DataHub for UNIX Operating Systems
User′s Guide
Release 2

Document Number SC26-8668-01

Note!

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

Second Edition (May 1996)

This edition replaces and makes obsolete the previous edition, SC26-8668-0. The technical changes for this edition are summarized in “Summary of Changes” on page xviii but because the changes are extensive, they are not indicated by vertical bars in the left margin.

This edition applies to Release 2 of DataHub for UNIX Operating Systems, 5765-530, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for readers′ comments appears at the end of this publication. If the form has been removed, address your comments to:

IBM Corporation, Department J58
P. O. Box 49023
San Jose, CA 95161-9023
United States of America

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

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Subject to IBM′s valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products except those expressly designated by IBM, are the responsibility of the user.

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:

IBM Directory of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information which has been exchanged, should contact:

IBM Corporation, Department J01
P.O. Box 49023
San Jose, CA 95161-9023.

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

Trademarks

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

AIX	DataHub
AIX/6000	DB2
AIXwindows	DB2/6000
DATABASE 2	IBM
	NetView
	RISC System/6000

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

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

About This Book

This book explains how to use DataHub for UNIX Operating Systems' graphical user interface to manage your database and system environments from a single control point and also how to customize DataHub for your management requirements. In addition, this book tells you how to use DataHub's sophisticated rule-based system to automate your operations based on your company's business policies.

Who Should Read This Book

This book is written for database administrators, system administrators, and users who manage UNIX databases and systems. Readers of this book should have:

- Some experience with the operating systems that they intend to manage with DataHub (AIX, HP-UX**, or Sun Solaris**).
- Knowledge of the relational database management system (RDBMS) that they intend to manage with DataHub (DB2 Version 1, DB2 Version 2, INGRES**, ORACLE**, or Sybase**)

Terminology

Table 1 on page xiii provides a brief list of common DataHub terms and shortened names of several RDBMS types.

You can find definitions of DataHub terms in the text of this book. When the book introduces new terms, it italicizes and defines them. You can also find definitions in "Terms and Abbreviations" on page 367, which contains definitions of DataHub terms and technical terms that pertain to DataHub.

Table 1. DataHub Basic Terminology List

Term	Definition
AIX	Advanced Interactive Executive
DataHub	The DataHub for UNIX Operating Systems product
DB2	Refers collectively to: IBM DATABASE 2 for AIX/6000, Version 1 IBM DATABASE 2 for AIX, Version 2 Release 1 IBM DATABASE 2 for HP-UX, Version 2 Release 1 IBM DATABASE 2 for the Solaris Operating Environment, Version 2 Release 1
DB2 for AIX	IBM DATABASE 2 for AIX, Version 2 Release 1
DB2 for HP-UX	IBM DATABASE 2 for HP-UX, Version 2 Release 1
DB2 for Solaris	IBM DATABASE 2 for the Solaris Operating Environment, Version 2 Release 1
DB2 Version 1	IBM DATABASE 2 for AIX/6000, Version 1
DB2 Version 2	Refers collectively to: IBM DATABASE 2 for AIX, Version 2 Release 1 IBM DATABASE 2 for HP-UX, Version 2 Release 1 IBM DATABASE 2 for the Solaris Operating Environment, Version 2 Release 1
Host	A workstation operating system that DataHub can manage
HP	HP-UX
Managed database	A database that resides on a managed host and is managed by DataHub
Managed host	A host managed by DataHub
Mouse button 1	The left-hand button of a mouse when the mouse is configured for a right-handed user
Sun	Sun Solaris Operating Environment
Sybase database	A database under a SYBASE SQL Server**
Sybase server	SYBASE SQL Server

Conventions

This section identifies the typographic, special symbol, and command line entry conventions used in this book.

Typographic Conventions

This book uses the following typographical conventions:

Monospace font
: Text shown in monospace font should be typed exactly as shown; for example, `aixmh.tar`.

Bold monospace font
: Text shown in bold monospace font (commands and keywords) should be typed exactly as shown; for example, **lpstat**.

Italic font
: Words shown in italics are new terms with definitions specific to DataHub. These words are defined in the glossary.

Italic monospace font
: Text shown in italic monospace font represents a variable requiring substitution of an installation-specific value; for example, *archive_file*. Variables are used to represent strings of text that are dependent upon your particular configuration. If a chapter contains several variables, they are defined in the beginning of the chapter.

Symbol (Reverse Font) Conventions

Small black boxes surrounding text (reverse font) are used to identify steps for specific RDBMS or operating system types.

The black box symbols are:

Symbol	System
DB2	DB2 (all versions)
DB2v1	DB2 Version 1 only
DB2v2	DB2 Version 2 only
DB2AIX	DB2 for AIX only
DB2HP	DB2 for HP-UX only
DB2SUN	DB2 for Solaris only
Ing	INGRES
Ora	Oracle
Syb	Sybase (server or database)
AIX	IBM AIX
HP	HP-UX
Sun	Sun

Command Line Entries

This book shows sequences of command line entries that you must enter within the same step on separate lines. For example:

cd /u
mkdir *datahub_dir*

When a lengthy command line entry extends beyond the margin of the book, a backslash (\) is shown at the end of the line to indicate that the following line is a continuation of the current entry. Do not type the backslash continuation character as part of the command. For example:

export PATH=.:./bin:/usr/bin:/usr/local/bin:/usr/local/X11/bin:\
/usr/local/lpp/bin:$HOME/bin

How to Read the Syntax Diagrams

In this book, syntax is described using the following structure:

- Read the syntax diagrams from left to right and from top to bottom, following the path of the line. The following table shows the meaning of symbols which can appear at the beginnings or ends of syntax diagram lines.

Symbol	Indicates
———	the syntax diagram starts here
———	the syntax diagram is continued on the next line
———	the syntax diagram is continued from the previous line
———	the syntax diagram ends here

- Required items appear on the horizontal line (the main path).
- Optional items appear below the main path.

```
──STATEMENT──┬───────────────┬──────────────────────────────────
             └─optional–item─┘
```

- When you can choose from two or more items, the items appear vertically, in a stack. If you **must** include one of the items, one item of the stack appears on the main path. The default, if any, appears above the main path and is chosen by the compiler if you do not specify another choice.

```
             ┌─default–item──┐
──STATEMENT──┼─choice 1──────┼(1)──────────────────────────
             └─choice 2──────┘
```

Note:

[1] Because *choice 1* appears on the horizontal bar, one of the items must be included in the statement. If you don′t specify either *choice 1* or *choice 2*, the compiler implements the default for you.

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

```
──STATEMENT──┬──────────────────────┬──────────────────────
             ├─optional–choice 1────┤
             └─optional–choice 2────┘
```

- An arrow returning to the left above the main line is a *repeat arrow*, and it indicates an item that can be repeated.

```
             ┌──────────────────┐
──STATEMENT────repeatable–item──┴──────────────────────────
```

A repeat arrow above a stack indicates that you can choose more than one item from the list of stacked items, or repeat a single choice.

- If there is a comma in the path of the repeat arrow, you must use a comma to separate items in a series.

```
             ┌─,────────────────┐
──STATEMENT────repeatable–item──┴──────────────────────────
```

If the comma appears below the repeat arrow line instead of on the line as shown in the previous example, the comma is optional as a separator of items in a series.

- A syntax fragment is delimited in the main syntax diagram by a set of vertical lines. The corresponding meaning of the fragment begins with the name of the fragment followed by the syntax, which starts and ends with a vertical line.

```
──STATEMENT──┤ fragment ├──────────────────────────────────────
fragment:
├──syntax item──────────────────────────────────────────────────┤
```

- Keywords appear in uppercase (for example, STATEMENT). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, *item*). They represent user-supplied names or values.
- If punctuation marks, parentheses, arithmetic operators, or other symbols are shown, you must enter them as part of the syntax.

Using This Book Online

This book is available in two formats on the installation media:

DHUU1200.INF
: You can display this INF file with the Information Presentation Facility (IPF) book viewer (xview). DataHub provides xview. To install and use xview, refer to the *DataHub for UNIX Operating Systems Installation and Configuration Guide*.

DHUU1200.PS
: You can print this PostScript** file using a local PostScript printer.

Summary of Changes

This section lists product and document updates for Release 2.

Product Changes

Significant product additions include:

- Support for:
  - IBM DATABASE 2 for HP-UX, Version 2 Release 1
  - IBM DATABASE 2 for the Solaris Operating Environment, Version 2 Release 1
  - INGRES
- Grouping functions
- Enhanced filtering for objects displayed in the Object Tree window
- Find functions for locating objects in the Object Tree window
- Database discovery
- Enhanced watchdog functions
- ORACLE backup and recovery functions
- Migration support for current DataHub users

User′s Guide Changes

Significant book updates include:

- New sections for INGRES
- Documentation for grouping, filtering, find, database discovery, and watchdog functions

Installation and Configuration Guide Changes

Significant book updates include:

- Reorganized book structure
- Documentation for INGRES procedures and migration windows

Getting Started

Chapter 1. Getting Acquainted with DataHub

This section introduces you to DataHub by providing overviews of:

- The resources you can manage
- How you can manage these resources
- The management capabilities of DataHub
- The management control point

The Resources You Can Manage

You can use DataHub to manage

- Relational databases distributed on a network
- Hosts on which the databases reside
- DataHub itself

Managing databases and hosts encompasses administration and monitoring of the databases and operating systems. Examples of administrative tasks for database are creating changing, backing up, and recovering; for operating systems, examples of administrative tasks are displaying processes, file system usage, and statistics.

By monitoring your databases and operating systems, you can continuously check the health of your database and system resources. You can also take preventative or corrective action when a problem occurs. With DataHub′s monitoring capabilities, you can automate your operations based on your business policies. Monitoring your databases and operating systems with DataHub is covered in detail in Chapter 27, "Monitoring Your Databases and Operating Systems" on page 286.

You can manage any number of databases residing on any number of hosts. The database types include DB2 Version 1, DB2 Version 2, INGRES, ORACLE, and Sybase. The host operating systems include AIX for the RISC System/6000 (IBM), HP-UX (Hewlett-Packard), and Sun Solaris (Sun Microsystems).

In addition to managing databases and their hosts, you will need to manage DataHub itself and the DataHub control database. This includes such tasks as defining hosts and databases to be managed to the control database, authorizing individuals and groups to use the DataHub application, and adding user-defined objects and actions, as you want, to DataHub.

FYI: The term, 'DataHub application', refers to DataHub as typically viewed by the user—as a tool for system and database administration and monitoring.

Managing Your Resources with DataHub

When you install DataHub, you create a network consisting of DataHub and the hosts and databases it supports. The key components of this network are:

- The DataHub *control point*. This is a programmable workstation from which you invoke DataHub functions.
- The hosts on which the database systems you administer reside.

From the control point, you can make requests that DataHub performs immediately. For example, from the control point, you can display all of the tables that exist on a database or create a table. This type of activity (displaying objects or creating an object) involves interacting with the DataHub user interface directly and is referred to as *online*, or *foreground*, activity.

Additionally, DataHub provides an *offline*, or *background*, system. This system monitors and automatically controls host or RDBMS operations, according to your business policies, whether or not you are working at the control point. For further information, see Chapter 27, "Monitoring Your Databases and Operating Systems" on page 286.

Figure 1 on page 4 shows one of the possible configurations of a DataHub network, consisting of a control point and a few of the types of hosts and databases that DataHub currently supports.

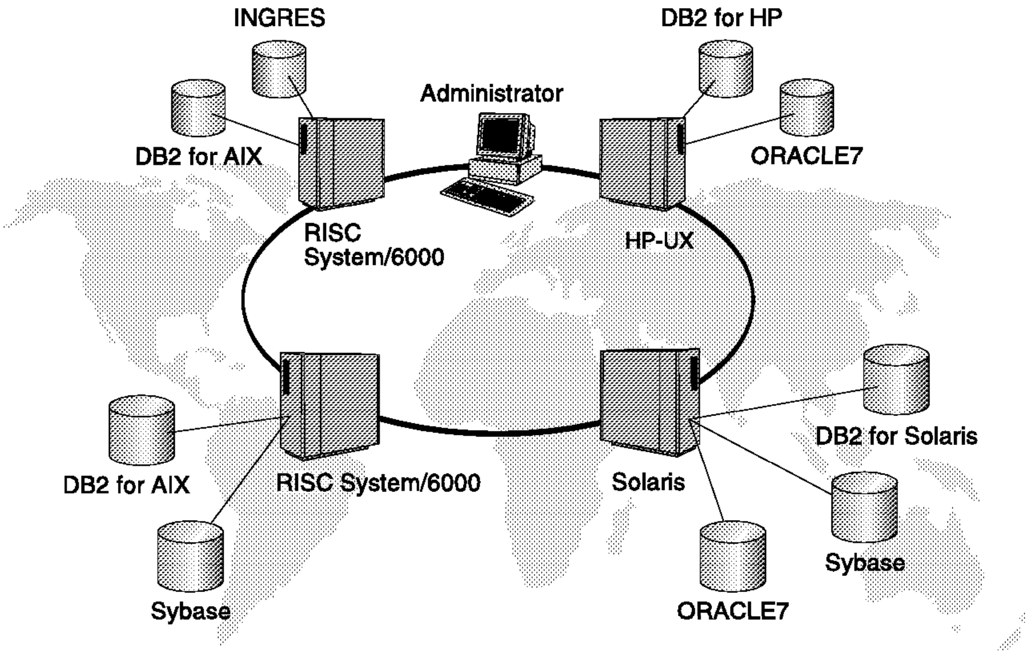


Figure 1. A DataHub Network

The Management Capabilities of DataHub

DataHub:

- Manages and monitors your distributed, multi-vendor databases as a centralized database management system
- Oversees heterogeneous UNIX operating systems on a network
- Automates operations based on your business policies
- Directly manipulates system and database objects using a graphical user interface (GUI).
- Exploits functions specific to each type of relational database system, while keeping low-level operations transparent to the user
- Provides robust and comprehensive database management functions such as table and view creation and copying.
- Alerts administrators when predefined thresholds are reached to prevent problems from occurring.

- Provides a powerful rule-based language and parser for specifying actions to be taken based on the monitoring results of one or more resources or systems.
- Supports standards such as SQL, TCP/IP, and SNMP.

You can also extend DataHub to call other tools by creating user-defined actions.

The Management Control Point

This section introduces two basic parts of the control point:

- The *DataHub control database*. This database contains all the information needed by DataHub to manage and monitor your hosts and databases.
- The *DataHub for UNIX Operating Systems window*. This is your interface to the hosts and databases you manage. This section will explain how to use it.

The Control Database

The control database is the database that contains the DataHub control tables and views. These objects hold display information, information about managed hosts and managed databases, checks passed to watchdogs and values returned by the watchdogs, and a number of DataHub operational items.

The control database typically resides at the control point; however, it can reside on any DataHub managed host within the network.

During installation, you might have chosen to install the DataHub control tables and views in one of your existing databases. From here on, the term control database will refer to the database where you installed the DataHub control tables and views, regardless of what other data you might have in that database.

The DataHub for UNIX Operating Systems Window

Figure 2 on page 6 shows the layout of the DataHub for UNIX Operating Systems window.

