected restores. As indicated earlier, you generally should not keep the logs on the same physical disk as the database, since in that case you would be unable to roll forward to your most recent transaction.

If the failed disk was part of a disk array, the database will still be running, and you can simply replace the disk in the array at a later time. A disk array consists of a collection of disk drives that appear as a single large disk drive to an application. See the *Administration Guide* for more information.

Chapter 6. The Basics of DB2 Performance

This chapter presents some basic considerations and actions regarding database performance at the server:

- Physical database design considerations
- Separating different types of data
- Basic capacity management
- Index considerations
- Reorganizing tables in a database
- Database configuration parameter settings
- Tuning the buffer pool size(s)
- Monitoring performance
- Other areas to watch

Physical Database Design

Logical database design considerations (for example, deciding what data to store in the database and defining keys) are addressed in Chapter 2, “More About Creating a Database” on page 27 and in Chapter 3, “Preparing a Database for Use” on page 33, and in more detail in *Administration Guide*.

The basic physical database design considerations are:

- Database physical storage
- Estimating space requirements for tables
- Additional space requirements
- Designing/choosing containers and table spaces

This book addresses the last item only. For details on all aspects of physical database design, see the *Administration Guide*. “Suggestions for Allocating Storage for a Dedicated Server” on page 30 provides suggestions regarding creation and placement of disks, containers, table spaces, and tables. Those suggestions reflect only one sample implementation, and may not apply to a large database configuration.

Separating Different Types of Data

One aspect of physical database design you might consider is to keep indexes and large object (LOB) data separate from the rest of the table data when you create a table. This option is described in “Step 1. Creating a Table” on page 33. These separate table spaces can possibly be on different media (containers) than the rest of the data. This flexibility can be used to increase database performance and availability. You can also allocate a table space to a specific buffer pool. See “Buffer Pool” on page 19 for a description of a buffer pool.

Indexes stored in a different table space from that used to store other table data can allow for more efficient use of disks by reducing the movement of read/write heads. You can also associate index table spaces with faster devices, which can speed up index access.

If you store long data types and LOB data in a different table space, you can allocate a special device to this type of data. For example, you may choose to store LOB data on devices that have a very large capacity but are slow, and put the rest of your data, which may be accessed more often, on a faster device. You can turn off logging of individual LOB columns for improved performance.

Basic Capacity Management

There are a few basic things you can do to manage the capacity of a database. This section describes how to:

- Check the amount of space available in a table space
- Add more space to an existing table space when it starts to get full

Checking Space Available in a Table Space

To check the amount of space available in a DMS table space:

1. From the Control Center, double-click on the **Table spaces** folder for the database you want to work with. A list of all the table spaces appears in the contents pane.
2. Scroll to the columns entitled **Allocated size**, **Size used**, and **Percentage used** to see details related to the amount of space available in a table space. Space is shown in pages where one page is 4 KB.

You can customize the order of the columns and which columns are displayed by using the **Customize Columns** push button.

To check the amount of space available in an SMS table space, use the facilities provided by your operating system to monitor space usage and to ensure that available room in the directory for the table space is not exhausted.

Adding More Space To a Table Space

The previous section describes how to check the capacity of a DMS table space. Capacity for a DMS table space is the total size of containers allocated to the table space. When a DMS table space reaches capacity (depending on the usage of the table space, 90% is a possible threshold), you should add more space to it. The database manager will automatically re-balance the tables in the DMS table space across all available containers. During re-balancing, data in the table space remains accessible.

The strategy is different for DMS and SMS table spaces. For a DMS table space that has reached its capacity, you can add another container, as follows:

1. From the Control Center, click mouse button 2 on the table space in the contents pane for which you want to add a container, and select **Alter** from the pop-up menu. The Alter Table Space window opens.
2. Click on **Add**. The Add Container window opens.
3. Select the **File** or **Raw device** radio button, and complete the fields. See the online help for assistance.
4. Click on **OK**.

In general, you cannot extend the size of an SMS table space very easily because SMS capacity depends on the space available in the file system and the maximum size of the file supported by the operating system. However, depending on your operating system, you may be able to increase the size of a file system using the operating system facilities. For an SMS table space on a UNIX-based system, you can increase the size of the table spaces by using the appropriate UNIX-based system commands. See the documentation for the UNIX-based system you are running. If the file system containing the SMS table space also contains non-DB2 files, you may be able to move these files to another file system, thus making more room available in the file system for DB2's use. You can also perform a redirected restore which involves restoring a table space into a larger number of containers than it was backed up from. See the *Administration Guide* for more information about redirected restores. You can perform a redirected restore from the Restore Database notebook (from the database you want to restore, select **Restore ->** **Database** from the pop-up menu).

Index Considerations

When you create a unique key, a primary key, or a foreign key an index is created. An index optimizes data retrieval without performing a lengthy sequential search; that is, you can avoid having the entire table scanned when data is queried. You do not decide when an index should be used to improve performance; DB2 makes this decision based on the available table and index information. However, you play an important role in the process by defining keys that create the necessary indexes. It is also important for you to collect statistics on your indexes, both when you create them and on an ongoing basis. Keys are described in “Step 3. Enforcing Business Rules for Data” on page 39. “Step 4. Collecting Statistics” on page 45 describes how to collect statistics.

The indexes you need are based on how you want to group the data and how the data is most frequently accessed. Creating a large number of indexes for a table that receives many updates can slow down processing of requests. Therefore, you should use indexes only where a clear advantage for frequent access exists. The simplest recommendation is that you define a key on the table columns you use the most.

Once you have a good understanding of the performance of your system, you may be able to improve performance by placing indexes in separate table spaces.

Note that you cannot change any clause of an index definition; you must drop the index and create it again. (Dropping an index does not cause any other objects to be dropped but may cause some packages to be invalidated.) The online help for the Control Center describes how to drop an index.

See the *Administration Guide* for more information on indexes.

Reorganizing Tables in a Database

A table can become fragmented due to many updates, causing performance to deteriorate. If you collected statistics and did not notice a visible performance improvement, reorganizing table data may help. When you reorganize table data you are rearranging the data of a table into a physical sequence according to a specified index, and removing the free space that is inherent in fragmented data. This can provide faster access to the data and thereby improve performance.

Before you reorganize tables, we recommend you collect the statistics of the data to allow the **reorgchk** command to see up-to-date statistics. This is described in “Step 4. Collecting Statistics” on page 45. The results of the **reorgchk** command will help you determine whether a reorganization of table data should be performed. The **reorgchk**

command has an option to cause statistics to be collected as part of its processing. See the *Command Reference* for information on this command.

To reorganize table data:

1. Open the Reorganize window:
   a. From the Control Center, expand the object tree until you find the **Tables** folder.
   b. Click on the **Tables** folder. Any existing tables are displayed in the contents pane.
   c. Click mouse button 2 on the table you want in the contents pane, and select **Reorganize** from the pop-up menu. The Reorganize window opens.
2. Optional: In the **Using temporary table space** field, change the name of the temporary table space where the table being reorganized can be temporarily stored.

   The value is initially <none>, specifying that the temporary copy should be stored in the table space where the table currently resides.
3. Optional: In the **Using index** field, specify an index to use to reorganize the table rows.

   If you specify <none> (the default), the table rows are reorganized without regard to order.
4. Click on **Reorganize Now** to reorganize the table immediately. You can also schedule the reorganization to occur at regular intervals by clicking on **Schedule**. You might find it useful to schedule this activity because reorganization can be time consuming and users will not be able to access the table being reorganized.
5. Collect the statistics again. This is described in “Step 4. Collecting Statistics” on page 45. This provides the most up-to-date statistics for the data so that you can get the fastest access to the data based on the new organization of the data and indexes.

   You should re-bind application programs that use static SQL after collecting statistics, because the SQL optimizer may choose a different access plan given the new statistics. In particular, you should re-bind those programs that reference tables for which new statistics are available. See the *Quick Beginnings* for your platform for instructions on binding application programs.

The *Administration Guide* provides more details about reorganizing table data and collecting statistics.

Setting the Database Configuration Parameters

An important way to improve performance is to adjust the configuration parameters. The parameters fall into two general categories:

- Database manager parameters (at the instance level) — Most of these parameters affect the amount of system resources that will be allocated to a single instance.
- Database parameters — Most of these parameters specify the amount of resources allocated to a specific database.

You may need to modify the default values of these parameters if your environment has any of the following:

- A large amount of memory (greater than 64 MB)
- Large databases
- Large numbers of connections
- High performance requirements for a specific application
- Different machine configuration/use
- Unique query/transaction loads or types

Using the Performance Configuration SmartGuide

This SmartGuide helps you tune performance and balance memory requirements for a single database per instance by suggesting which configuration parameters to modify and providing suggested values for them.

To use the SmartGuide:

1. From the Control Center, click with mouse button 2 on the database for which you want to configure performance.
2. Select **Configure Performance** from the pop-up menu. The Performance Configuration SmartGuide opens.
3. Follow the steps in the SmartGuide.

   If you click on **Apply these recommendations immediately** on the **Results** page (last page) of the SmartGuide, then when you click on **Done** after you have finished selecting the appropriate options, the recommended configuration parameters for your system will be applied to the database immediately. However, updates to parameters take effect under the following conditions:

   - For database configuration parameters, all applications must first disconnect from the database. The changes will take effect only at the first new connect to

the database. If the **activate database** command was used to activate the database, the **deactivate database** command must be issued before the changes will take effect.

- For database manager configuration parameters, you must stop and then start the instance.

The next time you open this SmartGuide, you are given the option of backing out of the options you selected the previous time the SmartGuide was open.