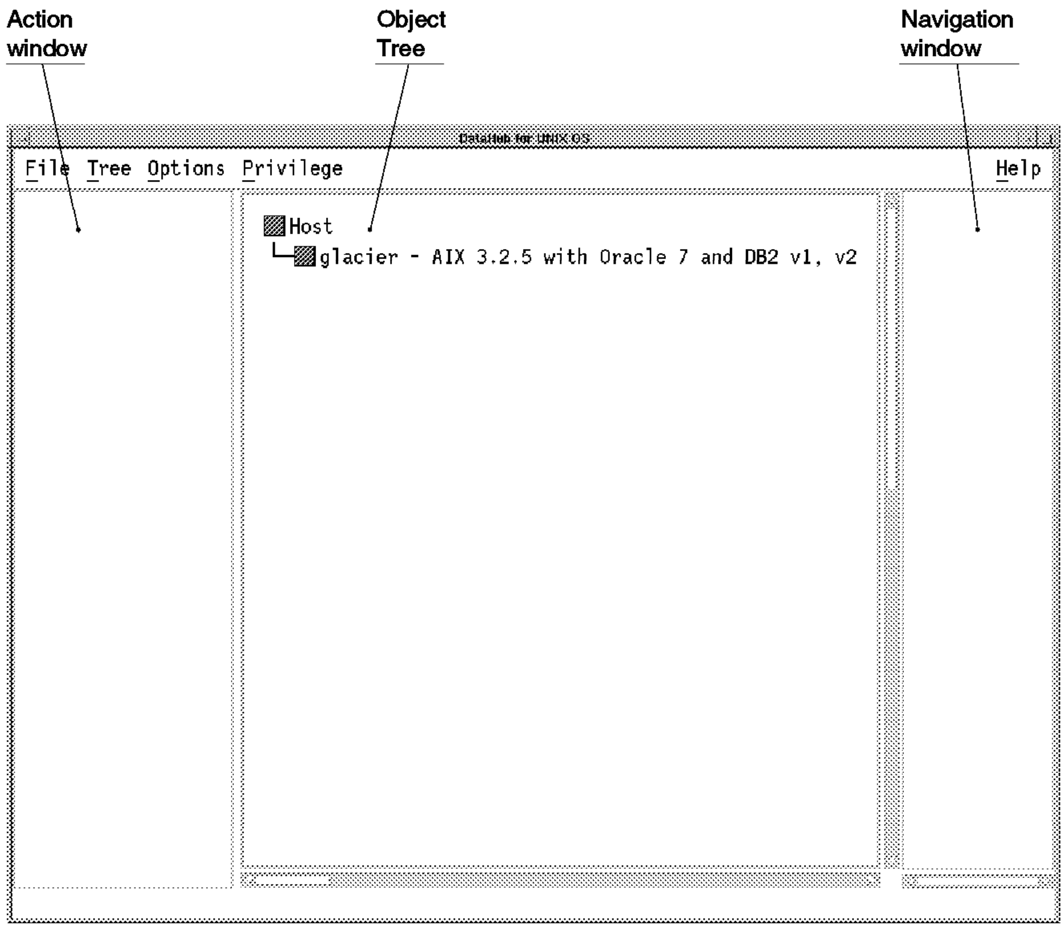


Figure 2. DataHub for UNIX Operating Systems Window

The DataHub for UNIX Operating Systems window has a menu bar at the top, a status message line, and three areas. Depending on the size at which you have your window, each of the three areas has scroll bars: horizontal and vertical. The appearance of the scroll bars is dynamic, depending on whether the window can display all of the information in a particular area. Use the vertical scroll bar to move up and down in an area, and use the horizontal scroll bar to move right and left.

The menu bar′s pull-down menus let you perform such tasks as controlling your view of the DataHub window, opening and saving files, managing access to DataHub, and displaying online help.

The status message line is in the lower left of the DataHub for UNIX Operating Systems window. This message line lets you know the status of

DataHub actions; this is useful to see whether or not an action has completed.

For example, if you create a DataHub application named ‘App1’ between two tables, when the creation process is finished the message:

Application ‘App1’ created.

appears in the status message line.

The center area is called the *Object Tree window*. The Object Tree window is used to display host objects and database objects and their relationships in the form of a hierarchical tree. You can select one or more object types or object occurrences from this object tree to perform systems management and database management functions.

The left area is called the *Action window*. It contains all the actions which can be performed on the objects you select in the object tree. Only actions applicable to the selected objects are shown. These actions vary from database to database because DataHub supports specific functions available for each type of database.

The right area is called the *Navigation window*. The Navigation window shows you the path of objects between the Host object type and the object that is currently selected in the Object Tree Window. You can use this window to “jump back” quickly to another position in the object tree. The Navigation window is useful when the object tree grows through successive object display requests.

The Menu Bar

The menu bar has the following pull-down menus

- **File**
- **Tree**
- **Options**
- **Administer**
- **Help**

File Pull-Down Menu

The **File** pull-down menu has choices that let you manage connections to databases and hosts, open and save object tree files, and exit DataHub itself.

For information on connections, see “Working with Connections Established by DataHub” on page 10; for information on opening and saving files, see “Saving and Restoring Object Trees” on page 20.

Tree Pull-Down Menu

The **Tree** pull-down menu has choices that let you manipulate the object tree (see "Tips on Manipulating the Object Tree" on page 15).

Options Pull-Down Menu

The **Options** pull-down menu has choices that let you

- Write SQL statements
- Display the DataHub status window
- Find an object in the object tree using a string
- Log the contents of the DataHub display window to a file

For information on using the

- **Query** choice, see Appendix A, "Working with Queries in the DataHub Query Window" on page 340
- **Display** choice, see "Viewing SQL Statements Generated by DataHub" on page 23
- **Find** choice, see "Finding Objects in the Object Tree" on page 24
- **Start Logging** choice, see Appendix G, "Logging the DataHub Display Window" on page 363

Administer Pull-Down Menu

The **Administer** pull-down menu has choices that let you:

- Manage access to DataHub itself
- View and delete watchdog action logs
- Delete all watchdog check values
- Start or stop the monitoring system on a managed host

For information, see

- Chapter 4, "Managing Administrator Authorizations and Privileges" on page 35, about managing access to DataHub itself
- "Viewing and Deleting the Action Log" on page 313, about managing watchdog action logs
- "Working With Stored Watchdog Check Values" on page 304, about managing watchdog check values
- "Starting and Stopping Watchdogs" on page 295, about starting and stopping the monitoring system

Help Pull-Down Menu

The **Help** pull-down menu has choices that let you use online help and begin an online tutorial. For instructions on how to use DataHub′s online help, see “Online Help” on page 9.

Selecting Objects in DataHub Windows

When you are working in DataHub windows, there are a number of ways you can select objects.

- Click once on an object with mouse button 1 to select that single object.
- Click on an object with mouse button 1, hold down the button, and drag to select objects next to each other on the screen.
- Hold down the Ctrl key and click on a number of objects to select many objects located in different parts of the screen.

Gray Fields in DataHub Windows

When you are specifying parameters in a window, some parameters have a “gray” appearance. You cannot specify these fields that have this *graying*.

If the field has a radio button or check box, there might be a colored box in the radio button or check box. However, the button or box is non-functional because the field has graying.

Sometimes whether or not a field has graying depends on what other parameters are specified in the window. If you specify the other parameters, the gray field converts to a regular parameter that you can specify.

Online Help

For information on various aspects of DataHub, you can consult the online help. The help is intended to explain what you see on the screen. Use the online help to get immediate assistance while you are involved in a task.

To access the online help

- For the full range of help information, including an online tutorial:
  1. At the top of the DataHub window, click on the **Help** choice. A menu appears.
  2. Select the choice you are interested in from this menu.

- For information about an object type on the object tree:

  1 Select the object type.
  2 Press the F1 key.[1]

- For information about how to proceed after you select an object on the object tree:

  1 Select the object.
  2 Press the F1 key.[1]

- For information about an action invoked by a push button on the DataHub window:

  1 Position the cursor over the push button.
  2 Press and hold down the mouse button.
  3 Move the cursor away from the push button.
  4 Release the mouse button.
  5 Press the F1 key.[1]

- For information about a window's purpose and fields, you can do one of the following actions:

  – Click on the window's **Help** push button.
  – Press the F1 key.[1]

- To close a help window, you can do one of the following actions:

  – Click on the system menu icon in the upper-left corner of the window and select **Close** from the menu that appears.
  – Press the **Esc** key. You might need to press it several times to close the help window.

Working with Connections Established by DataHub

To use the control point to display objects that exist on a database or Sybase server, you must have a connection established to that database or Sybase server. If you try to display objects and a connection is not already established, a login window opens so you can enter a user name and password to establish a connection to that database or Sybase server.

[1]

Exception: If you are invoking DataHub from a programmable workstation supported by a Sun Solaris** host, press your keyboard's Help key instead of F1.

After you log on to a database or Sybase server, a database connection is established and you can display its objects. You do not need to log on again unless the connection is closed to that particular database or Sybase server.

When you first start up DataHub and begin your session at the control point, you log on to the control database. Then, during your session in DataHub, you can log on to multiple databases or Sybase servers at the same time.

Working with Database Connections

The **DB Connections** choice of the **File** pull-down menu lets you see all the connections to individual databases and Sybase servers for the current DataHub session.
You can also:

- Open new connections to databases
- Close existing database connections
- Work with finished jobs
- Work with asynchronous jobs

You perform any of these tasks from the Connections window. To open the Connection window, select the **DB Connections** choice. The Connections window displays a list of current database connections, the names of the users and the databases or Sybase servers they are connected to, and the names of the hosts.

Opening and Closing Database Connections

You might want to open and close particular database connections if your current connections do not allow you to perform certain operations.

For example, suppose that your current connection to a database is under a user name that does not have authorizations to a set of tables you want to alter, and you know another user name and password combination that does have the authorizations. You could close the connection that exists under the first user name and then open a new connection under the other user name.

The following procedures are done from the Connections window.

To open a new database connection:

1 Click on the **Open** push button. The Managed Database Connect window opens.

2 Enter the database name or select it by using the **List** push button.

3 Enter the user name and password and click on **Connect**.

Ora If you are an ORACLE database user and have external identification, you can select the **Identification External** check box, enter your operating system user name and password, and then click on **Connect**.

To close an existing database connection:

Select the database or Sybase server name and click on the **Close** push button. The connection is closed and is removed from the display.

Attention: Do not close the connection to the control database.

Working with Finished and Asynchronous Jobs

For a particular asynchronous process, you can:

- Display information such as the command used to start the process
- Terminate the process
- Delete jobs that finished
- Set the status for the message pop-up windows

You can perform some processes in DataHub asynchronously by specifying the **Async. Exec.** option when you specify parameters for the process (for example, backup of a database).

To work with finished and asynchronous jobs:

Click on the **Async Jobs** push button. A window opens that displays any asynchronous jobs running on connected databases or Sybase servers.

To set the asynchronous job pop-up window:

1. Click on the **Set Message** push button. The Set Message window opens.
2. Specify a message pop-up by clicking on the Message Popup radio button and then clicking on the **OK** push button. A window will pop-up and display a message when the job is finished.

To display information on a particular asynchronous process:

1. Select the process and click on the **Information** push button. A message window opens with information about the process.
2. Click on the **OK** push button to close the window.

If you terminate an asynchronous job prior to the completion of that job, DataHub terminates it and changes the active status to "Ready". You can still look at the output of the job.

To terminate an asynchronous process prior to its completion:

Select the process and click on the **Kill Process** push button.

DataHub terminates the process and changes the active status to "Ready". You can get information about the job by selecting the process and clicking on the **Information** push button.

To delete finished processes from the window:

Select the **Delete Done Jobs** push button. Finished jobs are removed from the window.

For more information on this window, see online help.

Working with DBM Connections

Before you can perform certain commands on an RDBMS, you need the right operating system connections. Actions such as starting or shutting down a database require a connection that uses an operating system user id and password combination; this combination must have the necessary authority (as determined by the RDBMS type) to perform the action.

For the RDBMS types that DataHub supports, typical actions that need special authority are:

DB2 Commands that require SYSADM authority

Ing Operations that require the INGRES System Administrator permission

Ora System-privileged SQL and DBA commands that usually require the INTERNAL connection

Syb Operations that require the System Administrator or System Security role

For more information on these types of actions, see the database administrator′s guide for the particular RDBMS.

Within DataHub, you can open and close these types of connections by using the **DBM Connections** choice of the **File** menu. You might need to open and close particular privileged connections if your current connections do not have the proper authority to perform the operation you want.

To close a special privileged connection:

1. Select the **File** menu and select the **DBM Connections** choice.

   The DBM Connections window opens. It shows any existing privileged connection you might have.

2. Select the connection you want to close and select **Close** push button.

   The connection is closed and is removed from the window.

3. Click on the **OK** push button to close the window.

To open a special privileged connection:

1. Select the **File** menu and select the **DBM Connections** choice.

   A window opens. It shows any existing privileged connection you might have.

2. Select **Open** to open a connection. The Operating System Login for Database Operation window opens.

3. Enter the necessary information. See the online help for an explanation of the parameters.

4. Click on the **OK** push button to complete the login process.

   The new connection is displayed in the DBM Connections window.

Working with Operating System Connections

If you perform an action that requires you to log in directly to an operating system, you open a connection to that operating system. Such actions include testing a watchdog check, viewing or delete the watchdog action log, or invoking a user-defined action that needs to connect to an operating system (for example, to perform a command on a managed host). You can close these direct connections as needed.

To close a specific operating system connection:

1. Select the **File** menu and select the **Host Connections** choice. A window opens and shows any existing direct operating system connections.

2. Select the connection you want to close and select the **Close** push button.

   The connection is removed.

3. Select the **OK** push button to close the window.

Tips on Manipulating the Object Tree

You can manipulate the object tree in several ways:

- Displaying objects of a particular type
- Removing displayed objects from it temporarily (called *collapsing* the object tree) and restoring them at a later time (called *expanding* the object tree).
- Removing displayed objects without being able to restore them by expanding the object tree (called *hiding* the object tree).
- Refreshing the object tree
- Navigating through the object tree

Displaying Objects on the Object Tree

After you define a host to the control database, it appears on the object tree automatically. When it appears on the object tree, you can display objects associated with it on the object tree; after you display these objects on the object tree, you can display additional objects associated with *them*; and so on.

For example, suppose you have defined a host to the control database. When you open the DataHub window, this host will appear on it. At that time, you can also display objects associated with this host—databases, users, and so on.

To display objects associated with an object already on the object tree:

1 Either:

- Select the object that is already on the object tree and click on the **Display** push button.
- Double-click on the object.

The Display Types window opens. It lists the object types associated with the object you selected or double-clicked on.

Exception: If DataHub associates only one object type with the selected object, no window will open. Instead, the associated object type and its occurrences will appear in the tree. For example, User is the only object type that DataHub associates with Sybase groups. Suppose that the tree lists the Sybase groups that use a given Sybase database. If you double-click on one of these groups, the User object type and any users belonging to the group will appear in the tree.

2 From the Display Types window, select the object type whose occurrences you want added to the object tree; then click on the **OK** push button.

3 The object type you selected and any occurrences of that type are displayed on the object tree.

> **Note:**
>
> If DataHub does not have a current connection to the database or Sybase server where the objects reside, the Managed Database Connect window opens following the Display Types window. For information on connections, see "Working with Connections Established by DataHub" on page 10.

Filtering Objects

For some object types, a Filter window opens; in this window, you can specify particular values for certain characteristics of the object occurrences. Then, DataHub will only display those objects that have those values. There are two types of Filter windows: with the first type, you can filter on the owner and name of an object; with the second type, you can filter only on the name of an object.

Filtering by Owner and Name

The occurrences of object types like tables are qualified by both an owner and a name (the object type depends on the database type). When you select to display these object types, a filter window opens that has an **Owner** field and a **Name** field. The SQL wildcard character (%) appears initially in both fields.

The Filter window fields (Owner and Name) are case-sensitive.

To see objects that have a particular owner:

- In the **Owner** field, if you want to specify the exact owner name, delete the %, enter the owner name you want, and select the **OK** push button.

  If you want to use the wildcard character, enter the characters you want on either side of the % and select the **OK** push button.

To see only those objects with particular characters in the last component of their fully-qualified name,

- In the **Name** field, if you want to specify the exact name, delete the %, enter the name you want, and select the **OK** push button.

If you want to use the wildcard character, enter the characters you want on either side of the % and select the **OK** push button.

For example, to see all tables that end in "TEST", you could enter TEST to the right of the % sign and select the **OK** push button. The tables are then displayed on the object tree.

To see all available objects (regardless of owner or name), just click on the **OK** push button.