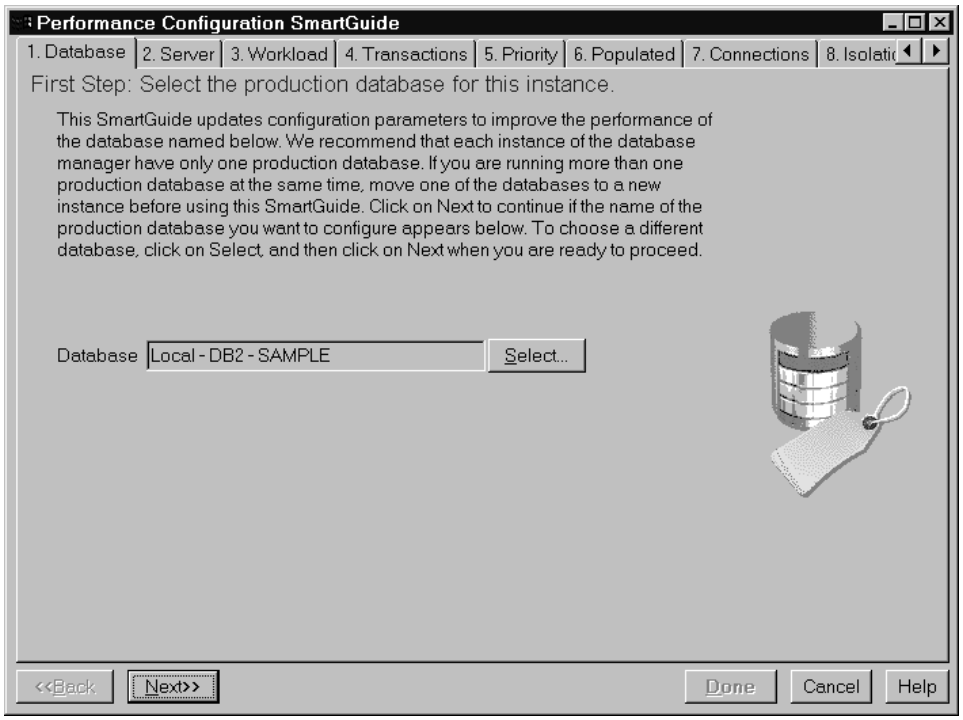

Figure 24. Performance Configuration SmartGuide

We recommend that each instance of the database manager have only one production database. It is difficult to tune a system with more than one production database per instance running concurrently.

The recommended values are designed to improve the performance of, though not necessarily optimize, your database system. These values should be thought of as a starting point on which you can make further adjustments to obtain optimized performance. In most cases they will provide better performance than the default values because they are based on information about your workload and your server.

The parameters affected are directed at Data Manipulation Statements (DML). This SmartGuide is not intended to improve the performance of utilities.

In order to use the SmartGuide effectively, you need to know the following about the database and your system:

- The volume of workload on your system.
- The portion of the server's memory (RAM) that can be used by this database (not including the operating system).
- Whether there is anything else running on your system. For example, the system may also be a server supporting CICS transactions.
- Whether it is more important for you to optimize transaction performance (more transactions per minute) or the time required to recover the database after a failure.
- What a typical database transaction is. Choose the number of SQL statements in a single unit of work (between commits) that best reflects the database. Also, estimate the number of transactions per minute running in your database. If you are not sure which value to use, accept the default provided by the SmartGuide.
- The number of users connecting to the database.
- The type of workload on the system. For example:
  - There are many queries of the data (data warehousing)
  - The data is updated (order entry)
  - Isolation level (row locking):

    The *isolation level* determines the number of locked rows and the locks' duration when a user either reads or changes data. DB2 uses *locking* to maintain data integrity during concurrent processing. Locking guarantees that a transaction maintains control over a database row until it has finished, and prevents another application from changing a row before the ongoing change is complete. See the *Administration Guide* for more information.
  - Whether there is any production data in the database. If there is, its volume is included in the recommendations.

We recommend that you rerun this SmartGuide again if the size of the database significantly increases (for example, more than 20%) or if the machine characteristics change (for example, more memory is added).

Tuning the Buffer Pool Size(s)

Buffer pools are defined in “Buffer Pool” on page 19. Each table space is associated with a buffer pool. When you create a table space, the default buffer pool (called IBMDEFAULTBP) is used. You can modify the size of the default buffer pool. The default of one buffer pool will usually suffice.

For a higher level of database tuning, you can create additional buffer pools and assign table spaces to individual buffer pools. For example, you could create four buffer pools, one each for catalogs (small), temporary space, indexes, and data. The extended buffer pool allows DB2 to take advantage of large amounts of memory. See the *Administration Guide* for more details.

Altering the Size of the Default Buffer Pool

The Performance Configuration SmartGuide provides a suggested buffer pool size based on your environment. However, if you want to increase the size of the default buffer pool further, follow the steps below.

1. From the Control Center, expand the database for which you want to alter the buffer pool size.
2. Double-click on the **Buffer Pool** icon. IBMDEFAULTBP is displayed in the contents pane of the Control Center.
3. Click with mouse button 2 on IBMDEFAULTBP, and select **Alter** from the pop-up menu. The Alter Buffer Pool window opens.
4. Modify the **Size in 4 KB pages** field, and click on **OK**.

The change is effective the next time the database is started up (that is, the next time the **db2start** command is issued).

Monitoring Performance

This section provides a brief overview of the Snapshot Monitor and the Event Monitoring tools. It then describes how to set up a simple monitoring session using the Snapshot Monitor, and how to setup a simple monitoring session using an Event Monitor and the Event Analyzer.

The Snapshot Monitor and Event Analyzer provide the following benefits:

- Comprehensive, flexible data collection

  Over 200 performance attributes are supported including buffer pool and I/O, lock and deadlock, sorting, communication, agent, and logging information. Data is shown for database managers, databases, table spaces, tables, buffer pools, connections, transactions, and SQL statements.

- Easy-to-use, intuitive viewing

  Snapshot data can be viewed in real time using easy-to-read graphs or textual views conveniently organized into logical groups. Both details and summary views are provided, with the ability to access more detailed information.

- Powerful data analysis capabilities

  You can customize measurements by allowing spreadsheet-like formulas. For example, rather than monitoring an absolute measurement, you can monitor a ratio calculated from two related measurements.

- Robust alerting capabilities

  For any performance measurement, you can define exception conditions by specifying a threshold value. When the threshold value is reached, you can specify any or all of the following actions: notification through the Alert Center, audible alarm, or execution of a script, program or message. Records are logged by default for alarm and warning thresholds on the **Alerts** page of the Journal.

You use the Snapshot Monitor when you have a problem that is occurring now. It lets you take a snapshot of database activity and performance data at a point in time. You create an event monitor when you need to know completion information such as how long a transaction took or how much CPU a statement used. You then use the Event Analyzer to read the data recorded from the event monitor.

Figure 25 on page 81 illustrates these concepts. With the Snapshot Monitor, for a given point in time, a snapshot is returned for a performance variable. The Snapshot Monitor displays these points for comparison over time. Each point on the graph represents a data value. The steps for using the Snapshot Monitor are provided in “Monitoring Performance at a Point In Time” on page 82.

With an event monitor, two events are being monitored for a given database, connections and statements. For each database connection, there is one connection event record produced. For each statement executed in that connect, there is a statement record produced. Each connection event record maps to one row in the Connections View window of the Event Analyzer. This window shows information for each application that connected during the monitored period, including:

- Application name
- Execution ID
- Connect time
- Total CPU time
- Lock wait time
- Total sort time
- Deadlocks
- Disconnect time
- Application ID

Each statement event record maps to one row in the Statements View window in the Event Analyzer.

The steps for creating an event monitor and viewing the resultant data using the Event Analyzer are provided in “Analyzing an Event for a Period of Time” on page 87.

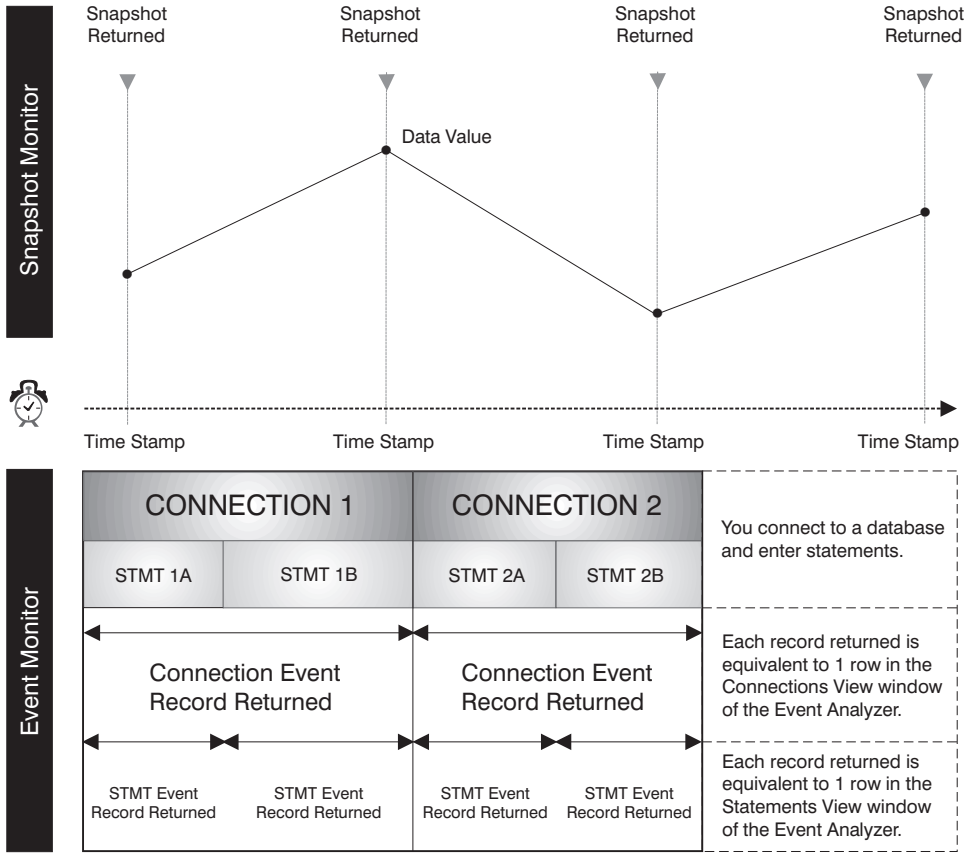

Figure 25. Comparison: Snapshot Monitor and Event Monitor Tools (Event Monitor, Event Analyzer)

Considerations for Monitoring and Tuning a Database

Consider the following before monitoring and tuning a database:

- Define your objectives. For example:
  - Understand how applications use resources at the instance level at a specific point in time. For example, database concurrency is reduced if a special application is started.
  - Understand which instance-level events have occurred running applications. For example, there is poor overall performance running applications.