You can also use delimited identifiers in the filter window. For information on delimited identifiers, see Appendix C, "Using Delimited Identifiers and Quoted Names" on page 345.

Filtering by Name Only

If an object type has more than 25 object occurrences, a filter window opens in which you can filter on the name of an object occurrence. To filter on the name, follow the steps described on page 16. The Name field is case-sensitive.

You can use percent signs (%) as wild card characters. With wildcards, you can:

- Display names that include the characters that the wild card stands for
- Display names limited to the characters next to the wild card

For example, suppose that the last component of the fully-qualified names of several tables is either a string that includes both "JOB" and "TEST" and other characters, or a string limited to "TEST". If you have %TEST% in the **Name** field, you might see such names as:

- "TUCHMAN.NEWTEST"
- "TUCHMAN.JOB1TEST"
- "TUCHMAN.JOB2TEST"
- "TUCHMAN.TEST"
- "TUCKER.MYTEST"
- "TUCKER.TEST"

If you enter JOB%TEST%, you might see such names as:

- "TUCHMAN.JOB1TEST"
- "TUCHMAN.JOB2TEST"

Tip: You have just seen how to display objects of one type (such as Database) that are associated with a single object of another type (such as Host). You can also:

- Display objects of *more than one* type that are associated with a single object of another type. You do this from the Display Types window. Select the types of objects you want to see and click on the **OK** push button.

  For example, suppose you want to display the databases and users associated with a host called "onehost". You could start by double-clicking on "onehost". The Display Types window would then open. From this window, you would select **Database**, select **OS User**, and click on the **OK** push button.

  If you decide not to display the objects of a type you have already selected, deselect the type before clicking on **OK**.

- Display objects of one type that are associated with *more than one* specific object of another type. To do this, first select one of the specific objects (be sure not to double-click). Then:

  - If the next specific object you want is right above or below the object you just selected, hold down the mouse button and move the mouse cursor to this next object.

  - If other objects lie between the next specific object and the one you just selected, move the mouse cursor to the next specific object, hold down the control key, and click on mouse button 1.

  For example, suppose you want to display the databases associated with the hosts called "onehost" and "anotherhost". You could start by selecting "onehost" from the object tree. If "anotherhost" is right above "onehost", hold down the mouse button and move the mouse cursor to "anotherhost". Otherwise, if there are other objects displayed between "onehost" and "anotherhost", move the mouse cursor to "anotherhost", press the control key, hold the key down, and click on mouse button 1.

Caution: In the foregoing, you saw several series of instructions that ended with clicking on a window's **OK** push button. Clicking on **OK** causes DataHub to start processing the request entered from the window. Therefore, if you click on a push button from a window to request DataHub to perform an action—for example, if you click on **Add** or **Open**—be sure to also click on **OK.**

Removing Displayed Objects from the Object Tree

You can remove displayed objects from the object tree by

- Collapsing objects
- Hiding objects
- Dropping object occurrences

When you drop object occurrences, they are deleted from the database or host system. The DataHub **Drop** action for a particular object type is discussed in the section of this book that describes that object type.

When you collapse or hide objects, they no longer appear in the object tree; however, they still exist in the database or host system.

To collapse a set of objects in the object tree:

1 Select the object you want to collapse.

2 Select the **Tree** menu and select Collapse.

All of the objects branching out from under the object disappear and a plus sign (+) appears next to the object icon.

Displaying Collapsed Objects

When you display a collapsed object by expanding, all of the objects under it are displayed again on the object tree. However, the database is not queried; if an object occurrence that would otherwise be displayed under the collapsed object is added to the database while the object was collapsed in the object tree, it does not appear in the object tree if you expand the collapsed object. To display such added object occurrence, you must instead refresh the object tree.

To display a set of collapsed objects in the object tree

1 Select the object you want to display.

2 Select the **Tree** menu and select Expand.

All of the objects under that object are displayed on the object tree.

Hiding Objects in the Object Tree

You can also remove an object from the object tree by hiding it. The object still resides in the database. When you hide an object, you cannot restore it by simply expanding the object tree. To restore a hidden object occurrence, you need to refresh its object type (see "Displaying Hidden Objects in the Object Tree" on page 20); to restore a hidden object type, you need to display it again.

To hide an object occurrence

1 Select one or more object occurrences you want to remove from the object tree.

2 Select the **Tree** menu and select Hide.

The selected object occurrences disappear from the object tree.

To hide an object type

1 Select one or more object types you want to remove from the object tree.

2 Select the **Tree** menu and select Hide.

The selected object types and any object occurrences under those object types are removed from the object tree.

Displaying Hidden Objects in the Object Tree

Use the DataHub **Refresh** action to display hidden objects.

To refresh hidden object occurrences

1 Select the object type to which the occurrences belong.

2 Select the **Refresh** push button.

Any hidden object occurrences under that type are restored to the object tree.

To display a hidden object type, follow the steps in “Displaying Objects on the Object Tree” on page 15.

Saving and Restoring Object Trees

From session to session, you might want to always display the same objects in the object tree. For example, you might want to always display all the databases residing on a particular host. In DataHub, you can save configurations of the object tree to files and then use them in future sessions.

To save the current object tree configuration to a file

1 From the **File** menu, select **Save Tree As**. The Select File window opens.

For information on the fields in this window, see online help.

The Selection field initially shows the HOME directory for the operating system user who started this DataHub session. You specify the file name for the save operation in the Selection field.

For example, to save the current object tree in the file named first.tree located in the /u/mydir directory, type

/u/mydir/first.tree

in the Selection field.

2 Click on **Save**.

To use a saved object tree

1 From the **File** menu, select **Open**. The Select File window opens.

For information on the fields in this window, see online help.

The Selection field initially shows the HOME directory for the operating system user who started this DataHub session. You specify the file that has the object tree in the Selection field.

For example, to open the object tree that was saved in the file first.tree located in the /u/mydir directory, type

/u/mydir/first.tree

in the **Selection** field.

2 Click on **Open**. The new object tree is displayed.

Once you open a file, the default file name is the file you opened. Then, you can use **Save Tree** to save the current version of the object tree.

For more information on how to use the Select File window, see the online help.

To return the object tree to the configuration it has on startup, select **New Tree** from the **Tree** menu.

Refreshing the Object Tree

When you select an object type in the object tree, a **Refresh** push button appears in the Action window.
You click on this push button to:

- Restore hidden occurrences of the selected object type.
- Supply the tree with new occurrences of the selected object type from the database catalog tables.
- Remove occurrences of this object type from the object tree that might have been deleted by another user after you first displayed the object type.

In many cases, DataHub does this automatically. In other cases, you need to resort to the **Refresh** action.

For example, suppose a DB2 Version 2 user named JONES has the tables named:

```
1992TESTS
1993TESTS
1994TESTS
EMPTEST
DEPTTEST
DIVTEST
```

and you want to display on the object tree the tables with names that fit the filter `%TESTS%`. You double-click on the database occurrence, select Table from the list of display types, and enter `JONES` in the Owner field `%TESTS%` in the Name field of the Filter window. The following tables will be displayed on the object tree:

```
JONES.1992TESTS
JONES.1993TESTS
JONES.1994TESTS
```

Next, you want to display all of the tables belonging to JONES that have TEST in the name of the table. Suppose also that JONES has created a new table named 1995TESTS and deleted the DIVTEST table after you had first displayed the tables above. You select the Table object type and click on the **Refresh** push button. In the Filter window, enter `JONES` in the Owner field and `%TEST%` in the Name field.

The object tree refreshes and the following tables are displayed on the object tree:

```
1992TESTS
1993TESTS
1994TESTS
1995TESTS
EMPTEST
DEPTTEST
```

The **Refresh** action resulted in displaying the newest table (“1995TESTS”) as well as the rest of the tables that remain that fit your filter.

Navigating the Object Tree with the Navigation Window

You can navigate through the object tree in two ways:

- Use the right-hand scroll bar on the Object Tree window to scroll the window up and down until you have found the object you want. Select the object.

- Use the Navigation window

To use the Navigation Window

Either:

- Select the object to which you want to jump and hit the Enter key on your keyboard, or
- Double-click on the object to which you want to jump

The object appears in the object tree window and is selected.

> **FYI:** You can only use the Navigation window to navigate to an object that has a position on the object tree within the path from the root object type to the object that is currently selected.

Viewing SQL Statements Generated by DataHub

You can view the SQL statements generated by DataHub in the DataHub Display window. You might find this window helpful after you have performed a DataHub action and you want to check on the status of SQL statements submitted by DataHub.

To display the DataHub Display window

Select the **Options** pull-down menu and select **Display**. A window opens and displays the SQL commands issued from the beginning of the current DataHub session or the last time you cleared the history.

Use the scroll bar to scroll through the history of SQL commands.

You can clear the history by clicking on the **Clear** push button.

To close the display window, click on the **OK** push button.

> **FYI:** DataHub will show the key SQL statements that it issues to the underlying RDBMSs.

You can log the contents of the DataHub display window to a log file. See Appendix G, “Logging the DataHub Display Window” on page 363 for more information.

Finding Objects in the Object Tree

Use the **Find** menu choice of the **Options** pull-down menu to search for an object occurrence in the object tree.

The search proceeds downward through the object tree proceeding from the object that is currently selected. If the object occurrence is located above the currently selected object, the search wraps to the beginning of the object tree and continues downward.

For example, suppose you have the Group User object type displayed with all of group users belonging to three different database groups, and you have a group user named TESTER2 in all three database groups (see Figure 3 on page 25). The currently selected object is the Group User object type icon under the third database group.

If you search on the string TESTER2, the group user TESTER2 in the third database group is selected. Then, if you again search on the string TESTER2, the search wraps to the top of the object tree and the next TESTER2 object occurrence (in the first database group) is selected.

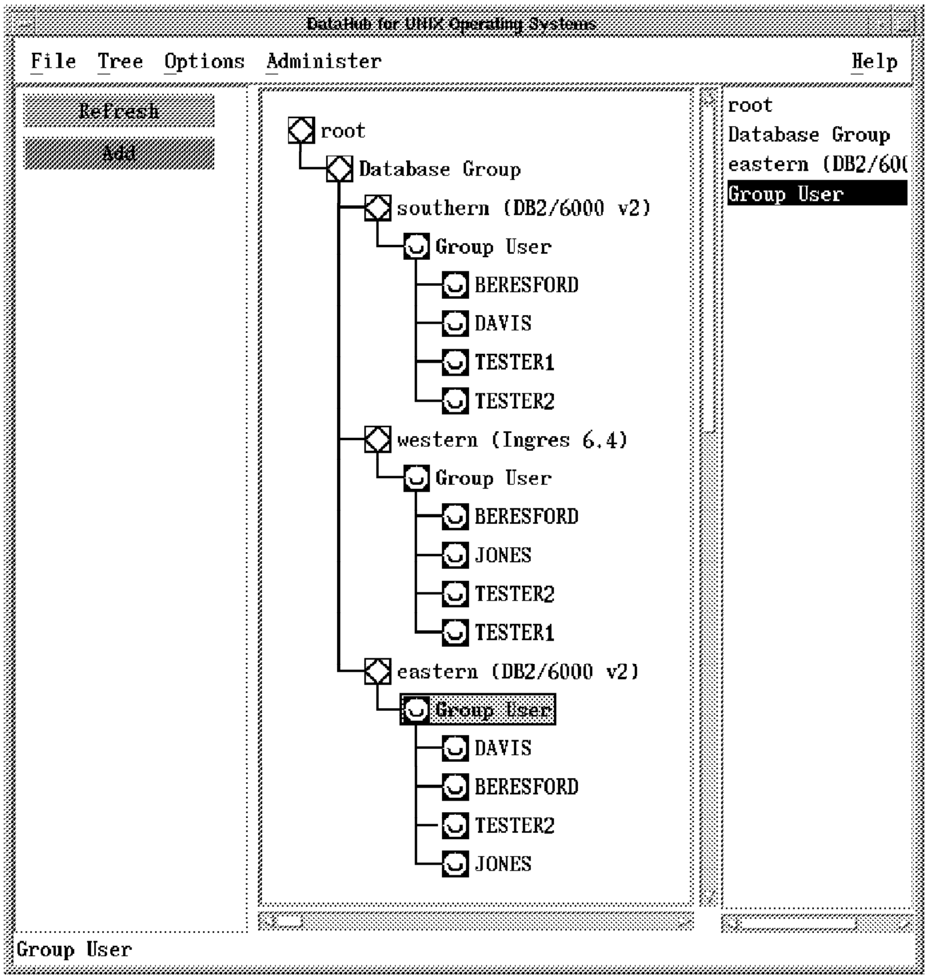


Figure 3. Example of using the Find menu choice

You can use the standard UNIX operating system wildcards within your search string. For example, to search for all object occurrences that have the string TESTER in their name, enter *TESTER* in the Search String field.

The label of the object occurrence as it appears in the object tree must exactly match the search string for the search to be successful; the search is case sensitive. If the object occurrence label is qualified by an owner name or has an attached description, the search will fail unless you use wildcards.

The Search String field is case-sensitive.

To search for an object in the object tree

1 Select the **Options** pull-down menu and select **Find**. The Find Object window opens. For information on this window, see online help.

2 Enter a string to search on in the **Search String** field. If you are not sure of the exact name, it is recommended that you use wildcard characters (for example, enter asterisks on either side of your string).

3 Select the **OK** push button.

   The Find Object window closes and the first object occurrence encountered that has that string is selected in the object tree.

Chapter 2. Starting Up and Exiting DataHub Sessions

Before you can start DataHub,

- The LANG environment variable must be set to a valid language.

  How you set the LANG variable depends on the UNIX environment you are using. See the *DataHub for UNIX Operating Systems Installation and Configuration Guide* or your system administrator for more information.

- The system must be able to find the datahub file.

  This can be done in a variety of ways. You can change to the directory containing the datahub file, set your PATH environment variable so it includes the directory containing the file, or set up a symbolic link from /usr/local/bin to the DataHub gui directory. See your system administrator for more information.

To invoke DataHub

1 Open a window on your workstation. AIXwindows/6000 or X Windows and OSF/Motif must be active.

2 Type datahub at the command line and press the **Enter** key. The Control Database Connect window opens.

3 Enter the user name and password you are using to log into the control database.

Ora If you are an ORACLE database user and have external identification, you can click on the Identification External check box and enter your operating system user name and password.

4 Click on the **Connect** push button.

At this point, the DataHub main window opens. The object tree shows the root object type, the Host object type, and any hosts you have defined to DataHub. To add a host, follow the procedure described in "Adding Host Definitions and Changing Host Descriptions" on page 66. After you have added a host, you can display other types of objects residing on that host by following the procedure described in "Displaying Objects on the Object Tree" on page 15.

Tip: To use DataHub, you need:

- An operating system login with the authority to execute the DataHub program on the system where you are running the DataHub control point
- Ability to (at least) select the control tables and views in the control database
- Membership in at least one of the DataHub administration groups that are authorized to use DataHub

For more information on granting the ability to use DataHub, see "To add or remove users to or from a DataHub admin group" on page 47.

After starting DataHub, if you select an object on the object tree and no push buttons appear in the Action window, it means that you do not yet belong to a DataHub administration group.

To exit DataHub

1 Click on the **File** pull-down menu and select the **Exit** choice. A question box opens asking for confirmation.

2 Click on the **Yes** push button to exit DataHub.

Setting Up the DataHub Management Environment

Chapter 3. Adding Your Existing Hosts, Sybase Servers, and Databases

When you have installed DataHub for the first time and you start it right after installation, you will see on the object tree the host that has the control point; however, no other hosts will be displayed on the object tree even though you might also have remote managed hosts on which you installed DataHub drivers (see the *DataHub for UNIX Operating Systems Installation and Configuration Guide* for more information). During installation, you chose a database in which to store the DataHub control tables and views; we call this database your control database.

If your control database is a DB2 for AIX, INGRES, or ORACLE database and you display all databases on the control point host, you′ll only see the control database displayed on the object tree. If your control database is a Sybase database and you display all Sybase servers on the control point host, you′ll only see the Sybase server that has the control database.

For instructions on how to display database occurrences of the different RDBMSs that reside on a host, see the chapter for that RDBMS type.

To manage your remote hosts (as well as the Sybase servers and databases that exist on your managed hosts) with DataHub, you have to define them to the control database. The definitions are then stored in the control database and DataHub uses these definitions to manage the hosts, Sybase servers, and databases.

Adding Definitions for Remote Hosts

To manage the remote hosts on which you have installed DataHub drivers, you need to specify host definitions and then add them to DataHub. The definitions are then stored in the control database and DataHub uses these definitions to manage the remote hosts.

In order to add a remote host definition, you need to know the network name for the host, the directory where DataHub is installed on that host, the type of host, and a host description.

The host name can be either the TCP/IP host name or a TCP/IP address. It must be the name by which the network knows this host.

If you do not know the host name, you can use the DataHub **Network** action to find the TCP/IP host name of the systems, both servers and clients, that are part of your network (see “Displaying Your Networked Systems” on page 33).

The host description will appear in the object tree next to the hostname for this host.

To add the definition of a remote host to the control database

1 Select the Host object type in the object tree.

2 Click on the **Add** push button. The New Managed Host window opens.

3 In the window, you must specify the network name for the host, the directory where DataHub is installed on that host, the type of host, and a host description.

If you want DataHub to automatically discover databases that exist on the host you are adding, select the **Database Discovery** check box. See “Discovering Existing Databases or Sybase Servers” for an explanation of DataHub′s database discovery process.

If you want to add out-of-the-box monitoring to this host, select the **Add Standard Monitoring** check box. See “Out-of-the-Box Monitoring” on page 293 for more information.

4 Click on the **OK** push button of the New Managed Host window.

If you selected **Database Discovery**, the Database Discovery window opens (see page “Discovering Existing Databases or Sybase Servers”). Otherwise, the added host object occurrence is displayed on the object tree.

Discovering Existing Databases or Sybase Servers

When you add a new host to be managed by DataHub, you can choose to have DataHub search for databases or Sybase servers existing on that host. DataHub gives you the option of adding them at the same time that you add the host (see “Adding Definitions for Remote Hosts” on page 30). You can also discover databases or Sybase servers on a managed host object occurrence by using the **Discovery** action button.

You tell DataHub to search for existing databases or Sybase servers in one of the following ways:

- To search on a host being added to DataHub, select the **Database Discovery** check box in the New Managed Host window
- To search on a host already managed by DataHub, select the host object occurrence and select the **Discovery** push button.

The Database Discovery window opens; this window has two parts: the Database Discovery Search Criteria and the Database Discovery Selection. First, you specify characteristics for the search. Once DataHub has found the databases fitting those characteristics, you choose which ones to add as managed databases.

To discover databases

1 In the Database Discovery Search Criteria section of the window, specify the database type you want DataHub to search for by clicking the **Add** push button for the Database Type parameter and selecting one or more from the list.

For detailed information on the parameters in the Database Discovery window, see the online help.

For a **Default** search type, DataHub searches the default local file system only. For a **Full Filesystem** search type, DataHub searches all NFS mounted file systems. A full file system search can be slow, depending on your system set-up. You should use **Full Filesystem** only if a **Default** search type fails to discover all databases.

2 To begin the search, click on **Discover Databases**.

When the search is finished, a message box opens and tells you how many databases were found that fit the specified characteristics.

3 Click on **OK** in the message box to close it and then click on **Add** in the Database Discovery Selection section of the window.

A window opens that lists the databases that were found.

4 Select one or more databases to add and click on **OK**.

They are added to the Database Definitions list box.

5 Click on **OK** to complete the process of adding those databases or Sybase servers you selected to be managed by DataHub. If you are discovering databases or Sybase servers as part of adding a new managed host, the host is also added.

The discovered databases are not immediately displayed in the object tree. To display the discovered databases, select the host object occurrence, select the **Display** action button, and select the Database object type from the selection list. For Sybase, you have to display the Sybase Server object type first and then the Database object type to see discovered databases on that server.

Displaying Your Networked Systems

Using the DataHub **Network** action, you can graphically display the systems that are part of your network.

To display the systems

1 Select the Host object type and click on the **Network** push button.

If you have a large network, this action might take some time.

A window opens and displays the systems on the network connected to the control point host. If you have a large network, you might have to maximize the size of this window to see everything.

The window is sorted by name; it has two areas. The left-hand area has a list of the network segment. The segment that has the control point host is initially selected.

In the right-hand area of the window is a graphical representation of all the hosts on the selected segments. If you select any of the check boxes in the left-hand area, the graphical representation of the network is displayed in the right-hand area.

2 Close this window by clicking on the **OK** push button.

Adding Definitions for Your Existing Sybase Servers

Sybase servers reside on a host and provide the access to the Sybase databases on that host. DataHub manages the Sybase servers and, through them, their databases.

Before adding the definition of a Sybase server, you need to know the server name, the $SYBASE directory for that server, and the description you want for the server.

To add the definition of a Sybase server to the control database

1 Select the host on which this Sybase server resides and double-click on the host. The Select One or More window opens.

2 Select **Sybase Server** and click on **OK**. The Sybase Server object type is displayed on the object tree.

3 Select the Sybase Server object type and click on the **Add** push button. The Add Server window appears.

4 Fill out all of the fields in the Add Server window. For an explanation of the parameters, see the online help.

If you want to add out-of-the-box monitoring to this Sybase server, select the **Add Standard Monitoring** check box. See “Out-of-the-Box Monitoring” on page 293 for more information.

5 Click on **OK**.

Adding Definitions for Your Existing Databases

To manage your existing databases, you need to add their definitions to the control database.

Syb If your existing databases are Sybase databases, you do not have to add their definitions to the control database. The Sybase databases are managed directly by the Sybase server.

Important: To manage an INGRES installation, you must add the INGRES master database, iidbdb, to DataHub as a managed database.

Before adding the database, you need to know

- Name of the database
- Database Type
- DB2 Instance name
- Ing Ora The database directory. The database directory corresponds to $ORACLE_HOME for ORACLE and $II_SYSTEM for INGRES.
- Description of the database

To add the definition of an existing database

1 Select the Database object type and click on the **Add** push button. The Add Database window opens.

2 Enter the parameters in the window. For an explanation of the parameters, see the online help.

DB2 Ing Ora If you want to add out-of-the-box monitoring to this database, select the **Add Standard Monitoring** check box. See “Out-of-the-Box Monitoring” on page 293 for more information.

Chapter 4. Managing Administrator Authorizations and Privileges

To be able to start and use DataHub, you need:

- An operating system login with authority to execute the DataHub program on the control point
- Access to the control database
- Certain privileges and authorities on the control tables and views in the control database
- Privileges to display and act on objects in the DataHub object tree.

Right after installing DataHub, the only user that can start and use DataHub is the user name you provided as the owner of the control tables and views during installation (see the *DataHub for UNIX Operating Systems Installation and Configuration Guide* for more information). To authorize other users to use DataHub, this user:

1. Creates user names in the control database
2. Gives those users privileges to connect to the control database
3. Authorizes those users for DataHub operations by granting them specific access to certain DataHub control tables and views, depending on what types of operations they will be performing.
4. Adds those users to one or more DataHub administration groups that have specific DataHub privileges

Creating users for the control database and granting privileges for the control database is discussed in "Creating Control Database Users" on page 36.

Granting privileges for user or Sybase login names and granting authorizations are discussed in Chapter 12, "Managing Access to Databases" on page 172.

Authorizing users for DataHub operations is discussed in "Authorizing Users to Control Tables and Views for DataHub Operations" on page 36.

Creating DataHub administration groups and managing the display and action privileges for groups is discussed in "Managing DataHub Access with Administration Groups" on page 42.

Assigning individual users to DataHub administration groups is discussed in "Managing Access by Individual Users" on page 47.

Authorizing Users to Control Tables and Views for DataHub Operations

DataHub provides a utility that you can run to grant users specific access (SELECT, INSERT, UPDATE, or DELETE) to certain DataHub control tables and views, so the users can perform different types of DataHub operations.

To invoke this utility, change to the DataHub install directory and run “dhuauth”. See the online help for “dhuauth” for more information.

The types of operations for which you can authorize users are:

Basic operations
Users can display objects on the object tree and work with database objects.

Host management
Users can add, change definitions, drop, and stop manage hosts.

Sybase server and database management
Users can

- Add, change definitions, drop, and stop manage databases
- Add, change definitions, and stop manage Sybase servers.

Working with DataHub applications
Users can perform actions on DataHub applications.

Working with watchdog checks and rules
Users can add, update, and delete watchdog checks and rules.

DataHub user and group management
Users can create a new DataHub administration group, manage users in DataHub administration groups, and change object display and action privileges for DataHub administration groups.

Important: Do not authorize basic users for these operations.

Creating Control Database Users

The control database holds all of the control tables and views for DataHub itself. Therefore, to be a valid user of DataHub, you must be a valid user defined to the control database. This control database user name must have certain access privileges on the control database and be authorized, at a minimum, to SELECT the DataHub control tables and views, and to SELECT and UPDATE the “alert_table” table. Some DataHub operations (such as adding a managed host or working with watchdogs) require authorizing the user to do UPDATE, INSERT, and DELETE on DataHub control tables and views.

How you grant access privileges and the authority to SELECT the DataHub control tables and views depends on what RDBMS type you are using for your control database. You might have to create a new user name on the control database itself, and grant that user access privileges (such as connect) to access the database, and the authority to SELECT the DataHub control tables and views. If your control database is a Sybase database, you might have to create a new login name on the Sybase server that has the control database and then give a user name associated with that login the privileges to SELECT the DataHub control tables and views.

This chapter explains some of the things you need to do for a control database user based on what RDBMS type you are using for your control database.

> **FYI:** To display an object type (like User) from the control database occurrence on the object tree, double-click on the control database occurrence, select the object type from the Display Types window, and click on **OK**.

Creating new users is discussed in "Creating New Database Users" on page 173. Granting database access privileges, object privileges, and object authorizations is discussed in Chapter 13, "Managing Security of Databases" on page 181 (for databases) and "Server Roles" on page 81 (for Sybase server logins).

DB2 Control Database

If DB2 is your control database, you must choose the name of the new user from the list of operating system user names. Create the new user by following the procedure in "Creating New Database Users" on page 173.

Granting Access Privilege to the New User

The DataHub user needs access privilege to the control database. You must grant at least the CONNECT privilege to the new control database user.

You can grant access privileges by:

- Granting the privilege directly to the user.
- DB2v2 Granting the privilege to the group to which the user belongs. In this case, all users in the group get the privilege. For a discussion of DB2 Version 2 groups, see "Managing Database Groups" on page 176.

All users accessing a DB2 database are automatically members of the group PUBLIC.

To grant access privileges to an individual new user

1 Select the user name and click on the **DB Authorities** push button. The Select One or More Authorities window opens.

2 Select the access privilege you want for this user. For information on this window, see the online help.

3 Click on the **OK** push button.

To grant access privileges to a DB2 Version 2 group

1 Select the group occurrence and click on the **DB Authorities** push button. The Select One or More Authorities window opens.

2 Select the access privilege you want for this user. For information on this window, see the online help.

3 Click on the **OK** push button.

Authorizing Use of DataHub Control Tables and Views

To start DataHub, a control database user must be able to SELECT the DataHub control tables and views. To be able to perform some DataHub actions that involve modifying DataHub control tables, a control database user must be able to do UPDATE, DELETE, and INSERT on selective DataHub control tables and views. For example, to add a managed host using the **Add** action on the Host object type, you need SELECT and INSERT privilege against the table "dba_host".

DataHub provides an executable file for granting authorizations to users (see "Authorizing Users to Control Tables and Views for DataHub Operations" on page 36).

To grant the authorizations for a control database user, you can:

- Grant the authorizations directly to the user
- DB2v2 Grant the authorizations to the group to which the user belongs. In this case, all users in the group get the authorizations. For a discussion of DB2 Version 2 groups, see "Managing Database Groups" on page 176.
- Give the user (or group) the DBADM privilege (see "Granting Access Privilege to the New User" on page 37). This gives the user full access to the objects in this database, including the DataHub control tables and views.

See “Administering Authorizations” on page 184 for information on administering authorizations.

INGRES Control Database

If INGRES is your control database, you can choose any name for the new user. Create the new user by following the procedure in “Creating New Database Users” on page 173.

Authorizing Use of the DataHub Control Tables and Views

To start DataHub, a control database user must have the SELECT authorization on the DataHub control tables and views. To be able to perform some DataHub actions that involve modifying DataHub control tables, a control database user must be able to do UPDATE, DELETE, and INSERT on selective DataHub control tables and views. For example, to add a managed host using the **Add** action on the Host object type, you need SELECT and INSERT privilege against the table “dba_host”.

DataHub provides an executable file for granting authorizations to users (see “Authorizing Users to Control Tables and Views for DataHub Operations” on page 36).

To grant the authorizations for a control database user, you can:

- Grant the authorizations directly to the user
- Grant the authorizations to the group to which the user belongs. In this case, all users in the group get the authorizations. For a discussion of INGRES groups, see “Managing Database Groups” on page 176.

See “Administering Authorizations” on page 184 for information on administering authorizations.

ORACLE Control Database

If ORACLE is your control database, you can choose any name for the new user. Create the new user by following the procedure in “Creating New Database Users” on page 173.

Granting Access Privileges to the New User

To be able to access the control database, a control database user must have the CREATE SESSION privilege. You can give this privilege to an individual user, or give it to a role that is granted to the user.

For information on ORACLE roles, see “Administering ORACLE Roles” on page 182.

To grant a privilege to an individual control database user

1 Select the user and click on the **Privilege** push button. The Select One or More Authorities window opens.

2 Select at least the CREATE SESSION privilege and click on the **OK** push button.

Authorizing Use of the DataHub Control Tables and Views

To start DataHub, a control database user must have the SELECT authorization on the DataHub control tables and views. To be able to perform some DataHub actions that involve modifying DataHub control tables, a control database user must be able to do UPDATE, DELETE, and INSERT on selective DataHub control tables and views. For example, to add a managed host using the **Add** action on the Host object type, you need SELECT and INSERT privilege against the table "dba_host".

DataHub provides an executable file for granting authorizations to users (see "Authorizing Users to Control Tables and Views for DataHub Operations" on page 36).

To grant the authorizations for a control database user, you can:

- Grant the authorizations directly to the user
- Grant the authorizations to the role that is given to the user. In this case, all users with this role get the authorizations. For a discussion of ORACLE roles, see "Administering ORACLE Roles" on page 182.

See "Administering Authorizations" on page 184 for information on administering authorizations.

Sybase Control Database

If your control database is a Sybase database, create the new user by following the procedure in "Creating New Database Users" on page 173.

> **FYI:** If a login on the Sybase server with the control database has the sa_role, you do not have to create a user name on the control database for that login. That login has full access to the DataHub control database objects (including the control tables and views).

Granting Access Privilege to the New User

To start DataHub, a control database user must be able to access the control database and to SELECT the DataHub control tables and views. To be able to perform some DataHub actions that involve modifying DataHub control tables (like the **Add** action on the Sybase Server object type, to add a managed server), a control database user must be able to UPDATE, DELETE, and INSERT on selective DataHub control tables and views. For a Sybase control database, you can give a user the privilege to access the control database in one of the following ways:

- Define a Sybase server login for the user and then create a database user associated with that login to the control database.
- Define a Sybase server login for the user and then create an alias to the control database database owner, dbo, for that login. By adding the alias, you give the user all the access privileges to the control database and control tables and views as the database owner has.

  For information on creating Sybase aliases, see "Sybase Aliases for Sybase Database Users" on page 175.

  For information on assigning Sybase server roles, see "Managing Logins on a Sybase Server" on page 78.