- Determine what information you will analyze. Areas that cause bottlenecks should be monitored. Some of these might be hardware related and the Snapshot Monitor can help you monitor them. For example:
  - Database connection activity
  - Table space, buffer pool, and I/O activity

  Some might be environment-related and the Event Analyzer can help you monitor them. For example:
  - Too many database tasks are scheduled during peak time
  - There is a high number of user connections
  - Database partitioning (hardware load balancing) is not well optimized
  - The server is being used for more than just a database server

  Some of the visible effects are, for example:
  - Queries/responses are slow
  - Scheduled tasks are not completing on time
  - Applications are timing out

The next section describes how to set up a simple monitor session using the Snapshot Monitor, and how to use the Alert Center to keep track of any performance-related problems.

For details of how to obtain the monitor data, analyze the monitor information, determine what changes are required, and implement the changes, see the online help for the DB2 administration tools.

The monitoring tools are optionally installable wherever you can install the Administration Tools. Chapter 1, “DB2 Administration Tools and Basic Concepts” on page 3 describes where you can install the Administration Tools.

Monitoring Performance at a Point In Time

Use the Snapshot Monitor if you want to do complex data collection and analyze the data to pinpoint potential problems. You can watch performance data change over time.

The Snapshot Monitor lets you:

- Graph performance information
- Define performance variables
- Set the capture frequency of performance snapshots
- View the results of performance calculations
- Define threshold values and threshold actions
- Generate and store alerts

The following types of information are captured:

- Information about long-lived activities (such as database activity when an application is taking too long to complete)

- Counters that keep track of information about the current level of activity (such as the number of open cursors for a database)
- Cumulative information about database activity (such as the maximum number of connections made while a database instance is active, or the total number of SQL statements executed against a particular database)
- Other informational data (such as the time stamp for the last backup)

Taking snapshots at predefined intervals provides a picture of the current state of the activity in the database manager and its applications. This information can be used to:

- Detect performance problems
- Analyze performance trends
- Tune database manager and database configuration parameters
- Analyze the performance of database applications

Performance information is available for the following database objects:

- Instance
- Database
- Table
- Table space
- Database connections

For each, a variety of performance variables can be monitored. The Performance Variable Reference Help, available from the **Help** menu of any Snapshot Monitor window, provides a description of all the performance variables. These variables are organized into categories. By default, all performance variables are monitored, but the categories can be turned on and off through the administration tools. The following categories have been set on by default:

- Instance: Agents, Connections, Sort
- Database: Lock and Deadlock, Buffer Pool and I/O, Connections, Lock and Deadlock, Sort, SQL Statement Activity
- Table: Table
- Table space: Buffer Pool and I/O
- Database Connections: Buffer Pool and I/O, Lock and Deadlock, Sort, SQL Cursors, SQL Statement Activity

From the Control Center, you can have only one Snapshot Monitor capturing snapshots from an instance of a database manager. This means that the API that is used to get snapshot information is issued only once for all monitored database objects in a database manager. This decreases the overhead on a database manager by the Snapshot Monitor.

For detailed information on how to use the Snapshot Monitor, see its online help.

Defining a Simple Monitor Scenario

To monitor the SAMPLE database for the total number of connections:

1. From the Control Center, open the Command Center using the icon on the toolbar and enter the following command to connect to the SAMPLE database:

   ```
   connect to sample
   ```

2. From the Control Center, click with mouse button 2 on the SAMPLE database and select **Snapshot monitoring -> Show monitor profile**. The Monitor Profile notebook opens.

3. On the **Definition** page, if this is the first time you are using the monitor, accept the default profile, db2smpv, that is supplied with the Snapshot Monitor.

   A profile is associated with an instance only, not a specific database.

4. Accept the default of 20 seconds for the **Profile sampling interval**. The more frequent the sampling, the more data you will collect. However, the more frequent the sampling, the higher the chance that performance will be affected.

5. Select the **Perform monitor actions** you prefer. When a threshold is reached, the alert action that you specify later in this scenario will occur either once per warning or alarm, or at every sampling interval during the warning or alarm condition, depending on what you select here.

6. Move through the notebook pages until you reach the **Database** page.

7. In the **Category** group box, select the **Connections** category. It should say (On) beside it to show that this category is turned on for monitoring.

8. In the **Performance variable** list, select the performance variable called Total Connections. This is the performance variable that you will monitor specifically in this example.

9. In the **Threshold** group box, select the upper left check box. This activates the upper alarm threshold entry box.

10. Type the number 0 in the upper alarm threshold entry box. You are telling the monitor that your upper threshold value for the total number of connections to the SAMPLE database is zero.

11. When you selected the upper left threshold check box, you also activated the **Upper alarm actions** group box. Select the type of alarm you want in the **Upper alarm actions** group box. For the purpose of this example, select **Show in Alert Center** for the action taken when an upper threshold value is reached.

    If the upper threshold value of zero is reached, an icon will appear in the Alert Center to represent the database object (in this case the SAMPLE database) that is in a state of alert.

12. Click on **OK** to save your settings and close the notebook.

13. From the Control Center, click with mouse button 2 on the SAMPLE database and select **Snapshot monitoring -> Start Monitoring** from the pop-up menu to begin

monitoring. The database will be monitored continuously until you select **Snapshot monitoring -> Stop Monitoring**.

Performance variables will be monitored for the database at the sampling interval you selected. Some of the performance variables are:

- Total connections
- Remote connections to the instance
- Percentage of remote connections executing
- Local connections to the instance
- Number of local databases with current connections
- Rate of Unit of Work commits (count/second)
- Percentage of executing applications waiting for lock
- Average number of locks held per application
- Deadlocks detected
- Total number of locks currently held by all applications in the database
- The number of SQL statements that were attempted, but failed
- The number of SELECT SQL statements that were executed
- The number of SQL UPDATE, INSERT, and DELETE statements that were executed

14. Next, you will generate some simple activity against the database. Enter the following SQL statement in the Command Center (which you access from the toolbar):

    ```
    select * from employee order by lastname
    ```

    At this point in the scenario, the Alert Center may open automatically. Just ignore it for now and we will work with it later.

15. To see the standard information that is collected for a snapshot taken of the database, from the Control Center, click with mouse button 2 on the SAMPLE database and select **Snapshot monitoring -> Show monitor details** from the pop-up menu. The Performance Details window opens and you can see all the performance information being captured for the SAMPLE database.

    a. Look for your Total Connections performance variable by scrolling down the rows. The Total Connections performance variable will have a red icon in front of it indicating that you have reached an alarm threshold and have generated an alert. The alert indicator icons are red for an alarm, yellow for a warning, and green for a value within its thresholds.

    b. Look at the column called **Value**. This column shows the total connections to the SAMPLE database. Look at the column called **Upper alarm**. This column shows that the upper alarm threshold was set to zero. The **Value** column is equal to or greater than the **Upper alarm** and therefore an alert was generated.

    As you move through the Snapshot Monitor scenario, you can use the Help menu (or F1) to open the online help and see the other tasks you can do from the window.

16. You can see the Total Connections performance variable plotted on a graph. From the Performance Details view, select the Total Connections performance variable

(and any other ones you might be interested in) and open the performance graph by selecting **Performance variable -> Show performance graph**.

17. You can also see a summary of all the databases you are currently monitoring. To open a Monitored Database Summary view:

    a. From the Control Center, click with mouse button 2 on the **Databases** folder object and select **Show monitor summary** from the pop-up menu. The Monitored Databases Summary window opens. It shows the SAMPLE database being monitored. The icon beside the database name is red. This indicates that an alert has been generated for this database.

    b. Scroll across the columns of performance variables until you find Total Connections. The number for Total Connections corresponds to the number of connections you made to the SAMPLE database. The value is greater than zero and therefore an alert was issued.

    c. Later in this example, after you have looked at your performance views and the Alert Center, remember to stop your Snapshot Monitor session.

       To stop your monitoring session, from the Control Center, click with mouse button 2 on the SAMPLE database. Select **Snapshot monitoring -> Stop monitoring** from the pop-up menu to stop the monitoring of the SAMPLE database.

The Alert Center is a simplified view of the Snapshot Monitor that shows you the results of monitoring. It gives you a dynamic view of the database objects in trouble.

Any database you select to be monitored, or any object that is being monitored, will appear as an icon in a state of alert in the Alert Center if they reach any of their threshold values. They appear only for the period of time during which the threshold is exceeded.

If you want to keep a close watch on the objects being monitored, keep the Alert Center open. You can also modify the Control Center settings so the Alert Center opens automatically if a new warning or alarm is added to it. From the Alerts Center you can also temporarily suspend the alerts while monitoring continues.

What Do I Do When an Object Appears in the Alert Center?