Authorizing Use of the DataHub Control Tables and Views

To start DataHub, a control database user must be able to SELECT the DataHub control tables and views. To be able to perform some DataHub actions that involve modifying DataHub control tables, a control database user must be able to do UPDATE, DELETE, and INSERT on the DataHub control tables and views. For example, to add a managed host using the **Add** action on the Host object type, you need SELECT and INSERT privilege against the table "dba_host".

DataHub provides an executable file for granting authorizations to users (see "Authorizing Users to Control Tables and Views for DataHub Operations" on page 36).

To authorize a control database user for these, you can:

- Grant the authorizations directly to the user
- Grant the authorizations to the group to which the user belongs. In this case, all users in the group get the authorizations. For a discussion of Sybase database groups, see "Managing Database Groups" on page 176.
- Create an alias to the control database database owner, dbo, for this user's server login. By adding the alias, you are giving the user all the privileges on the control database as the database owner has.

  In this case, because the user is identified to the control database as "dbo", the user's display and action privileges cannot be further restricted through the DataHub **User Management** and **Group**

Management menu choices. You can access these menu choices by selecting the **Privilege** choice under the **Administer** pull-down menu.

FYI: For a user connecting to DataHub using a Sybase server login with the sa_role role, that user will have full access to all of the tables and views in the control database.

Because the user is identified to the control database as "dbo", the user's display and action privileges cannot be restricted through the DataHub **User Management** and **Group Management** choices found under the **Privilege** choice of the **Administer** menu.

See "Administering Authorizations" on page 184 for information on administering authorizations.

Managing DataHub Access with Administration Groups

In DataHub, you can define *DataHub administration groups* (*DataHub admin groups*, for short) to organize those users who can access DataHub. You might set up different DataHub admin groups to perform different kinds of tasks. For example, you might set up one group for administrative tasks, such as creating and archiving databases, and another group for operational tasks, such as running utilities and optimizing performance.

Managing access to DataHub by administration groups of users (DataHub admin groups) involves:

- Defining new DataHub admin groups
- Managing authorizations to invoke actions on objects
- Granting and revoking privileges to display objects

Defining New DataHub Admin Groups

DataHub comes with a predefined DataHub admin group called DATAHUB. Any user in this group is authorized to invoke any of the actions allowed for an object in the object tree and display any object on the object tree.

However, you might want to define your own DataHub admin groups and customize the list of actions available for the users that you assign to those groups. For example, you might want to prevent some users from being able to invoke the **Stop Managing** action for all hosts or for all objects that have the **Stop Managing** action.

You might also want to customize the list of object types that certain users can display on the object tree. For example, you might want to prevent some users from displaying all of the user names for a particular database.

FYI: The action and display privileges for the DATAHUB group cannot be modified.

To customize the types of things certain users can access, you

1. Create a named DataHub admin group for these users
2. Specify the actions this group can perform on objects
3. Specify the objects this group is allowed to display

To create a named DataHub admin group

1 At the top of the DataHub window, click on the **Administer** menu, select **Privilege**, and then select **Create Group**. The New DataHub Group window appears.

2 Type the name of the group you want to add.

3 Select the **OK** push button.

The new DataHub admin group has no action or display privileges until you add them using the **Group Management** and **Display Privileges** choices under the **Privilege** choice of the **Administer** pull-down menu. For information, see Managing Authorizations for Object Actions and Granting and Revoking Privileges to Display Objects.

Granting and Revoking Privileges to Display Objects

When you select an object on the object tree and click on the **Display** push button, a list of object types eligible for display appears. This list of object types is determined by the display privileges granted to DataHub admin groups.

For example, you can grant one DataHub admin group privileges to display only views and no tables on a database. Then, when a user in this group selects a database on the object tree and clicks on the **Display** push button, only the **View** object type is listed.

You can grant or revoke display privileges for displaying built-in DataHub objects and user-defined objects.

To grant or revoke display privileges for DataHub admin groups

1 At the top of the DataHub window,

   a Click on the **Administer** menu.

   b Select **Privilege**.

   c Select **Display Privileges**.

d Select either

- **Built In**, for DataHub object types
- **User Defined**, for user-defined object types

The Select Groups for Privileges window opens.

2 In the Select Groups for Privileges window, select the DataHub admin groups you want and click on the **OK** push button. The Select Platforms window opens. A list of RDBMS types and operating system types is displayed in this window.

There are different object types you display at the operating system and RDBMS levels. For example, hosts and OS users are object types which are displayed at the operating system level; tables, Sybase servers, and databases are displayed at the RDBMS level.

3 In the Select Platforms window, select one or more RDBMS types or operating system types for whose objects you currently want to specify the display privileges.

For example, because new DataHub admin groups have no display privileges for any object types, select all of the RDBMS types and operating system types you have on your system, so that you can grant the groups' display privileges for all object types.

If you have existing DataHub admin groups and you want to change the groups' access for a particular set of object types, for example, all Sybase object types, select **Sybase**.

4 Once you have selected the platforms you want, click on the **OK** push button. A window opens in which you can specify the display privileges for each DataHub admin group.

The window is divided into two areas. In the left-hand area are rows consisting of:

- From Class (such as Host)
- Qualifier on the object type (depending on the operating system or database type for which the display privilege is applicable)
- To Class (such as Database)

Hint: You need to use the horizontal scroll bars to see all three columns in the left-hand area. You might need to use the horizontal scroll bar in the right-hand area if you selected multiple DataHub admin groups.

The From Class is the object type that you double-click on in the object tree to display the To Class object type (you display the "To Class" from the "From Class").

For example, if you double-click on a host in the object tree, you get a list of object types you can display. In this example, the host is the "From Class" object type and the object types in the Display Types window are the "To Class" object types.

In the right-hand area of the window are columns representing the DataHub admin groups you specified.

5 To authorize a DataHub admin group for a specific displaying To Class-From Class combination, select the area under the DataHub admin group name in the same row as the object types and click with mouse button 1. A **Yes** is displayed in the area.

To revoke a display combination from a DataHub admin group, click on the **Yes** under the DataHub admin group name; it is removed.

To authorize all display combinations in the window to a DataHub admin group, click on the DataHub admin group name. A question box opens asking you if you want to grant all actions to the group.

- Select **Yes** to grant all display privileges.
- Select **No** to revoke all display privileges.
- Select **Cancel** to cancel the operation.

For more information about this window, see online help.

6 To commit the changes, click on the **OK** push button. Otherwise, click on the **Cancel** push button.

Managing Authorizations for Object Actions

Each object on the object tree has certain actions associated with it. For example, the **Create** action is associated with such objects as databases, tables, and views. When you select an object in the object tree, the actions available for that object are displayed in the Action window. You can specify what actions you want available to the users in a DataHub admin group.

For example, to ensure that the users in a particular DataHub admin group cannot drop databases, you remove the group's privilege for the **Drop** action on databases. The **Drop** push button will not display when a user in that group selects a database in the object tree.

To authorize one or more DataHub admin groups for object actions

1 At the top of the DataHub window,

   a Click on the **Administer** menu.

   b Select **Privilege**.

   c Select **Group Management**.

The Select Groups for Privileges window opens.

2 In the Select Groups for Privileges window, select the DataHub admin groups you want and click on the **OK** push button. The Select Platforms window opens. A list of RDBMS types and operating system types is displayed in this window.

There are DataHub actions for objects at both the operating system and RDBMS levels. For example, hosts and OS users are objects which have DataHub actions at the operating system level; tables, Sybase servers, and databases have DataHub actions at the RDBMS level.

3 In the Select Platforms window, select one or more RDBMS types or operating system types for whose objects you currently want to authorize the selected DataHub admin groups' to perform actions on the objects.

For example, because new DataHub admin groups are not authorized to perform any actions on any objects, select all of the RDBMS types and operating system types you have on your system, so that you can authorize the DataHub admin groups to perform all actions on all the objects.

If you have existing DataHub admin groups and you want to change the groups' authorizations for a particular set of actions on a particular set of objects, for example, actions on Sybase objects,

4 Once you have selected the platforms you want, click on the **OK** push button. A window opens in which you can specify the actions available for each DataHub admin group.

The window is divided into two areas. In the left-hand area are rows consisting of:

- A name of an action (such as Drop)
- The object associated with the action (such as Database)
- The type of RDBMS for this combination (such as Sybase)

In the right-hand area are columns representing the DataHub admin groups you specified.

Hint: You need to use the horizontal scroll bars to see all three columns in the left-hand area. You might need to use the horizontal scroll bar in the right-hand area if you selected multiple groups.

5 To authorize a DataHub admin group to perform a specific action-object combination, select the area under the group name in the same row as the action and click with mouse button 1. A **Yes** is displayed in the area.

To revoke an authorized action from a DataHub admin group, click on the **Yes** under the group name; it is removed.

To authorize all action-object combinations in the window to a group, click on the group name. A question box opens asking you if you want to grant all actions to the group.

- Select **Yes** to grant all action privileges.
- Select **No** to revoke all action privileges.
- Select **Cancel** to cancel the operation.

For more information about this window, see online help.

6 To commit the changes, click on the **OK** push button. Otherwise, click on the **Cancel** push button.

Managing Access by Individual Users

Each DataHub admin group has specific privileges to display objects on the object tree and to invoke actions in the Action window. To grant control database users privileges to display objects and invoke actions, you must add them to a DataHub admin group.

But first: Users must be defined to the control database before they can be added to a DataHub admin group. See "Creating Control Database Users" on page 36 for more information.

To add or remove users to or from a DataHub admin group

1 At the top of the DataHub window,

a Click on the **Administer** menu.

b Select **Privilege**.

c Select **User Management**.

The Select Groups for Privileges window opens.

DataHub comes with a predefined DataHub admin group called DATAHUB. Any user in this group is authorized to invoke any of the actions allowed for an object in the object tree and display any object on the object tree.

2 Select the DataHub admin groups you want and click on the **OK** push button. A window opens. It is split into two areas:

- User Name column.

  This column lists in rows all the control database users that have been put in this column.

- A set of columns, each of which is headed by the name of a DataHub admin group defined to DataHub.

For more information about this window, see online help.

Under a DataHub admin group name, a **Yes** appears if the user in that row is assigned to that DataHub admin group. If the user is not assigned to the DataHub admin group, the area is blank.

Right after DataHub is installed, the only user assigned to a DataHub admin group is the user defined as the owner of the control tables and views during the installation; this user is assigned to the DATAHUB group.

To assign other control database users to DataHub admin groups, you:

a. Add the user names to the column of users, then
b. Add the user to one or more DataHub admin groups

3 To add user names to the column of users, click on the **Create User** push button. The Select User window opens. This window shows a list of all the users defined to the control database that are not currently in the User Name column. Select one or more users and click on the **OK** push button.

If all of the users defined in the control database are already added to the User Name column, when you click on the **Create User** push button, a message will state that all database users have been assigned to a DataHub admin group.

The users you selected are displayed in the column.

4 To add a user to a DataHub admin group, select the area under the group name in the same row as the user and click with mouse button 1. A **Yes** is displayed in the area.

To remove a user from a DataHub admin group, select the **Yes** under the group name; the **Yes** is removed.

To add all users in the window to a DataHub admin group, select the group name. A question box opens asking you if you want to add all users to the group.

- Select **Yes** to add all users to the group.
- Select **No** to remove all users from the group.
- Select **Cancel** to cancel the operation.

5 To commit the changes, click on the **OK** push button. Otherwise, click on the **Cancel** push button.

Chapter 5. Managing Multiple Objects

DataHub provides a grouping function that allows you to perform database systems management tasks on a number of managed objects with a single action. Working with groups of objects can help you reduce repetitive tasks and simplify the display of objects in the DataHub window.

You can use the grouping function to work with multiple objects in two ways:

1. Create DataHub groups in the DataHub object tree that represent a collection of managed objects. You can perform tasks on these groups on an ongoing basis.
2. Select multiple objects in the object tree and invoke a single, one-time action for all of them. Working with objects in this way does not create a group.

Which of these two group options you use depends on the type of work you are doing and whether it is a one-time task or something that you do on a regular basis.

DataHub groups represent a collection of objects. The objects in the group have a common structure and can be managed as a unit on an ongoing basis. For example, it might be useful to create a database group when managing the database resources for an application that runs on several hosts and requires the same databases, users, and tables on each host. Such applications are typical in the banking and retail industries and sometimes run on hundreds of servers. In this case, the database objects that you want to manage with DataHub already exist. If not, DataHub's normal administration functions can be used to create them as needed.

You can also perform database management tasks on more than one object at a time without creating group objects. This is most useful for tasks that you are not going to perform on an ongoing or regular basis. You simply select the objects in the object tree and then select the action you want to perform on them. The action is carried out on all selected objects. This method of working with multiple objects is described in more detail in "Managing Multiple Objects without Creating Groups" on page 63.

Working with Groups and Group Actions

You can create three kinds of groups:

Host Groups
Host groups contain hosts of the same type or different types. For example, a host group could be all AIX hosts, all HP-UX hosts, or a mix of Sun Solaris and AIX hosts.

Server Groups
Server groups apply only to Sybase. Server groups can contain multiple Sybase servers on one or more hosts of the same or different type.

Database Groups
Database groups apply to DB2, ORACLE, and Ingres databases. These groups can contain databases of the same type on one or more hosts of the same or different type. For example, you can create a group of DB2 databases, residing at AIX, HP-UX, and Sun Solaris hosts.

Figure 4 shows an example of the DataHub object tree with a database group displayed.

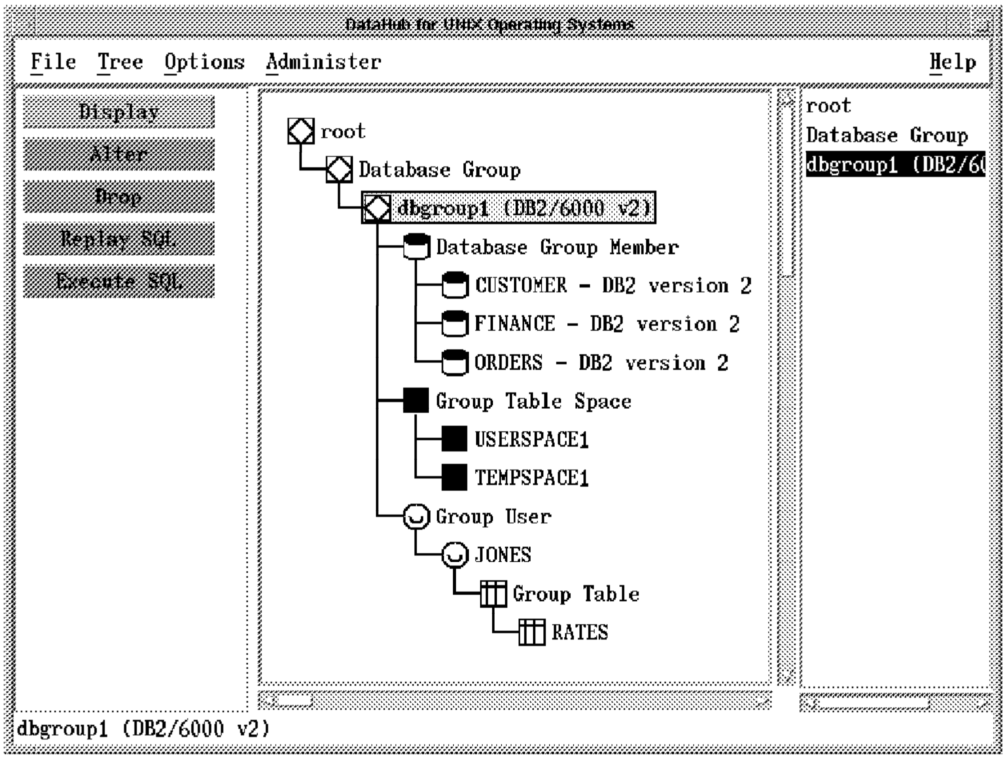


Figure 4. Example of a Database Group

dbgroup1 An occurrence of a database group. There are three databases in this group, shown as:

- "FINANCE"
- "CUSTOMER"
- "ORDERS"

These are database group members. If you want to perform a task on all three members of the group, select database group "dbgroup1". To perform a task on only one of the three databases, select it as a database group member from within the group. All the actions which are available for databases outside of the group are also available from inside the group. This way you need not go outside of the group to perform database tasks on individual databases that are part of a group.