The Alert Center will automatically open by default to display any monitored objects that are in a state of alarm or warning (that is, their thresholds have been exceeded). (You can change this default from the Tools Settings window.)

If you see an object in the Alert Center:

1. Double-click on its icon. The Monitoring Details View window opens, showing all the performance variables being monitored.
2. Check the color of the icon and analyze the data.

The color of the icon reflects the condition of the most severe alert within a group of performance variables. A red icon indicates an alarm. A yellow icon indicates a warning.

The data returned for the performance variable is displayed. See the online help available from the **Help** menu of any Snapshot Monitor window for instructions on how to analyze the data.

Analyzing an Event for a Period of Time

The Event Analyzer is also a DB2 performance tool. You use this tool when you want diagnostic information for an event that has taken place. You use the Event Analyzer in conjunction with an event monitor. For example, you can use an event monitor to trace database activity such as connections, transactions, statements, and deadlocks, while a database is active. An event monitor can also record cumulative performance data that is logged when an application disconnects from the database. After the event monitor has created the event monitor file, you look at your performance information using the Event Analyzer.

The Event Monitor tools let you perform the following:

- Create event monitors to monitor the types of database events that are of interest to you.
- Activate an event monitor to start collecting event data. The data is stored in a file.
- Stop an event monitor from collecting event data.
- View the trace-type summary information that is produced by the Event Monitor.
- Remove an event monitor when you no longer have a need for it. You are also given the option to clean up its trace files.
- Display a list of event monitors associated with the database.
- View the definition of an event monitor.

The Event Analyzer lets you view the data generated by an event monitor for the following event types:

- Database connection activity (the period of time between a connection and its disconnection)
- Transactions (units-of-work)
- SQL statement executions
- Detection of deadlock activity

You can create an event monitor for the following event types; however, to view data generated for them, use the db2evmon executable (described in the *Command Reference* and the *System Monitor Guide and Reference*):

- Database activity

- Table space activity
- Table activity

To analyze event data using an event monitor and the Event Analyzer, follow the steps below. They represent only one example of how to create an event monitor for connection and statement events.

1. From the Control Center, click mouse button 2 on the SAMPLE database and select **Monitor events** from the pop-up menu. The Event Monitors window opens.
2. Click on the **Create** icon in the toolbar. The Create Event Monitor window opens.
3. Enter a name for the event monitor in the **Name** field, and select **Connections** and **Statements**. By default, deadlocks are monitored and monitoring starts automatically.
4. Click on **Ok**. The Event Monitors window is refreshed, and now lists the monitor that you have just created.

   The Directory column lists the directory where the event monitor files will be written. Note in this simple example the default directory is being used for the files.
5. Connect to the SAMPLE database:
   a. From the **Control Center** toolbar, click on the **Command Center** icon. The Command Center opens.
   b. Enter **connect to sample** and click on the large push button at the top of the window. You connect to the SAMPLE database.
6. Generate activity against the SAMPLE database by issuing a simple query from the Command Center such as:

   ```
   select * from employee
   ```

   Information will be gathered for connections and SQL statements:

   - Enter **connect reset** to complete the connection event.
   - Close the Command Center.
7. Next, turn off the event monitoring by clicking mouse button 2 on the event monitor you want to use and selecting **Stop Event Monitoring** from the pop-up menu.

   This forces the event monitor to write the trace file. If not turned off, a write would only occur when the buffer is full or all connections end.
8. In this simple scenario, you can view the resultant event data by clicking with mouse button 2 on the event monitor you created, and selecting **View Event Monitor Files** from the pop-up menu. The Monitored Periods View window opens.

   However, in some cases you must use the Event Analyzer to view event monitor data, for example if:

   - Files are moved after they were written. You might use a different directory to store event monitor data files, if you are keeping previous output for comparison purposes.
   - Files reside on remote servers. The Event Monitors window can only access local event monitor files.

To access the Event Analyzer:

a. From the desktop, click on the Event Analyzer icon. The Event Analyzer window opens. (You can also invoke the Event Analyzer from its icon in the Control Center's tool bar.)

b. Enter the default directory for the event monitor in the **Path** field. Click on **Ok**. The Monitored Periods View window opens.

9. Click mouse button 2 on a monitored period, and select **Open as -> Connections** from the pop-up menu. The Connections View window opens. This shows the list of connections that were made during the event monitoring session. (There may be more than one connection listed. The connection you are interested in may not be the first one in the list.)

10. Click mouse button 2 on a connection, and select **Open as -> Statements** from the pop-up menu. The SQL Statements View window opens. All statements for the selected connection are displayed. Columns of information are provided for each statement, including:

    - Operation
    - Package name
    - Creator
    - Start time
    - Elapsed time
    - Total CPU time
    - Text

The online help for the event monitor and the Event Analyzer provide detailed instructions for creating event monitors and viewing the resultant event data.

How To Improve the Performance of a Problem Query Using Visual Explain

Visual Explain is a tool for analyzing and tuning SQL statements. It presents a graphical view of the access plan for explained SQL statements. Tables and indexes, and each operation on them, are represented as nodes, and the flow of data is represented by the links between the nodes. You can use the information available from this graph to find ways to tune your SQL queries for better performance.

Visual Explain captures information about how SQL statements are compiled. This information allows you to understand the plan and potential execution performance of SQL statements.

This information can help you:

- Assist in designing application programs.

- Assist in database design.
- Understand how two tables are joined: the join method, the order in which the tables are joined, and the occurrence and type of sorts.
- Determine ways of improving the performance of SQL statements (for example, by creating a new index).
- View the statistics that were used at the time of optimization. You can then compare these statistics to the current catalog statistics to help you determine whether re-binding the package might improve performance. It also helps you determine whether collecting statistics might improve performance.
- Determine whether or not an index was used to access a table. If an index was not used, Visual Explain can help you determine which columns could be included in an index to help improve query performance.
- View the effects of performing various tuning techniques for the purpose of better performance, by comparing the before and after versions of the access plan graph for a query.
- Obtain information about each operation in the access plan, including the total estimated cost and number of rows retrieved.

How to Analyze a Simple Dynamic SQL Statement

This section provides a simple example of how to get started analyzing a dynamic SQL query.

1. From the Control Center, click mouse button 2 on the SAMPLE database, and select **Explain SQL** from the pop-up menu. The Explain SQL Statement window opens.
2. In the **SQL text** field, enter the following SQL statement:

   ```
   select * from staff order by name
   ```

3. Click on **OK**. The Access Plan Graph window opens. The graph represents the path that the optimizer chose as the most efficient in order to provide the results for your query.

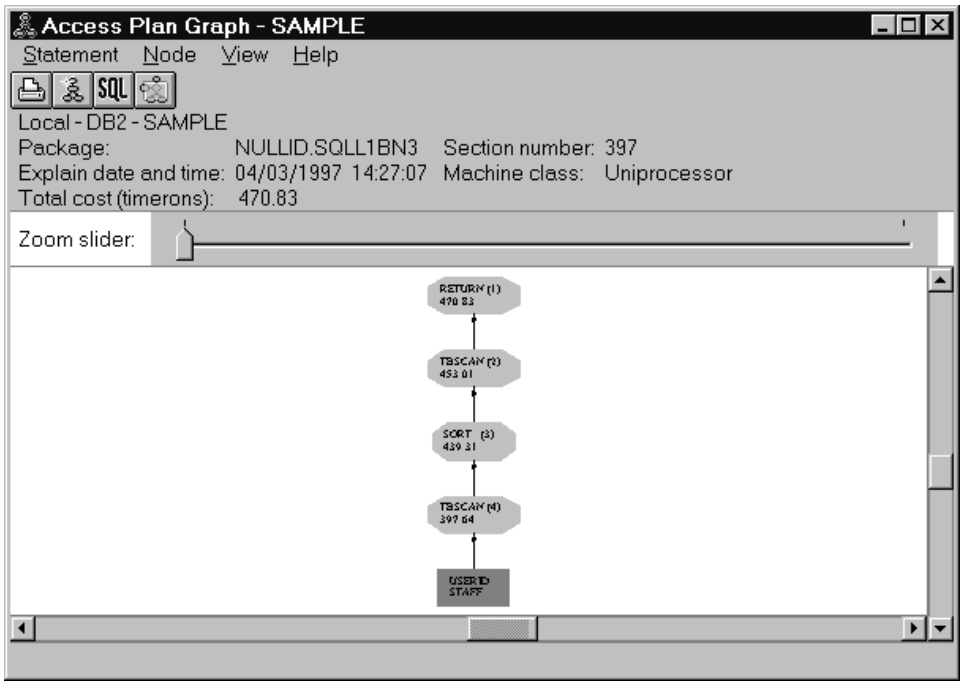

Figure 26. Access Plan Graph

4. Optional: Double-click on any of the nodes (for example, the RETURN operator node). The Operator Details window opens, showing the details for that operator.

The explained SQL statement is saved automatically. To view it later:

1. From the Control Center, click mouse button 2 on the SAMPLE database, and select **Show explained statements history** from the pop-up menu. The Explained Statements History window opens.
2. Locate the entry you want. You can look at the **SQL text** column to see the SQL statement you had previously explained.
3. Click mouse button 2 on the entry, and select **Show access plan** from the pop-up menu. The Access Plan Graph window opens.

The online help for Visual Explain (accessible from the **Help** menu) provides details on how to interpret the Access Plan Graph window in order to improve the performance of SQL statements. The online help also contains detailed examples to help you learn how to use Visual Explain. From the online help's table of contents, see the heading “Analyzing SQL statements using Visual Explain”, and the subheading “Introduction to Visual Explain”.

Other Areas to Watch

Some other items to consider in helping to achieve maximum performance are whether your system requires more memory or more CPU. In order to check these, you must use the facilities provided by the operating system you are running.

If paging occurs, you need to reduce overall memory consumption by DB2 and other applications. The buffer pool commonly uses a lot of memory which results in performance improvements through reduced I/O. However, if the buffer pool is too big it can cause paging. The Performance Configuration SmartGuide provides recommendations for the size of the buffer pool.

You should also consider that the application you are running might be the source of performance problems. The *Road Map to DB2 Programming* provides an overview of the tools available to help you determine the cause of the problem in the application so that it can be fixed. (This assumes the application was written in-house.)

Managing Initialization Overhead

When an application is started, you can improve the time that is spent on database initialization with the **activate database** command. If a database has not been started, and a connect is encountered in an application, then the application must wait while the database manager starts up the required database before it can do any work with that database. This is a startup "cost" that is borne only by the first application to access a particular database. Once the database is started, all other applications can connect to it and use it without this time cost.

Databases activated with the **activate** command can be shut down using the **Stop** action from the pop-up menu for an instance, or using the **deactivate database** command. For more information on these commands, see the *Command Reference*.

Part 4. Appendixes

Appendix A. Working with Scripts and Jobs

This chapter introduces you to the Command Center, the Script Center, and the Journal, and describes how you can use them together. They are all accessible from the Control Center.

The Command Center and Script Center provide interactive windows where you can:

- Execute SQL statement and DB2 commands. From the Script Center, you can execute operating system commands from a script by doing the following:

  1. Select **Script -> New**. The New Command Script window opens.
  2. For **Script Type** select the **OS command** radio button.

  From the Command Center, you can execute operating system commands by preceding them with an exclamation mark (!).

- See the execution result of one or many SQL statements and DB2 commands in a result window. You can scroll through the results and generate a report of the execution.
- Create, save, execute, and schedule command scripts.
- Get quick access to the DB2 administration tools such as the Control Center from the main toolbar.
- From the Command Center, see the access plan and statistics associated with an SQL statement before execution. You do this by selecting the **Access plan** menu item, or the **Access Plan** page, or selecting the **Create access plan** icon in the toolbar.

Creating and Saving a Command Script

A command script lets you issue many commands at once, without the need to type and execute each command individually. A command script can be scheduled to run as a job at a time that you specify. From the Command Center, a script file can contain logic in a supported operating system script language, such as REXX, if you precede the operating system commands with an exclamation mark (!).

The following are the high-level steps for creating and saving a command script using the Command Center. The benefit of creating a script in the Command Center instead of the Script Center is that you can execute the script first before saving it into the Script Center. (To create a script in the Script Center, select **Script -> New**.)

1. From the **Control Center** toolbar, click on the **Command Center** icon. The Command Center opens. You can also start the Command Center from its own icon in the **Server Administration** folder.

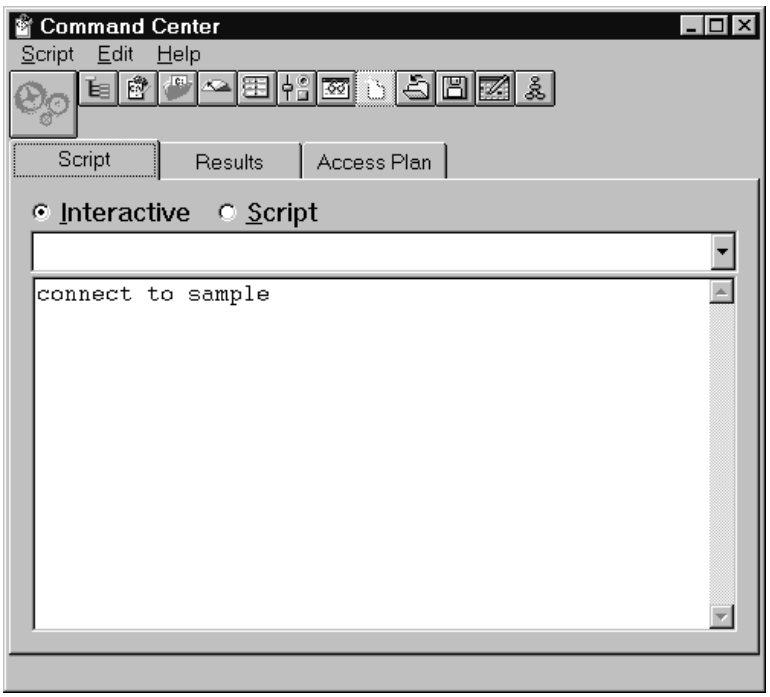

Figure 27. Command Center

2. Click on the **Script** radio button and enter your commands in the **Script** page of the Command Center. (If you execute a command now by clicking on the “gears” push button, or by selecting **Execute** from the **Script** menu, the results of the command will be displayed on the **Results** page of the Command Center.)

   The **Script** page has two modes: interactive and script. In script mode the entry area acts like an editor. Script mode is intended for you to create and edit a script. In interactive mode it acts like a command entry prompt and the command entry is cleared each time you execute a command. Interactive mode is the default.

3. From the **Script** menu, select **Save as**. The Save window opens.
4. Complete the fields, and select **To Script Center**. The command script is saved, and will be accessible through the Script Center.

You can edit a script inside the Script Center or outside the Script Center using your own editor. If you run a script outside the Script Center, the results will not be logged in the Journal.

Using an Existing Script with the Script Center

To use the Script Center with your pre-existing scripts, that is, those not created with the Script Center:

1. In the **Control Center** toolbar, click on the **Script Center** icon. The Script Center opens.
2. Select **Script -> Import**. The File Browser window opens.
3. Select an existing script file and click on **OK**. The New Command Script window opens. The script will be displayed in the lower part of the window which is a script editor. Complete the **Instance**, **Script name**, **Script description**, and **Working directory** fields, and select a **Script type**.
4. Click on **OK**. The script will be created in the Script Center.

Scheduling a Saved Command Script To Run

To schedule the script job:

1. Click on the **Script Center** icon on the Control Center toolbar. The Script Center opens.

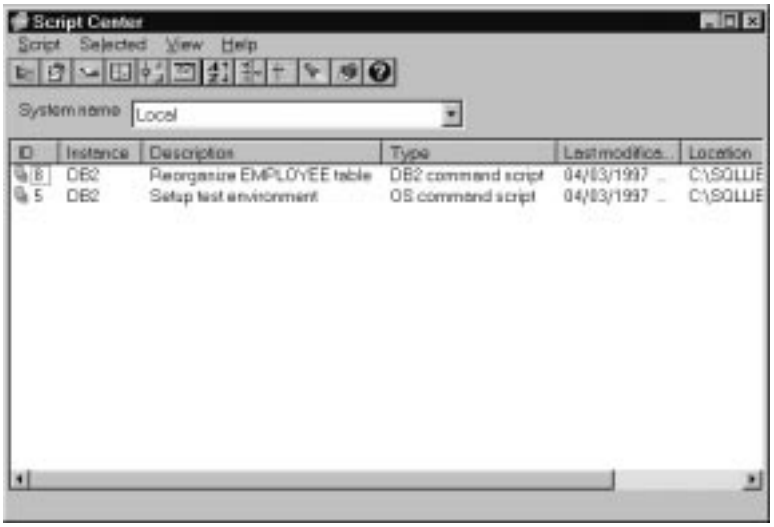

Figure 28. Script Center

Here, you can view information (for example, description and script type) about all command scripts that are known to the system, and you can perform the following tasks:

- Create a command script that contains DB2 and operating system commands
- Run a saved command script immediately
- Schedule a script to run at a later date or at a regular interval
- Access the Journal from the toolbar to see the jobs that use a particular script and to see the status of all scheduled jobs
- Edit a saved command script

For example, you might want to create a script that will collect statistics for several tables. You could then schedule the job to run overnight.

2. Click with mouse button 2 on the script you want to schedule for execution, and select **Schedule** from the pop-up menu. The Scheduler window opens.
3. If you wish, select the frequency for the job, and a completion action such as a completion message or another command script to be launched.

   You can schedule jobs by running them unattended at a given time every x hours / days / weeks / months, multiple times a week, or multiple times a month.
4. Click on **OK**.

Working with Jobs

Use the Journal to work with jobs. To open the Journal:

1. Click on the **Journal** icon from the Script Center toolbar. The Journal opens.

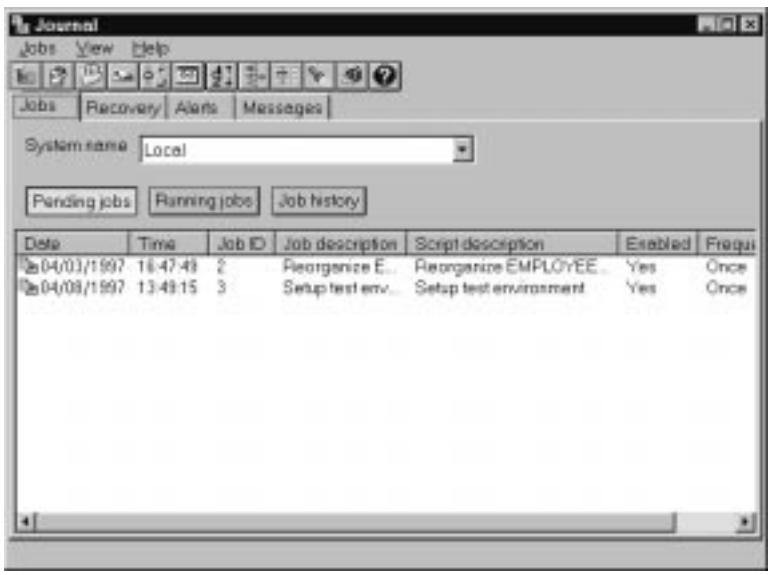

Figure 29. Journal

2. Click on the **Pending jobs** push button to see your job listed, to see all information about jobs, and to perform actions on the job such as rescheduling it, showing the scripts associated with it (likewise, from the Script Center you can see the jobs with which a script is associated), or running it immediately. When a saved script is modified, all jobs that are dependent on it will inherit the new modified behavior.