Group Tablespace
There are two occurrences of the Group Table Space object type, "TEMPSPACE1" and "USERSPACE1". These two table spaces exist in each of the three database group members "FINANCE", "CUSTOMER", and "ORDERS".

Group User Group user "JONES" is the owner of the group table "RATES". "JONES" exists as a user in each of the three database group members, "FINANCE", "CUSTOMERS" and "ORDERS".

Group Table
The group table, "RATES", is a table that exists in each of the three database group members. It is owned by user "JONES".

Group objects such as group table spaces, group users, and group tables, can be added to the group definition to document the objects used by an application. All tasks on groups can be performed on the group occurrence. See "Management Tasks You Can Perform on Groups" on page 53 for more information.

The following tables show the object hierarchies for the different types of groups:

Table 2. Host and Sybase Server Group Hierarchies

Host Groups	**Sybase Server Groups**
Host Group	Server Group
Host Group Member	Server Group Member
User Password Set	Group Database
	Group User
	Group Table
	Group Database Device
	Group Login
	User Password Set

Table 3. DB2 V1 and V2 Group Hierarchies

DB2 V1 Groups	**DB2 V2 Groups**
Database Group	Database Group
Database Group Member	Database Group Member
Group User	Group Table Space
Group Table	Group User
User Password Set	Group Table
	User Password Set

Table 4. Oracle and Ingres Group Hierarchies

ORACLE Groups	**Ingres Groups**
Database Group	Database Group
Database Group Member	Database Group Member
Group Tablespace	Group User
Group User	Group table
Group Table	User Password Set
User Password Set	

Management Tasks You Can Perform on Groups

You can perform the following types of management tasks on groups:

- Create, modify, and delete groups. You can also display the relationships of group objects to each other (see “Creating, Modifying, and Deleting Groups” on page 54).
- Monitor group objects by defining watchdog checks and rules for them (see “Monitoring Groups” on page 58).
- Manage group objects by executing SQL commands on them (see “Executing SQL Commands on Groups” on page 60).

Any action invoked on the group as a whole is applied to all members of the group. For example, if you create a watchdog check for a database group, the check is created for every database in the group. Any action performed on a group object is applied to all objects of that type in all members of the group. Using Figure 4 on page 51 as an example, if you create a watchdog check on group table “RATES”, owned by “JONES”, the check is created for every table “RATES” owned by “JONES”, in every database in the database group.

You cannot invoke group actions for an individual member. If you select a member of a group, such as a database in a database group, a server in a server group or a host in a host group, you can work with the individual member in the same way that you would with any individual object, not part of a group. All actions you can perform on the object when it is not part of a group can be invoked for the object when it is displayed as a group member.

Creating, Modifying, and Deleting Groups

When you create a host, server, or database group, you must first create the group, then create and assign a user password set for the group and its members. When this is done, you can add group objects, such as group users, group tables, and so on.

To create a group

1 Double click on the root object type and then select the group object type that you want to create (for example, Host Group).

2 Select the group object type displayed in the object tree and select the **Create** action.

A Define Group window appears, similar to Figure 5.

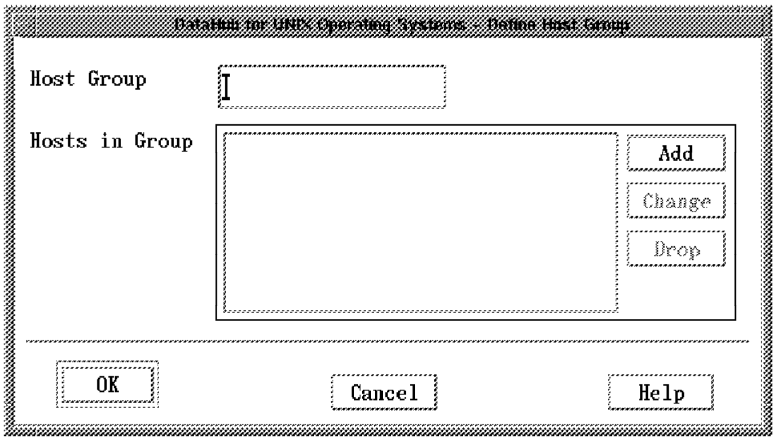


Figure 5. The Host Group Object Window

3 Enter a group name.

4 Click on **Add**.

5 Select the names of the objects that you want to add to the group in the window that appears. These objects are the hosts, servers, and databases that wil be the group members.

6 Click on **OK**. The new group occurrence is displayed under the corresponding group type in the object tree.

Modifying Groups

You can modify a group by adding, changing, and dropping the members of the group.

To add, change, or drop members of groups

1 Display and select the group occurrence that you want to work with.

2 Click on the **Alter** action. The Change Group window appears. From this window you can use the **Add**, **Change**, and **Drop** buttons in the window to add, replace, or remove members of the group.

When a new member is added to an existing group, previous group actions, such as adding group watchdog checks or executing SQL statements, are not automatically performed for the new group member. When group members are dropped, watchdog checks are not dropped for the individual members. DataHub converts them from group watchdog checks to individual watchdog checks. They can be updated or dropped as individual watchdog checks.

Deleting Groups

When you delete a group, you are dropping the DataHub definition of the group. The individual group members are not deleted or changed.

To drop a group

1 Select the group occurrence that you want to drop.

2 Click on the **Drop** action.

3 Confirm that you want to drop the group in the window that appears.

Adding and Removing Group Objects

Database objects that are common to all of the members of a group are referred to as group objects. For example, you might have a group table named “RATES” that resides at each of the databases in a database group. When you add a group object, you are adding the definition of existing objects to the definition of the group.

To add a group object

1 Select the group occurrence to which you are adding the group object and click on the **Display** action. (You can also double-click on the group occurrence.)

2 From the list of object types, select the type of object you want to add and click on **Ok**. This object type is displayed on the object tree.

3 Select the object type in the object tree and click on the **Add** button. The Add Group Object window opens.

4 In the Add Group Object window, enter the name of the group object.

5 Click **OK**.

This action does not create new objects at each of the hosts, servers, or databases within the group. The group object is a group definition for objects that exist for the group members. Note that DataHub allows you to create group objects that do not have corresponding, existing objects at one or more of the group members.

To remove a group object

1 Select the group object that you want to remove.

2 Click on the **Remove** action.

3 Confirm that you want to drop the group object in the window that appears.

4 Click **OK**.

As with groups, removing a group object does not affect the actual objects that it represents. These objects reside at the hosts, servers, or databases.

Creating, Choosing, and Modifying User Password Sets

Before you can perform actions on a group you must create a user ID and password for the group as a whole and specify the user ID and password of each group member. The user ID and password for the group is called a user password set and can be displayed in the DataHub object tree.

The user password set links the user ID and password for each group member to the user ID and password for the whole group. Once you have created a user password set, you can perform group actions without being prompted for the user ID and password for each member.

To create a user password set

1 Double click on a group occurrence and select User Password Set from the displayed list of object types.

2 Select the User Password Set object type that is displayed on the object tree.

3 Select the **Create** action.

4 Enter a User Password Set name and its password in the Create User/Password Set window. For all types of groups except host groups, you need to select a user password set type of either **DBM** or **DB**.

The User Password Set type for database groups and server groups can be assigned one of two kinds of logins. For some operations on database groups and databases, you need an OS login normally used to manage the database. This type of login is called a DBM login. For operations to execute SQL commands you need only a user ID and password combination to access the database, which does not necessarily have to be an OS login. See "Working with Connections Established by DataHub" on page 10 for more information.

5 Click on **OK**.

A second window opens showing all members of the group. Information about each member is shown.

6 Select each member in turn and click on the **Change** button to specify a user ID and password combination for that member. User IDs and passwords are verified when they are created or changed. The newly specified user ID is displayed in the window in the user name column, without the password.

7 Click on **OK**.

The User Password Set object you created is displayed on the object tree. You can create more than one user password set. For example, you can create different sets associated with different authorizations.

If you create only one user password set, it becomes by default the active one and is used for all subsequent tasks on group objects that require user IDs and passwords. If there is more than one set, use the **Choose** action to set the active user password set. If you do not choose a user password set, a DataHub window prompts you to select one the first time you attempt to do something that requires one.

To choose a User Password Set

1. From the object tree, select the User Password Set occurrence that you want to use.
2. Select the **Choose** action.
3. Enter the name of the set you want or click on the **List** button to get a list of sets to choose from.
4. Enter the password for the chosen set.

When you return to the DataHub window the color of the current or chosen User Password Set object icon is changed to indicate that it is now the active set. You can now use functions that require the user ID and password without being prompted.

To change User Password Set passwords

1. From the object tree, select the User Password Set occurrence that you want to change.
2. Select the **Update Set** action.
3. Enter current password, the new password, and then retype the new password.
4. Click **OK**.

To change user IDs and passwords for group members

1. From the object tree, select the User Password Set occurrence that you want to change.
2. Select the **Update Member** action. A window appears listing group members with their user names and other information.
3. Select a member from the list and click on the **Change** button. The Change Host/User Password window appears.
4. Change the user ID and password.
5. Click **OK**.

Monitoring Groups

This section describes the steps for creating watchdog checks and watchdog rules for groups. To monitor objects in a group, you create group watchdog checks. To define actions to be taken and the conditions under which they are to be taken, you create group watchdog rules. If the defined conditions are met on any of the monitored group members or objects, the action is taken. Objects representing watchdog checks and rules for groups and group objects appear in the DataHub object tree just as they do with regular objects.

To create a group watchdog check, select the group occurrence or any group object, but not an individual member. For example, to create a group watchdog check that monitors freespace in table space “USERSPACE1”, you can either select the group table space “USERSPACE1” or group “dbgroup1”. (See Figure 4 on page 51.) You can then display the object type Group Watchdog Check and select it after it is displayed on the object tree. Then, select the **Create** action. You need to define only one group watchdog check. It will cause table space “USERSPACE1” to be monitored for freespace in every database of the database or server group.

There are two advantages to creating the watchdog check under the object with which you want to associate it rather than the group occurrence:

1. Creating the watchdog check under an object such as a table space allows you to list all the predefined watchdog checks which exist for the table space object type.
2. You will be able to display the group watchdog checks for the table space under the table space. If you created them under the group occurrence, they will be displayed under the group occurrence.

You can create watchdog checks for individual group members. See Chapter 27, “Monitoring Your Databases and Operating Systems” on page 286 and “Working With Watchdog Rules” on page 305 for information about individual watchdog checks and rules.

To create a group watchdog check

1 Display the Group Watchdog Check object type under the group object that you want to monitor. To do this, double-click on the group object, select Group Watchdog Check from the Display Types window, and click on **OK**.

2 Select the Group Watchdog Check object type in the object tree and the **Create** action.

3 In the window that appears, define the group watchdog check.

When working with checks on groups, the list of available checks from which you can choose depends on whether the group is homogeneous or heterogeneous. For a homogeneous group, DataHub presents all the checks for that group type. For a heterogeneous host groups, DataHub lists all the checks that have a common definition and name for all supported host types.

To create a group watchdog rule

1 Display the group watchdog checks to which you want to add a rule.

2 Select all of the group watchdog checks on which you want to define this rule. You can select more than one check by holding the **Ctrl** key while clicking on the check name. The watchdog checks must be group watchdog checks selected from the same group.

3 Select the **Create Rule** action. The Create Rule window opens.

4 Specify the rule. For explanations of the parameters on this window see the online help.

To modify group watchdog checks and rules

1 Display and select the watchdog check or rule that you want to modify.

2 Click on the **Alter** action button.

3 In the window that appears, modify the watchdog check or rule.

4 When complete, click **OK**.

To drop group watchdog checks and rules

1 Display and select the watchdog check or rule that you want to drop.

2 Click on the **Drop** action button.

A window appears prompting you to click on **Yes** to continue or **No** to cancel.

3 Click on **Yes**.

Executing SQL Commands on Groups

You can perform administrative tasks on groups of databases for DB2, Ingres, and ORACLE and on groups of servers for Sybase by using the **Execute SQL** action in the Action window. The SQL statement you enter for a database or server group is applied to all of the members of the group. To re-try SQL statements that fail, use the **Replay SQL** action in the Action window.

Using Execute SQL

From the Execute SQL window, you can enter DDL, stored procedures, and grant, revoke, alter, create, insert and update statements for groups of databases or servers. Group 'Select' statements are not permitted.

To run an SQL command on a Database or Server Group

1. Select a Server Group occurrence or a Database Group occurrence and click on the **Execute SQL** action.

   The Specify Group Commands window opens. This window shows the name of the selected group, the name of the selected User Password Set, and the Database Type.

2. Click on the **Add** button to specify SQL commands.

3. Enter one or more commands.

   DataHub tries to make a temporary connection to each database in the database group. If it is not possible to connect to all of the members of the database group, DataHub proceeds only if you confirm that you want to continue. When the commands have been issued to the group members DataHub releases the connections to free up system resources as quickly as possible.

4. Click on **OK**.

   The command is executed on all databases in the database group (or all servers in the server group). DataHub executes the SQL statements simultaneously on all members of the group and waits until all group members respond. If the SQL statement executes successfully on all group members, DataHub returns to its main window.

When SQL Commands Fail

If the SQL statements cannot be executed successfully on one or more of the databases or servers, an error message is displayed that shows the SQL commands that failed, and the error message and error code. Failed commands can be retried and executed again using the **Replay SQL** action.

If more than one SQL command is specified and one of the commands fails, the commands following it are not executed. For example, if the SQL commands are SQL1, SQL2, and SQL3, and SQL2 fails, SQL3 is not tried. SQL2 and SQL3 are both stored as failed commands.

Whenever one SQL command for a given group member fails, a rollback statement is issued for that member. Note that for members that have executed commands SQL1, SQL2, and SQL3 successfully, no rollback statement is issued. Hence, integrity is guaranteed for individual members.

If the command failed on every member, DataHub informs you by issuing an error message to that effect. In this case, none of the group members are affected and group integrity is not disturbed.

Using Replay SQL

You can maintain integrity at the level of the entire group (group integrity) by using the **Replay SQL** action to execute the SQL statements on the members where the failures occurred. The **Replay SQL** action allows you to replay SQL commands that failed for members in a group when the **Execute SQL** action was run. You can use **Replay SQL** to view all sets of commands that have been executed on a group, and remove sets of SQL commands from the command history.

To replay SQL commands that fail for group members

1. Select a Server Group or Database Group occurrence.
2. Click on the **Replay SQL** action.

   A window appears showing which commands have been executed for the selected group and which User Password Set was used.
3. Use the **Previous Commands** and **Next Commands** buttons to select the SQL commands that you want to replay.
4. Select the **Failed Group Member** where the SQL statement is to be executed. The failed SQL statements and the associated error messages from the previous execution are displayed in the **Failed Commands** list box.
5. Click on **Replay Commands for Group Member**. If these commands fail again, the existing failure information is overwritten with the new failure information.

   If the command set contains more than one member, you can choose not to execute the first one or more commands. The reason for this is that when the command set is executed, a command in the command set might have failed, but previous commands might have succeeded and autocommitted. If one of the commands that completed successfully is repeated, it can result in an error.

The list of commands that have been executed is specific to the User Password Set that was used. For example, if you used the Execute SQL action with User Password Set A, you would not see the commands that have been executed using User Password Set B. This is because SQL commands can contain sensitive information, such as a password in a CREATE USER statement. The result of many SQL commands is dependent on the user ID under which the command is tried. For example, a CREATE TABLE command creates the table for a certain user.

Managing Multiple Objects without Creating Groups

You can work with multiple objects without creating groups. When you select more than one object type or occurrence in the object tree, the actions which can be performed on selected objects are shown in the Action window. For example, if you want to create the same table in multiple databases, you select the object type Table under those databases and the action **Create**. If you want to give user authorizations on multiple tables, you select the tables and the action **Authorization**.

To create multiple objects

1. Select the object type (such as Table) under the database objects where you want to create a new object. The database objects need not be on the same database or server.
2. The Create Object window opens, where “Object” is the name of the type of object you selected. For example, **Create Table** appears if you selected multiple occurrences of the object type Table. At the top of the window are the databases or servers under which you selected the object type Table.
3. Select the host, server, or database from which you want DataHub to extract any pertinent information.

   For example, if you are creating multiple Oracle tables, DataHub uses the Oracle database which you select at the top of the window to extract the users known to this database. If a user does not exist for one of the databases, the action fails for that database and is cancelled before attempting to create tables at the remaining databases. This is true for Oracle, Ingres, Sybase and DB2 V1 databases. Note that DB2 V2 does not require that users exist.
4. Enter the name of the user for which you want to create the object. You can also click on the **List** button to get a list of users known to the database you selected.
5. Enter the object name.

   In the case of a table:

   a. Click on **Add** to define the table columns.
   b. The Add Column window opens. Enter column definitions as required. Click on the **Add** button after each column definition to add the column to the table definition.
   c. Click on **Close** when you have defined all columns. This returns you to the previous window.
6. Specify the remaining information.

7. Click **OK**.

The object you just created will be displayed in the object tree under each of the hosts, servers, or databases.

The **Create** action is available for the following object types:

Sybase

database type
login type
user type
table type

DB2 V1

user type
table type

DB2 V2

user type
table type

ORACLE

user type
table type

Ingres

table type

To drop multiple objects: You can also drop multiple objects of the same or a different type, for example, users and tables. If you drop multiple objects you are prompted to indicate whether you want to drop all objects without confirmation or whether you want to be prompted for confirmation for each object being dropped. If you opt for no confirmation, all objects selected are dropped immediately without further prompting. Otherwise, the **Drop** action works as it does with single objects.

Managing Hosts

Chapter 6. Managing Hosts with DataHub

To be able to manage hosts with DataHub, you set the host definitions and add them to DataHub. The definitions are then stored in the control database and DataHub uses these definitions to manage the hosts.

Managing host definitions with DataHub involves:

- Adding and modifying host definitions
- Stop managing hosts
- Monitoring host information

Adding Host Definitions and Changing Host Descriptions

In order to add a host definition, you need to know the network name for the host, the directory where DataHub is installed on that host, the type of host, and a host description.

The host name can be either the TCP/IP host name or a TCP/IP address. It must be the name by which the network knows this host.

The host description will appear in the object tree next to the hostname for this host.

If you do not know the host name, you can use the DataHub **Network** action to find the TCP/IP host name of the systems, both servers and clients, that are part of your network (see "Displaying Your Networked Systems" on page 33).

The host description will appear in the object tree next to the hostname for this host.

To add the definition of a remote host to the control database

1. Select the Host object type in the object tree.
2. Click on the **Add** push button. The New Managed Host window opens.
3. In the window, you must specify the network name for the host, the directory where DataHub is installed on that host, the type of host, and a host description.

   If you want DataHub to automatically discover databases, Sybase servers, or INGRES installations that exist on the host you are adding, select the **Database Discovery** check box. See page "Discovering Existing Databases or Sybase Servers" on page 31 for an explanation of DataHub's database discovery process.

If you want to add out-of-the-box monitoring for this host, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

4 Click on the **OK** push button of the New Managed Host window.

If you selected **Database Discovery**, the Database Discovery window opens (see "Discovering Existing Databases or Sybase Servers" on page 31) Otherwise, the added host object occurrence is displayed on the object tree.

You can change the description that appears next to a host occurrence on the object tree by using the **Definition** action. You can also use the **Definition** action to add out-of-the-box monitoring to the host.

To change the description that appears next to a host occurrence

1 On the object tree, select the host.

2 Click on the **Definition** push button. The Change Definition window opens.

3 Enter a new host description. (For explanations of the window parameters, see online help.)

If you want to add out-of-the-box monitoring for this host, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

Please note:

1. If you have previously added out-of-the-box monitoring for this host and you select **Add Standard Monitoring**, DataHub will add another set of monitors.
2. If you change the other parameters in this window and then run into problems, please verify that you have entered the correct values.

4 Click on the window's **OK** push button.

Stop Managing Hosts

If you stop managing a host, the host definition is removed from the control database and you are no longer able to manage that host using DataHub.

Attention: Do not stop managing the host that is the current control point. If you do this, the host is deleted from the managed host list and no one can login to DataHub any more.

Also, do not stop managing the host that has the control database. The host with the control database must always be a managed host.

To delete the definition of a host from the control database:

1 On the object tree, select the host occurrence.

2 Click on the **Stop Managing** push button. A window opens asking you if you want to stop managing the host.

3 From the newly opened window, click on the **Yes** push button to confirm.

Monitoring Host Information

You can use DataHub actions to display several kinds of the following information about hosts. Examples are:

- How disks and CPUs are being utilized
- How disk space is allocated
- Information about file systems, processes run by specific users, and general CPU usage
- Who uses the hosts
- What groups the users belong to

To find out how disks and CPUs are being utilized for a host:

1 On the object tree, select the host.

2 Click on the **Disk I/O** push button. The I/O Statistics window opens. It is split into two sections:

CPU utilization
Shows the percentage of time the system has spent in user mode (%User), system mode (%Sys), idle state (%Idle), and I/O waiting mode (%IO-Wait).

Time per physical disk
Shows for each physical disk on the system the percentage of time the disk is busy.

Both sections are updated every 10 seconds by default.

Color in the two sections identifies danger zones:

Color	**Meaning**
White	Safe
Yellow	Slight
Red	Danger zone

In this window, you can click on **Print** to print this window to a print file. You need to have already defined your printers to DataHub; see Appendix E, “Defining Printers to DataHub” on page 356 for more information.

You can change the settings for warning, danger, and the updating interval by clicking on the **Configure** push button. For information on changing the settings in the Disk I/O Configuration Window, see the online help.

For AIX, to open a window that shows how disk space is allocated:

1. On the object tree, double-click on the host you are interested in. The Display Types window opens.
2. From the newly opened window, select **Disk** and click on the **OK** push button.
3. On the object tree, select the disk occurrence you are interested in.
4. Click on the **Disk Layout** push button to see a graphical layout of disk space allocation.

   In this window, you can click on **Print** to print this window to a print file. You need to have already defined your printers to DataHub; see Appendix E, “Defining Printers to DataHub” on page 356 for more information.
5. Click on **OK** to close the Disk Layout window.

To display information about file systems on a host:

1. On the object tree, select the host.
2. Click on the **File Systems** push button. A window opens that displays the file systems on the host.

   You might have to size the window larger, as well as use the scroll bars to see all of the columns in this window. For each file system, the table in the window shows:

   - File system name
   - Total number of blocks on the file system
   - Number of blocks free
   - Number of blocks used

- Percentage of the file system that is used
- Directory the file system is mounted on

Click on the arrow buttons to scroll horizontally or vertically through the table information.

You can display the file system information graphically by using the three chart push buttons:

Bar Chart
Line Chart
Pie Chart

See the online help for information on how to display the file system information graphically. In the windows with the graphs, you can click on **Print** to print the window to a print file. You need to have already defined your printers to DataHub; see Appendix E, "Defining Printers to DataHub" on page 356 for more information.

3 Click on the **OK** push button.

To display processes for a host:

1 Select the host.

2 Click on the **Processes** push button. The Processes window opens.

The table in this window shows information about processes, for example, the sizes of the processes. The displayed fields have different titles, based on the operating system type of the host occurrence you selected.

See the online help for information about this window.

FYI: In the table in the Processes window, the units in which Size is measured vary with host type:

For this host type:	Size is measured in:
AIX	1KB units of core image
HP-UX	Blocks of core image
Sun Solaris	Pages of the image of swappable processes in main memory

Click on the arrow buttons to scroll horizontally or vertically through the table information.

3 When finished looking at the process information, click on the **OK** push button.

To determine what users are defined to the host:

1 On the object tree, double-click on the host. The Display Types window opens.

2 From the newly opened window, select **OS User** and click on the **OK** push button. The OS user object type and the operating system user names defined on the host are added to the object tree.

You can display the groups an operating system user belongs to by double-clicking on the operating system user occurrence.

To display the processes for an operating system user

1 Select the operating system user occurrence.

2 Click on the **Processes** push button. The User Processes window opens.

   This window shows specific user processes and information associated with each process. See the online help for information.

3 Click on **OK** to close the User Processes window.

To determine what groups are defined to the host:

1 On the object tree, double-click on the host. The Display Types window opens.

2 From the newly opened window, select **Group** and click on the **OK** push button. The Group object type and any existing groups are added to the object tree.

You can display the operating system users that belong to a specific group by double-clicking on the group occurrence.

Managing Sybase Servers

Chapter 7. Managing Sybase Servers

To be able to manage Sybase servers with DataHub, you set the Sybase server definitions and add them to DataHub. The definitions are then stored in the control database and DataHub uses these definitions to manage the Sybase servers.

Managing Sybase servers with DataHub involves:

- Adding and modifying Sybase server definitions
- Stopping management of Sybase servers
- Starting Sybase servers
- Shutting down Sybase servers
- Managing remote Sybase servers
- Managing logins on a Sybase server
- Managing dump devices
- Managing languages for Sybase servers
- Managing Sybase processes
- Managing auditing for Sybase servers

Managing auditing for Sybase servers is discussed in Chapter 14.

How to display a Sybase server: The first step in a number of the following tasks is to select or double-click on a Sybase server in the object tree. Before you can perform this step, you must display the server. Here is how you do this:

1 On the object tree, double-click on the host that supports the Sybase server. The Display Types window opens.

2 From the newly opened window, select **Sybase Server** and click on the **OK** push button. The Sybase server object type and any Sybase servers supported by the host you double-clicked on are added to the object tree.

Adding Sybase Servers and Changing Descriptions

Sybase servers reside on a host and provide the access to the Sybase databases on that host. DataHub manages the Sybase servers and, through them, their databases.

Before adding the definition of a Sybase server, you need to know the Sybase server name, the $SYBASE directory for that Sybase server, and the description you want for the Sybase server.

To add the definition of a Sybase server to the control database:

1 On the object tree, double-click on the host on which you want to add the Sybase server. The Display Types window opens.

2 From the newly opened window, select **Sybase Server** and click on the **OK** push button. The Sybase Server object type is displayed on the object tree.

3 Select the Sybase Server object type.

4 Click on the **Add** push button. The Add Server window opens.

5 Fill out all of the fields in the Add Server window. For an explanation of the parameters, see the online help.

   If you want DataHub to add out-of-the-box monitoring for this Sybase server, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

6 Click on the **OK** push button of the Add Server window.

You can change the description that appears next to a Sybase server occurrence in the object tree by using the **Definition** action. You can also used the **Definition** action to add out-of-the-box monitoring to a Sybase server.

To change the description of a Sybase server occurrence:

1 On the object tree, double-click on the host that supports the Sybase server. The Display Types window opens.

2 From the newly opened window, select **Sybase Server** and click on the **OK** push button. The Sybase Server object type and the Sybase servers supported by the host you double-clicked on appear on the object tree.

3 Select the Sybase server whose definition you want to modify.

4 Click on the **Definition** push button. The Update Managed Server window opens.

5 Change the server description in the Update Managed Server window. For explanations of the parameters, see online help.

   If you want to add out-of-the-box monitoring for this Sybase server, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

Please note:

1. If you have previously added out-of-the-box monitoring for this Sybase server, and you select **Add Standard Monitoring**, DataHub will add another set of monitors.
2. If you change the other parameters in this window and then run into problems, please verify that you have entered the correct values.

Then click on the window's **OK** push button.

Stopping Management of Sybase Servers

If you stop managing a Sybase server, all of the databases defined on that Sybase server cannot be managed by DataHub.

To stop managing a Sybase server:

1 On the object tree, select the Sybase server.

2 Click on the **Stop Managing** push button. A window opens asking you if you want to stop managing the Sybase server.

3 From the newly opened window, click on the **Yes** push button to confirm.

Starting Sybase Servers

To start up a Sybase server, you need system administrator (SA) privilege on that Sybase server.

To start up a Sybase server:

1 On the object tree, select the Sybase server you want to start up.

2 Click on the **Startup** push button. The Startup Server window opens.

3 From the Startup Server window, click on either of the following two startup modes:

Master Recover
If you choose Master Recover, the startup is run in single user SA mode. This mode is usually used to rebuild the master device after a system failure.

Normal
Runs startup in multiple user mode.

4 Click on the **OK** push button of the Startup Server window.

Shutting Down Sybase Servers

To shut down a Sybase server, you need system administrator (SA) privilege on that Sybase server.

To shut down a Sybase server:

1. On the object tree, select the Sybase server you want to shut down.
2. Click on the **Shutdown** push button. The Shutdown Server window opens.
3. From the Shutdown Server window, click on either of the following two shutdown modes:

   Normal
   Sybase waits for currently executing statements to finish, then shuts down the Sybase server.

   No Wait
   Sybase immediately shuts down the Sybase server without waiting for currently executing statements to finish.

4. Click on the **OK** push button of the Shutdown Server window. A message window opens telling you that the Sybase server is stopped.

Managing Remote Sybase Servers

On a Sybase server, you can define local and remote Sybase servers that will be known to that Sybase server. You can manage these local and remote Sybase servers using DataHub. For example, you can define a local server SYB_BACKUP for database dump and load, and define a remote server for executing remote procedure call from the local server.

In DataHub, you can:

- Define local and remote servers
- Drop local and remote servers
- Display server information
- Show the logins defined on the known servers
- Create remote logins

Creating remote logins is discussed in "Remote Logins" on page 80.

To display the local and remote servers for a particular Sybase server

1. Double-click on the particular Sybase server occurrence. The Display Types window opens.

2 Select **Server** from the list and click on the **OK** push button. The Server object type and any local and remote Sybase servers for the selected Sybase server are displayed on the object tree.

To define a local or remote Sybase server for a particular Sybase server

1 Select the Server object type and click on the **Create** push button. A window opens in which you can define the server.

2 Specify the parameters in the window. For information on the parameters, see online help.

3 When finished, click on the **OK** push button.

To drop a local or remote Sybase server for a particular Sybase server

1 Select the Sybase server and click on the **Drop** push button. The Drop Server window opens.

2 Select the Drop Logins check box if you want to drop the remote logins defined on the server.

3 Click on the **OK** push button.

You can display information about a local or remote Sybase server by selecting the Sybase server occurrence and clicking on the **Information** push button.

You can display the remote logins defined on a known Sybase server by double-clicking on the Sybase server occurrence.

Managing Logins on a Sybase Server

For a Sybase server and its databases, you have users of the databases and users of the Sybase server itself. The user of a Sybase server is given a *login*. A login is the account name set for a person to user the Sybase server itself. The login does not have to be the same as the user name defined in a database for that same person.

Giving people access to the data in a Sybase server database is a two-step process. You must:

- Create a login for them in the master database. With this login, they can log in to the Sybase server.
- Create a user name for them in the particular databases you want them to access.

To display all the logins on a Sybase server:

1 On the object tree, double-click on the Sybase server. The Display Types window opens.

2 From the newly opened window, select **Login** and click on the **OK** push button. The Login object type and any logins already defined to the Sybase server appear in the object tree.

To add a new login for a Sybase server:

1 On the object tree, select the Login object type.

2 Click on the **Create** push button. The Create Login window opens.

3 Specify values for parameters in the Create Login window. You must specify at least the login name and the password, and then retype the new password.

If you do not specify a default database, the default database is the master database.

> **Tip:** Specify a default database other than the master database. This ensures that users do not create objects in the master database.

4 Click on the **OK** push button of the Create Login Window. The new login appears in the object tree.

To update information for a login:

1 On the object tree, select the login.

2 Click on the **Modify** push button. The Modify Login window opens.

3 From the Change Login window, you can change: the login's default database, language, and full name. Also, if you want to, lock the login by clicking on **Login Locked**. Then click on the **OK** push button.