The other pages in the Journal window are:

- The **Recovery** page: Displays the recovery history (the details from backup and restore operations, and load operations) and lets you restore the recovery log.
- The **Alerts** page: Logs all alerts.
- The **Messages** page: Logs all messages issued through the DB2 administration tools.

The online help for the Journal provides detailed steps for working with jobs and logs.

Appendix B. Enabling Remote Administration

This chapter presents topics related to administering a remote database:

- Managing DB2 server instances accessible to remote clients
- Starting and stopping an instance

The tasks described in this appendix assume that you have administrative authority (DB2 SYSADM authority). That is, you should log on using the user ID that was created according to the instructions in the *Quick Beginnings* book for your platform. This user ID is necessary when you attempt to perform a task at the instance or database level that requires authority checking.

Managing DB2 Server Instances Accessible To Remote Clients

If you will need to manage a remote database, this section shows you how to:

- Add a remote system
- Add the instance you want to work with for that system
- Add the database you want to work with under that instance

DB2 first checks in the node directory (which contains an entry for all servers to which a database client can connect and the communications protocol used in the connection) to see if the remote system is already known. If it is not, you need to perform the steps in this section if you want to work with a system, instance, or database on a remote system. Essentially, you set yourself up as a client to the remote system.

After you install DB2, you can use the Client Configuration Assistant to search the network for systems, instances, and databases and then choose to configure communications for them (catalog the instances and databases, which creates an entry for them in the node directory and database directory, respectively). When this is done, they will be displayed in the Control Center so that you can work with them. See the *Quick Beginnings* book for your platform for instructions on using the Client Configuration Assistant. However, if this has not already been done, you can follow the steps below.

You must first add the system (catalog it, which creates an entry for it in the node directory) so that its instances and databases can be made known. Next, you must

add the instances and databases for the system (catalog them, which creates an entry for them in the node directory and database directory, respectively).

To add a remote system:

1. From the Control Center, click with mouse button 2 on the **Systems** object and select **Add**. The Add System window opens.
2. Enter the system name in the **System name** field.

   If the **Discover** configuration parameter for the instance is set to **search** and the **discover comm** configuration parameter is not blank, you can select **Refresh** to get a list of the remote systems. You can then select one of the systems from the list below the **System name** field.
3. Type the remote instance name in the **Remote instance name** field.
4. Select the type of operating system for the remote system from the **Operating system** list.
5. Select the protocol you want used for communications with the remote locations. For a local system, Local is automatically selected and is the only valid protocol. The possible protocols are:

   - APPC
   - IPX/SPX
   - NetBIOS
   - TCP/IP
   - Named pipe (Windows NT and Windows 95 operating systems only)

   Only the protocols that the computer is currently set up for appear in the listbox.
6. Enter the appropriate **Protocol parameters**.
7. Optional: Enter a comment to be associated with the system.
8. Select **Apply** to add the system to the node directory.

Next, add the instance you want to work with on that system:

1. From the Control Center, click with mouse button 2 on the **Instances** object belonging to the system you just added.
2. Select **Add**. The Add Instance window opens.
3. Enter the required values in the fields.
4. Click on the **Refresh** button to have a list of existing instances displayed.
5. Select the instance you want to work with.
6. Click on the **Apply** push button, then the **Close** push button.

Finally, add the database you want to work with under that instance:

1. From the Control Center, click with mouse button 2 on the **Databases** object.
2. Select **Add**. The Add Database window opens.

3. Enter the database name, type of communication protocol, and, optionally, an alias. An alias in this case is an alternative name used to identify a database.
4. Click on the **Refresh** button to have a list of existing databases displayed for that instance.
5. Select the database you want to work with.
6. Click on the **Apply** push button, then the **Close** push button.

Starting and Stopping an Instance

In this book you use the default instance that was created when DB2 was installed. It is called DB2. At some point, however, you might want to create another instance. For example, you can isolate a development environment from a production environment by defining a different instance for each. To create additional instances, use the instance creation tool (DB2ICRT). See the *Quick Beginnings* book for your platform.

The instance must be started on the server before you can perform operations on it or its databases. The DB2 Control Center automatically starts the local instance you select if it is not already started. However, you might want to use the Control Center to stop or start a different instance; for example, if you want to start a remote instance as a test, or if you have changed configuration parameters at the instance level. (Instance-level configuration parameters take effect when you start the instance.) The steps are provided below.

Note that at installation time you can select an option to indicate that you want the default DB2 instance started each time the system is started. As well, from the Tools Settings window in the Control Center (select the **Tools Settings** icon from the Control Center toolbar), you can select to automatically start the local DB2 instance whenever one of the administration tools is started.

To start an instance:

1. Select the instance.
2. Click on it with mouse button 2.
3. Select **Start**.

To shut down all activity on an instance:

1. From the Control Center, select one or more instances you want to stop. Their database(s) will be shut down.
2. Click with mouse button 2 on any of the instances you selected and select **Stop** from the pop-up menu. The Confirm Stop window opens listing all of the instances that you selected.

3. Optional: To disconnect all applications from selected instances, select the **Disconnect all applications** check box.

   If any applications are running that affect database integrity, these applications will complete before the instance stops.

4. To stop all of the instances selected in the box, click on **OK**. Any unselected instances are left untouched.

At this point, you can shut down the operating system if you need to.

Appendix C. Terminology Map

For those of you familiar with other database systems, the following table is available to help you understand the terminology used by DB2. It provides a list of basic DB2 administration-related terms that are equivalent to terms used for Microsoft SQL Server, Oracle, Informix, and Sybase. (n/a means not applicable.)

DB2 Universal Database V5	Microsoft SQL Server 6.5	Oracle V7.3	Informix OnLine V7.2	Sybase SQL Server V11
Physical Layer				
Table space	n/a	Tablespace	dbspace	n/a
SMS table space	n/a	n/a	n/a	n/a
DMS table space	n/a	table space	dbspace	n/a
Container (raw or file)	Database device (file)	Data file or raw disk	Chunk (cooked or raw disk space)	Database device (raw or file)
Logical Layer				
Server	Server (only 1 per system)	Server	Database server (many databases)	Server
Instance (1 or more. Each instance manages 1 or more databases.)	n/a	Instance (1 database to 1 instance)	n/a	n/a
Database	Database	Database	Database	Database
Database directories (created by DB2)	SQL Enterprise Manager registry	tnsnames.ora	sqlhosts file	Interfaces file
Node directory (created by DB2)	SQL Enterprise Manager registry	SQL*Net configuration files	sqlhosts file	Interfaces file
Database Manager Configuration File	NT registry	init.ora	onconfig file	Internal to the database
Database Configuration File	NT registry	init.ora	onconfig file	Internal to the database
Catalog tables	System tables (master database and those in each database)	Data dictionary	System catalog tables (also data dictionary)	System tables (master database and those in each database)
Database Objects				
Schema	Schema	Schema	Schema	Schema
Table	Table	Table	Table	Table
Table constraint	Rule and table constraint	Table constraint	Table constraint	Rule and table constraint

DB2 Universal Database V5	Microsoft SQL Server 6.5	Oracle V7.3	Informix OnLine V7.2	Sybase SQL Server V11
View	View	View	View	View
Index	Index	Index	Index	Index
Recovery log	Transaction log	Redo log and rollback segments	Logical log and physical log	Transaction log
Archive log	Transaction log dump	Archived redo log	Logical log backups	Transaction log dump
Users and user groups (operating system)	Users and database groups	Users and roles	Users and roles	Users and database groups
Package	n/a	n/a	n/a	n/a
Sample database (called sample)	Sample database (called pubs)	Sample tables ("Scott's" schema)	Sample database (called stores7)	Sample database (called pubs2)
Administration / Usage				
Control Center	SQL Enterprise Manager	Enterprise Manager	Enterprise Command Center	SQL Server Manager
Tables assigned to table spaces, containers assigned to table spaces	Database devices assigned to databases, tables assigned to databases	Tables assigned to tablespaces, datafiles assigned to tablespaces	Tables assigned to dbspaces, chunks assigned to dbspaces	Database devices assigned to databases, tables assigned to databases
Administration commands and statements	System stored procedures	Administration commands and statements	Online utilities	System stored procedures
Binding (a DB2 utility or a program so you can use it)	n/a	n/a	n/a	n/a
Backup database	Dump database	Backup database	Archive database	Dump database
Archive online log files (transaction log)	Dump transaction	Archive redo log files	Backup (logical log files)	Dump transaction
Restore from backup	Load database	Restore from backup	Physical restore	Load database
Roll-forward recovery	Load transaction	Recover	Logical restore	Load transaction
Crash recovery	Automatic recovery	Instance recovery	Fast recovery	Automatic recovery
Run statistics	Update statistics	Analyze	Update statistics	Update statistics
Load, Import, Export	bcp	SQL*Loader, Import, Export	Onload, Onunload, Onpload	bcp
Command Processor	ISQL/w	SQL*Plus	DB Access	ISQL

Appendix D. Basic Introduction to Troubleshooting

DB2 provides a *Troubleshooting Guide* that is intended for technical support representatives for DB2 servers and clients. It helps you:

- Identify problems or errors in a concise manner
- Solve problems based on their symptoms
- Use available diagnostic tools
- Develop a troubleshooting strategy for your day-to-day DB2 operations

The *Troubleshooting Guide* presents these basic troubleshooting topics:

- Good troubleshooting practices
- Troubleshooting on the server
- Troubleshooting on the client
- Troubleshooting host communications
- Troubleshooting applications
- Troubleshooting DB2 Extended Enterprise Edition

The *Troubleshooting Guide* presents these advanced troubleshooting topics:

- The DB2 process model
- Using logged information
- Taking traces
- Diagnostic tools for UNIX-based, OS/2, and Microsoft Windows operating systems

Up-to-date bulletins and technical documentation are available from the World Wide Web at http://www.software.ibm.com/data/db2/library/.

See the section at the end of this book for the details of how to contact IBM.

Appendix E. About DB2 Universal Database

DB2 is a relational database management system that is web-enabled with Java support; scalable from single processors to symmetric multi-processors; and multimedia capable with image, audio, video, and text support. The DB2 product consists of three products and components: *DB2 Workgroup Edition*, *DB2 Enterprise Edition*, and *DB2 Client Application Enabler*. These products have the following features.

DB2 Workgroup Edition

DB2 Workgroup Edition includes:

- *DB2 Universal Database server*, which enables local and remote clients and applications to create, update, control, and manage relational databases using Structured Query Language (SQL), ODBC, or CLI.
- *DB2 Client Pack* CD-ROM, which contains all the latest DB2 Client Application Enablers. With DB2 Client Application Enabler, clients from a variety of platforms can connect to any DB2 server, including DataJoiner.
- *DB2 Net.Data* CD-ROM, which contains all supported DB2 Net.Data (formerly known as DB2 World Wide Web Connection) products. DB2 Net.Data enables application developers to create Internet applications that access data from DB2 databases.

DB2 Workgroup Edition is licensed on a per user basis.

DB2 Enterprise Edition

DB2 Enterprise Edition includes all the functions provided in DB2 Workgroup Edition, plus support for host connectivity providing users with access to DB2 databases residing on host systems such as MVS/ESA, OS/390, OS/400, VM, and VSE.

DB2 Enterprise Edition supports unlimited user licensing.

DB2 Client Application Enabler

DB2 Client Application Enabler enables a client workstation to access the DB2 server.

Other DB2 Products

There are a variety of other DB2 products that you can order separately:

DB2 Application Developer's Kit

The DB2 Application Developer's Kit contains a collection of DB2 Universal Database products, clients, DB2 Connect products, Software Developer's Kits, and application development tools for all supported platforms. The AD Kit gives you all the tools you need to create multimedia database applications that can run on a variety of platforms and can connect to any DB2 Server, including DataJoiner.

DB2 Universal Database Personal Edition

DB2 Universal Database Personal Edition enables you to create and use local databases and to access remote databases if they are available. This product is available only for the OS/2, Windows 95, and the Windows NT operating system.

DB2 Connect Enterprise Edition

DB2 Connect Enterprise Edition (formerly known as DDCS Multi-User Gateway) provides access from clients on the network to DB2 databases residing on host systems such as MVS/ESA, OS/390, OS/400, VM, and VSE.

DB2 Connect Personal Edition

DB2 Connect Personal Edition (formerly known as DDCS Single-User) provides access from a single workstation to DB2 databases residing on host systems such as MVS/ESA, OS/390, OS/400, VM, and VSE. This product is available only for the OS/2, Windows 95, and the Windows NT operating system.

DB2 Universal Database Extended Edition

DB2 Universal Database Extended Edition (formerly known as DB2 Parallel Edition) provides the ability for a database to be partitioned across multiple independent computers of a common platform, each with its own processor(s), memory, and DASD. To the end-user and application developer, the database still appears as a single database on a single computer. This enables an application to use a database that is simply too large for a single computer to handle efficiently. SQL operations can operate in parallel on the individual database partitions, thereby speeding up the execution time of a single query.

DB2 Universal Database Tools

DB2 Universal Database includes the following tools to perform server administration tasks.

Note: Not all tools are available on every platform.

Control Center

A graphical interface that displays database objects (such as databases, tables, and packages) and their relationship to each other. Use the Control Center to perform administrative tasks such as configuring the system, managing directories, backing up and recovering the system, scheduling jobs, and managing media.

The Control Center includes the following facilities:

- *Command Center:* to enter DB2 commands and SQL statements in an interactive window, and to see the execution result in a result window. You can scroll through the results and save the output to a file.
- *Script Center:* to create mini applications called scripts, which you can store and invoke at a later time. These scripts can contain DB2 commands, SQL statements, or operating system commands. You can schedule scripts to run unattended. You can run these jobs once or you can set them up to run on a repeating schedule. A repeating schedule is particularly useful for tasks like backups.
- *Journal:* to view the following types of information: all available information about jobs that are pending execution, executing, or that have completed execution; the recovery history log; the alerts log; and the messages log. You can also use the Journal to review the results of jobs that run unattended.
- *Alert Center:* to monitor your system for early warnings of potential problems, or to automate actions to correct problems.
- *Tools Setting:* to change the settings for the Control Center, Alert Center, and Replication.

Performance Monitor

An installable option for the Control Center, the Performance Monitor is a graphical interface that provides comprehensive performance data collection, viewing, reporting, analysis, and alerting capabilities for your DB2 system. Use the Performance Monitor for performance tuning.

You can choose to monitor snapshots or events. The Snapshot Monitor enables you to capture point-in-time information at specified intervals. The Event Monitor allows you to record performance information over the duration of an event, such as a connection.

Visual Explain

An installable option for the Control Center, Visual Explain is a graphical interface that enables you to analyze and tune SQL statements, including viewing access plans chosen by the optimizer for SQL statements.

Appendix F. How the DB2 Library Is Structured

The DB2 Universal Database library consists of SmartGuides, online help, and books. This section describes the information that is provided, and how to access it.

To help you access product information online, DB2 provides the Information Center on OS/2, Windows 95, and the Windows NT operating systems. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. “About the Information Center” on page 120 has more details.

SmartGuides

SmartGuides help you complete some administration tasks by taking you through each task one step at a time. SmartGuides are available on OS/2, Windows 95, and the Windows NT operating systems. The following table lists the SmartGuides.

SmartGuide	Helps you to...	How to Access...
Add Database	Catalog a database on a client workstation.	From the Client Configuration Assistant, click on **Add**.
Create Database	Create a database, and to perform some basic configuration tasks.	From the Control Center, click with the right mouse button on the **Databases** icon and select **Create**->**New**.
Performance Configuration	Tune the performance of a database by updating configuration parameters to match your business requirements.	From the Control Center, click with the right mouse button on the database you want to tune and select **Configure performance**.
Backup Database	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to backup and select **Backup**->**Database using SmartGuide**.
Restore Database	Recover a database after a failure. It helps you understand which backup to use, and which logs to replay.	From the Control Center, click with the right mouse button on the database you want to restore and select **Restore**->**Database using SmartGuide**.

SmartGuide	Helps you to...	How to Access...
Create Table	Select basic data types, and create a primary key for the table.	From the Control Center, click with the right mouse button on the **Tables** icon and select **Create**->**Table using SmartGuide**.
Create Table Space	Create a new table space.	From the Control Center, click with the right mouse button on the **Table spaces** icon and select **Create**->**Table space using SmartGuide**.

Online Help

Online help is available with all DB2 components. The following table describes the various types of help.

Type of Help	Contents	How to Access...
Command Help	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter: **?** *command* where *command* is a keyword or the entire command. For example, **?** *catalog* displays help for all the CATALOG commands, whereas **?** *catalog database* displays help for the CATALOG DATABASE command.
Control Center Help	Explains the tasks you can perform in a window or notebook. The help includes prerequisite information you need to know, and describes how to use the window or notebook controls.	From a window or notebook, click on the **Help** push button or press the F1 key.

Type of Help	Contents	How to Access...
Message Help	Describes the cause of a message number, and any action you should take.	From the command line processor in interactive mode, enter: **?** *message number* where *message number* is a valid message number. For example, **?** *SQL30081* displays help about the SQL30081 message. To view message help one screen at a time, enter: **?** *XXXnnnnn* **| more** where *XXX* is the message prefix, such as SQL, and *nnnnn* is the message number, such as 30081. To save message help in a file, enter: **?** *XXXnnnnn* > *filename.ext* where *filename.ext* is the file where you want to save the message help. **Note:** On UNIX-based systems, enter: **\?** *XXXnnnnn* **| more** or **\?** *XXXnnnnn* > *filename.ext*
SQL Help	Explains the syntax of SQL statements.	From the command line processor in interactive mode, enter: **help** *statement* where *statement* is an SQL statement. For example, **help** *SELECT* displays help about the SELECT statement.
SQLSTATE Help	Explains SQL states and class codes.	From the command line processor in interactive mode, enter: **?** *sqlstate* or **?** *class-code* where *sqlstate* is a valid five digit SQL state and *class-code* is a valid two digit class code. For example, **?** *08003* displays help for the 08003 SQL state, whereas **?** *08* displays help for the 08 class code.

DB2 Books

The table in this section lists the DB2 books. They are divided into two groups:

- Cross-platform books: These books are for DB2 on any of the supported platforms.
- Platform-specific books: These books are for DB2 on a specific platform. For example, there is a separate *Quick Beginnings* book for DB2 on OS/2, Windows NT, and UNIX-based operating systems.

Most books are available in HTML and PostScript format, and in hardcopy that you can order from IBM. The exceptions are noted in the table.

You can obtain DB2 books and access information in a variety of different ways:

View To view an HTML book, you can do the following:

- If you are running DB2 administration tools on OS/2, Windows 95, or the Windows NT operating systems, you can use the Information Center. “About the Information Center” on page 120 has more details.
- Use the open file function of the Web browser supplied by DB2 (or one of your own) to open the following page:

  sqllib/doc/html/index.htm

  The page contains descriptions of and links to the DB2 books. The path is located on the drive where DB2 is installed.

  You can also open the page by double-clicking on the **DB2 Online Books** icon. Depending on the system you are using, the icon is in the main product folder or the Windows Start menu.

Search To search for information in the HTML books, you can do the following:

- Click on **Search the DB2 Books** at the bottom of any page in the HTML books. Use the search form to find a specific topic.
- Click on **Index** at the bottom of any page in an HTML book. Use the Index to find a specific topic in the book.
- Display the Table of Contents or Index of the HTML book, and then use the find function of the Web browser to find a specific topic in the book.
- Use the bookmark function of the Web browser to quickly return to a specific topic.
- Use the search function of the Information Center to find specific topics. “About the Information Center” on page 120 has more details.

Print To print a book on a PostScript printer, look for the file name shown in the table.

Order To order a hardcopy book from IBM, use the form number.

Book Name	Book Description	Form Number File Name
	Cross-Platform Books	
Administration Getting Started	Introduces basic DB2 database administration concepts and tasks, and walks you through the primary administrative tasks.	S10J-8154 db2k0x50
Administration Guide	Contains information required to design, implement, and maintain a database to be accessed either locally or in a client/server environment.	S10J-8157 db2d0x50
API Reference	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	S10J-8167 db2b0x50
CLI Guide and Reference	Explains how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.	S10J-8159 db2l0x50
Command Reference	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	S10J-8166 db2n0x50
DB2 Connect Enterprise Edition Quick Beginnings	Provides planning, installing, configuring, and using information for DB2 Connect Enterprise Edition. Also contains installation and setup information for all supported clients.	S10J-7888 db2cyx50
DB2 Connect Personal Edition Quick Beginnings	Provides planning, installing, configuring, and using information for DB2 Connect Personal Edition.	S10J-8162 db2c1x50
DB2 Connect User's Guide	Provides concepts, programming and general using information about the DB2 Connect products.	S10J-8163 db2c0x50
DB2 Connectivity Supplement	Provides setup and reference information for customers who want to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA Application Requesters with DB2 Universal Database servers, and customers who want to use DRDA Application Servers with DB2 Connect (formerly DDCS) application requesters. **Note:** Available in HTML and PostScript formats only.	No form number db2h1x50
Embedded SQL Programming Guide	Explains how to develop applications that access DB2 databases using embedded SQL, and includes discussions about programming techniques and performance considerations.	S10J-8158 db2a0x50
Glossary	Provides a comprehensive list of all DB2 terms and definitions. **Note:** Available in HTML format only.	No form number db2t0x50

Book Name	Book Description	Form Number File Name
Installing and Configuring DB2 Clients	Provides installation and setup information for all DB2 Client Application Enablers and DB2 Software Developer's Kits. **Note:** Available in HTML and PostScript formats only.	No form number db2iyx50
Master Index	Contains a cross reference to the major topics covered in the DB2 library. **Note:** Available in PostScript format and hardcopy only.	S10J-8170 db2w0x50
Message Reference	Lists messages and codes issued by DB2, and describes the actions you should take.	S10J-8168 db2m0x50
Replication Guide and Reference	Provides planning, configuring, administering, and using information for the IBM Replication tools supplied with DB2.	S95H-0999 db2e0x50
Road Map to DB2 Programming	Introduces the different ways your applications can access DB2, describes key DB2 features you can use in your applications, and points to detailed sources of information for DB2 programming.	S10J-8155 db2u0x50
SQL Getting Started	Introduces SQL concepts, and provides examples for many constructs and tasks.	S10J-8156 db2y0x50
SQL Reference	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views.	S10J-8165 db2s0x50
System Monitor Guide and Reference	Describes how to collect different kinds of information about your database and the database manager. Explains how you can use the information to understand database activity, improve performance, and determine the cause of problems.	S10J-8164 db2f0x50
Troubleshooting Guide	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169 db2p0x50
What's New	Describes the new features, functions, and enhancements in DB2 Universal Database. **Note:** Available in HTML and PostScript formats only.	No form number db2q0x50
Platform-Specific Books		
Building Applications for UNIX Environments	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a UNIX system.	S10J-8161 db2axx50
Building Applications for Windows and OS/2 Environments	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Windows or OS/2 system.	S10J-8160 db2a1x50

Book Name	Book Description	Form Number File Name
DB2 Extended Enterprise Edition Quick Beginnings	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for AIX.	S72H-9620 db2v3x50
DB2 Personal Edition Quick Beginnings	Provides planning, installing, configuring, and using information for DB2 Universal Database Personal Edition on OS/2, Windows 95, and the Windows NT operating systems.	S10J-8150 db2i1x50
DB2 SDK for Macintosh Building Your Applications	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Macintosh system. **Note:** Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0528 sqla7x02
DB2 SDK for SCO OpenServer Building Your Applications	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SCO OpenServer system. **Note:** Available for DB2 Version 2.1.2 only.	S89H-3242 sqla9x02
DB2 SDK for Silicon Graphics IRIX Building Your Applications	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Silicon Graphics system. **Note:** Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S89H-4032 sqlaax02
DB2 SDK for SINIX Building Your Applications	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SINIX system. **Note:** Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0530 sqla8x00
Quick Beginnings for OS/2	Provides planning, installing, configuring, and using information for DB2 Universal Database on OS/2. Also contains installing and setup information for all supported clients.	S10J-8147 db2i2x50
Quick Beginnings for UNIX	Provides planning, installing, configuring, and using information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for all supported clients.	S10J-8148 db2ixx50
Quick Beginnings for Windows NT	Provides planning, installing, configuring, and using information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for all supported clients.	S10J-8149 db2i6x50

Notes:

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name `db2d0e50` indicates that the *Administration Guide* is in English. The following letters are used in the file names to indicate the language of a book:

Language	Identifier	Language	Identifier
Brazilian Portuguese	B	Hungarian	H
Bulgarian	U	Italian	I
Czech	X	Norwegian	N
Danish	D	Polish	P
English	E	Russian	R
Finnish	Y	Slovenian	L
French	F	Spanish	Z
German	G	Swedish	S

2. For late breaking information that could not be included in the DB2 books, see the README file. Each DB2 product includes a README file which you can find in the directory where the product is installed.

About the Information Center

The Information Center provides quick access to DB2 product information. The Information Center is available on OS/2, Windows 95, and the Windows NT operating systems. You must install the DB2 administration tools to see the Information Center.

Depending on your system, you can access the Information Center from the:

- Main product folder
- Toolbar in the Control Center
- Windows Start menu.

The Information Center provides the following kinds of information. Click on the appropriate tab to look at the information:

Tasks Lists tasks you can perform using DB2.

Reference Lists DB2 reference information, such as keywords, commands, and APIs.

Books Lists DB2 books.

Troubleshooting Lists categories of error messages and their recovery actions.

Sample Programs Lists sample programs that come with the DB2 Software Developer's Kit. If the Software Developer's Kit is not installed, this tab is not displayed.

Web Lists DB2 information on the World Wide Web. To access this information, you must have a connection to the Web from your system.

When you select an item in one of the lists, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides search capabilities so you can look for specific topics, and filter capabilities to limit the scope of your searches.

Appendix G. Notices

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the

IBM Director of Licensing,
IBM Corporation,
500 Columbus Avenue,
Thornwood, NY, 10594
USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Canada Limited
Department 071
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

This publication may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM
ADSTAR
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
CICS
C Set++
C/370
DATABASE 2
DatagLANce
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
Distributed Relational Database Architecture
DRDA
Extended Services
FFST
First Failure Support Technology
IBM
IMS
Lan Distance
MVS/ESA
MVS/XA
NetView
OS/400
OS/390
OS/2
PowerPC
QMF
RACF
RISC System/6000
SAA
SP
SQL/DS
SQL/400
S/370
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WIN-OS/2

Trademarks of Other Companies

The following terms are trademarks or registered trademarks of the companies listed:

C-bus is a trademark of Corollary, Inc.

HP-UX is a trademark of Hewlett-Packard.

Java and HotJava are trademarks of Sun Microsystems, Inc.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Solaris is a trademark of Sun Microsystems, Inc.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, or service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Index

D

F

I

J

L

N

O

P

Q

R

S

T

U

V

Contacting IBM

This section lists ways you can get more information from IBM.

If you have a technical problem, please take the time to review and carry out the actions suggested by the *Troubleshooting Guide* before contacting DB2 Customer Support. Depending on the nature of your problem or concern, this guide will suggest information you can gather to help us to serve you better.

For information or to order any of the DB2 Universal Database products contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

Telephone

If you live in the U.S.A., call one of the following numbers:

- 1-800-237-5511 to learn about available service options.
- 1-800-IBM-CALL (1-800-426-2255) or 1-800-3IBM-OS2 (1-800-342-6672) to order products or get general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, see Appendix A of the IBM Software Support Handbook. You can access this document by selecting the "Roadmap to IBM Support" item at: http://www.ibm.com/support/.

Note that in some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.

World Wide Web
http://www.software.ibm.com/data/
http://www.software.ibm.com/data/db2/library/

The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more. The DB2 Product and Service Technical Library provides access to frequently asked questions, fixes, books, and up-to-date DB2 technical information. (Note that this information may be in English only.)

Anonymous FTP Sites
ftp.software.ibm.com

Log on as anonymous. In the directory /ps/products/db2, you can find demos, fixes, information, and tools concerning DB2 and many related products.

Internet Newsgroups
comp.databases.ibm-db2, bit.listserv.db2-l

These newsgroups are available for users to discuss their experiences with DB2 products.

CompuServe
GO IBMDB2 to access the IBM DB2 Family forums

All DB2 products are supported through these forums.

To find out about the IBM Professional Certification Program for DB2 Universal Database, go to http://www.software.ibm.com/data/db2/db2tech/db2cert.html

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

S10J-8154-00