To change the password for a server login:

1 On the object tree, select the login.

2 Click on **Change Password** push button. The Change Password window opens.

3 From the Change Password window, type the new password, retype the new password, and click on the **OK** push button.

A login can connect to the Sybase server after it is created. However, to perform certain tasks, such as removing logins, the login needs a Sybase server role assigned to it.

To select one or more Sybase server roles for a server login:

1. On the object tree, select the login.
2. Click on the **Role** push button. The Select One or More window opens. In this window is a list of roles defined on this Sybase server.
3. From the Select One or More window, select the roles you want for the login and click on the **OK** push button.

For more information about roles, see "Server Roles" on page 81.

Remote Logins

A *remote login* is a login which is used by someone at a remote server to connect to the Sybase server.

To display the remote logins for a particular login:

1. On the object tree, double-click on the login to which the remote logins you want to see are assigned. The Select One or More window opens.
2. From the newly opened window, select **Remote Login** and click on the **OK** push button. The Remote Login object type and any remote logins assigned to the login you double-clicked on are added to the object tree.

To create a remote login:

1. On the object tree, select the Remote Login object type.
2. Click on the **Create** push button. The Create Remote Login window opens.
3. From the Create Remote Login window, choose a remote server, specify a remote login name, and click on the **OK** push button.

You can display information on a particular remote login by selecting the remote login occurrence and clicking on the **Information** action.

You can drop a remote login by selecting the remote login on the object tree and clicking on the **Drop** push button.

Removing Sybase Server Logins

When you remove a Sybase server login, the login can no longer be used to access the Sybase server.

To remove a server login:

1. On the object tree, select the login.
2. Click on the **Drop** push button. A window opens asking you if you want to drop the login you selected.
3. From the newly opened window, click on **Yes**. The login is removed from the object tree.

Server Roles

From a Login occurrence on the object tree, you can display the server roles assigned to that particular login.

To display the roles assigned to a particular login

1. Double-click on the login occurrence. The Display Types window opens.
2. Select Server Role from the list and click on **OK**. The Server Role object type and the roles that are assigned to that login are displayed on the object tree.

You can also display the server roles available on a particular Sybase server. To do so:

1. On the object tree, double-click on the Sybase server. The Display Types window opens.
2. From the newly opened window, select **Server Role** and click on **OK**. The Server Role object type and the server roles available on the Sybase server you double-clicked on appear in the object tree.

You can display all of the logins that have a particular server role by double-clicking on the server role.

To assign server roles to one or more logins:

1. Display the Server Role object type and the server roles from the Sybase server occurrence.
2. Select the server role.

3 Click on the **Grantee** push button. The Select One or More window opens. In this window is a scrollable list of all of the logins defined on this Sybase server.

4 From the Select One or More window, select one or more logins to be assigned to the server role you selected. Then click on the **OK** push button.

You can also authorize a server role to perform certain actions on specific tables, views, or procedures. Once authorized, all of the logins with that server role are authorized to act on the objects. Setting authorizations is useful if you have an object which you want a whole group of logins to be able to act on.

To set authorizations for a server role:

1 Display the Server Role object type and the server roles from the Sybase server occurrence.

2 Select the server role.

3 Click on the **Authorization** push button. The Select One window opens. This window shows the list of Sybase databases on the Sybase server.

4 Select the database that has the objects you want and click on the **OK** push button. A window opens, displaying two columns, one for objects and the other headed by the name of the server role. If this server role is already authorized for specific objects, the object names are listed in the first column and the authorizations are listed in the column under the name of the server role. For an explanation of the authorizations, see online help.

5 To grant authorizations to use an object:

   a From the newly opened window, select the name of the server role and click on the **Add** push button. The Choose Grant Type window opens.

   b From the Choose Grant Type window, select the object to which the new authorizations are to apply. Then click on the **OK** push button. The Grant window opens.

   From the Grant window, you can specify the actions that you want to authorize your server role′s logins to perform on the selected object. Examples of such actions are select, insert, update, and delete.

   c From the Grant window, specify values for the parameters you are interested in. For explanations of the parameters, see online help.) Then click on the **OK** push button. The object is added to the Object column of the window that opened in step **4** .

6 To change authorizations that have already been granted:

a From the window that opened in step **4** , select the authorizations under the server role column and click on the **Change** push button. The Grant window opens.

From the Grant window, you can specify a different set of actions that you want to authorize your server role′s logins to perform on the selected object.

b From the Grant window, specify values for the parameters you are interested in. For explanations of the parameters, see online help.) Then click on the **OK** push button.

Managing Configuration Variables

The configuration variables for a Sybase server control various aspects of the server's memory allocation and performance. The System Administrator (SA) or System Security Officer (SSO) can reset these configuration variables to fine-tune performance and refine storage allocation.

To display the current configuration variables for a Sybase server

1 On the object tree, select the particular Sybase server and click on the **Configure** push button.

A window opens that displays all the configuration variables and, for each variable, certain information; for example, the run value (the value the Sybase server is currently running at). For more information, see the online help.

2 You can change and reconfigure the configuration values.

To change a value, select the configuration variable you want and click on the **Change** push button. A window opens in which you can enter a new value.

To reconfigure the Sybase server with new configuration values, click on the **Reconfigure** push button.

To reconfigure with override and accept the values supplied by the user, click on the **Reconfigure with Override** push button.

*For Sybase 11:*The Reconfigure command has no effect; it is included to allow existing scripts to run without modification.

Please note: For most of the configuration variables, you must run Reconfigure and restart the Sybase server to put the new values in effect.

3 Click on the **OK** push button in the Configuration window when finished.

Managing Dump Devices

A Sybase server uses logical devices to allocate operating system disk resources. This resource might be a disk partition, an operating system file, or a tape device. You specify a *dump device* to hold the contents of a database or transaction dump. You need dump devices to make backup copies of databases and transaction logs.

To display the dump device on a Sybase server:

1 On the object tree, double-click on the Sybase server. The Select One or More window opens.

2 From the newly opened window, select **Dump Device** and click on the **OK** push button. The Dump Device object type and any dump devices defined to the Sybase server appear in the object tree.

You can define dump devices on a server, display information about particular dump devices, and drop dump devices.

Before creating a dump device, you need to decide whether to use a tape or disk device and the name of the physical device.

To create a dump device:

1 On the object tree, select the **Dump Device** object type.

2 Click on the **Create** push button. The Create Dump Device window opens.

3 From the Create Dump Device window, specify the type of device, the physical name of the device, the logical name of the dump device, and the size. Then click on the **OK** push button.

To display information about a particular dump device, select the device and click on the **Information** push button.

To drop a dump device, select the device and click on the **Drop** push button.

Managing Languages for Sybase Servers

When you install Sybase, you choose a default language for the Sybase server. You can add languages so that the names of months and days are in another language, and you can specify the format for dates, such as day-month-year or year-day-month. Also, if messages in that language are installed on your Sybase server, messages appear in the language.

After you change the language, you must restart the Sybase server for the changes to take effect.

To display the languages that are defined to a Sybase server:

1. On the object tree, double-click on the Sybase server. The Select One or More window opens.
2. From the newly opened window, select **Language** and click on the **OK** push button. The Language object type and any languages defined to the Sybase server appear in the object tree.

To add a new language to a Sybase server:

1. From the object tree, select the Language object type.
2. Click on the **Create** push button. The Create Language window opens.
3. Enter the information in the fields of the Create Language window. Separate the names of months and days with commas, and do not have any blank spaces. (For more information, see online help.) Then click on the window's **OK** push button.

To update a particular language, select the language occurrence on the object tree and click on the **Update** push button.

To delete a particular language, select the language occurrence on the object tree and click on the **Drop** push button.

Managing Sybase Processes

With DataHub, you can display the processes active on a Sybase server. You can also terminate a particular process.

To display the current processes for a Sybase server:

1. On the object tree, select the server.
2. Click on the **Processes** push button. The Processes window opens.

This window displays process information. For each process id, the table displays information such as the status of the process, the login name that owns the process, and the database name associated with the process.

When you display the list of current processes, the Blocker column contains the process id of the blocking process, if there are any.

3 To show the locks on a particular process:

  a On the object tree, select the process.

  b Click on the **Locks** push button. A window opens and displays the locks for the process.

When you display the list of current processes, the Blocker column contains the process id of the blocking process, if any. You can release the block by using the **Kill** push button with the process that has the Spid that matches the value in the Blocker column.

4 To terminate a process:

  a Select the process from the Processes window.

  b Click on the **Kill** push button. A message box opens asking for confirmation.

Managing Sybase Locks

With DataHub, you can display information about locks on a Sybase server. You can also display the process for a given lock, kill the process having the lock, and display the name of the table that is locked.

To display lock information for a particular Sybase server

1 On the object tree, select the Sybase server occurrence.

2 Click on the **Locks** push button. A window opens and displays the locks on the Sybase server you selected.

For information on this window, see the online help.

3 When you are finished using this window, click on the **OK** push button.

To display the process associated with a particular lock

1 Select the lock and click on the **Processes** push button. A window opens and display the process associated with this lock.

For information on this window, see online help.

2 Click on the **OK** push button to return to the window with the lock information.

To terminate the process associated with a lock

1 Select the lock.

2 Click on the **Kill** push button. A message box opens asking if you want to kill the process.

3 Click on the **Yes** push button if you want to kill the process; otherwise, click on **No**.

To display the table that is locked

1 Select the lock.

2 Click on the **Table Information** push button. A message box opens and displays the name of the locked table.

3 Click on **OK** to close the message box.

Managing Databases

Chapter 8. Managing DB2 Databases

Managing DB2 databases with DataHub involves performing three classes of tasks:

- Tasks unique to DB2 (for example, managing DB2 instances)
- Tasks that are common to the management of most database types, but that you execute differently for each type (for example, starting and stopping)
- Tasks that are common to the management of most types, and that you execute in a similar way for each type (for example, managing database access).

This chapter describes DB2 tasks that belong to the first two classes listed above. These tasks are:

- Managing DB2 database instances
- Forcing off and quiescing application processes
- Creating DB2 databases
- Adding database definitions to the control database
- Cataloging and uncataloging databases
- Cataloging and uncataloging DCS information
- Modifying DB2 database definitions
- Restarting DB2 databases
- Stop managing DB2 databases
- Dropping DB2 databases
- Backing up and recovering DB2 databases
- Displaying and changing the configuration of DB2 databases
- Activating and deactivating DB2 Version 2 databases
- Managing histories for DB2 Version 2 databases
- Migrating data for DB2 Version 2 databases
- Explaining SQL and tables

Other chapters describe tasks that are common to DB2 and other database types, and that you execute in a similar way for each type. These tasks are:

- Managing access to databases (see Chapter 12)
- Managing security of databases (see Chapter 13)
- Managing data and data objects (see Chapter 15)
- Using watchdogs to monitor and control database activity (see Chapter 27)

Tip: In the object tree, database types are color-coded as follows:

Database Type	**Color**
DB2 Version 1	light blue
DB2 Version 2	black
INGRES	red
ORACLE	dark blue
Sybase	orange

Managing DB2 Database Instances

With the DB2 manager, you can have several instances of the database manager running on one AIX computer. For example, you might want to have two instances: one for production purposes and another for development. Each instance is considered a unique entity and can manage several databases. However, a given database belongs to only one instance.

Each instance that you manage has:

instance home directory
This is the home directory of the instance owner.

instance name
This is the name associated with the instance. It comes from the userid of the owner of the instance.

Managing DB2 database instances can involve:

- Identifying instances defined to a given host
- Starting and stopping the DB2 database manager
- Uncataloging nodes
- Cataloging and uncataloging (CPI-C)
- Cataloging and uncataloging (TCP/IP)
- Obtaining information about the applications that use an instance
- Monitoring information about instances
- Displaying and changing the configuration of an instance

How to display an instance object type and object occurrences: The first step in a number of the following tasks is to select or double-click on the instance object type or an instance. Before you can perform this step, you must display the object type or instance. Here is how you do this:

1. On the object tree, double-click on the host occurrence that has the DB2 instances. The Display Types Window opens.
2. From the newly opened window, select Instance and click on the **OK** push button. The instance object type and any DB2 instances defined to the host you double-clicked on are added to the object tree.

Identifying DB2 Instances on a Host

You can display the DB2 instances and the databases for those instances on the object tree.

To see the DB2 instances managed by DataHub on a particular host:

1. On the object tree, double-click on the host. The Display Types Window opens.
2. From the newly opened window, select Instance and click on the **OK** push button. The instance object type and any DB2 instances defined to the host you double-clicked on are added to the object tree.

FYI: You cannot add an instance directly to DataHub. An instance occurrence is added to DataHub when you add a database occurrence that belongs to that instance. When you add a database, you specify the name of the instance; at this point, DataHub adds the new instance name.

To see the databases that belong to an instance:

On the object tree, double-click on the instance occurrence. The database object type and any databases belonging to the instance you double-clicked on are added to the object tree.

Starting and Stopping the DB2 Database Manager

Cautions:

1. The database manager must be started before its databases can be accessed.
2. You cannot stop the database manager if any databases or database applications are still active.

Attention: Do not stop the instance of the database manager that has the control database. DataHub will become inoperative.

To start or stop the database manager:

1 On the object tree, select the instance of the database manager

2 Then:

- If you want to start the database manager, click on the **Start** push button.
- If you want to stop the database manager, click on the **Stop** push button.

A question box opens to ask you to confirm.

3 From the question box, click on **Yes**.

If there is no current DBM connection for this database, the Operating System Login for Database Operation window opens. Enter your name and your password, and click on the **OK** push button.

Cataloging CPI-C Node Information

You can add a Common Programming Interface for Communications (CPI-C) node entry to the node directory for a particular instance. Advanced Program-to-Program Communications (APPC) protocol is used to access the remote node.

Cataloging CPI-C node information adds a CPI-C node entry to the node directory. You must have SYSADM authority to catalog CPI-C information.

To catalog CPI-C information for an instance:

1 On the object tree, select the instance.

2 Click on the **Catalog CPIC** push button. The Catalog CPIC Node window opens.

3 From the Catalog CPIC Node window, specify values for the following parameters:

Node Name
Name of the node you want to catalog.

Destination Name
The symbolic destination name of the remote partner node, also known as the *host name* of the node where the target database resides. The host name is the name of the node that is known to the CPI-C network.

Comment
Describes the CPI-C node entry entered in the node directory. Do not use a carriage return or line feed character in the comment.

4 Press mouse button 1. The **Security Type** parameters appear in the Catalog CPIC Node window. Select one of these parameters:

None No security information is included in the allocation request sent to the partner LU.

Same A user′s name is included with the allocation request sent to the partner LU, together with an indicator that the user′s name has been already verified.

Program Both a user′s name and password are included in the allocation request sent to the partner LU.

5 Click on the **OK** push button of the Catalog CPIC Node window.

You uncatalog CPI-C node information using the DataHub **Uncatalog Node** action; see "Listing and Uncataloging Node Information" on page 95

Cataloging TCP/IP Node Information

You can add a Transmission Control Protocol/Internet Protocol (TCP/IP) node entry to the node directory for a particular instance.

Cataloging TCP/IP node information adds a TCP/IP node entry to the node directory. You must have SYSADM authority to catalog TCP/IP information.

To catalog TCP/IP information for an instance:

1 On the object tree, select the instance.

2 Click on the **Catalog TCPIP** push button. The Catalog TCP/IP Node window opens.

3 From the newly opened window, enter values for the following parameters:

Node Name
Name of the node you want to catalog.

Hostname
The symbolic destination name of the remote partner node, also known as the *host name* of the node where the target database resides. The host name is the name of the node that is known to the TCP/IP network.

Service Name
The service name of the database manager on the remote machine. This name must match the name in the corresponding entry of the /etc/services file.

Comment
Describes the TCP/IP node entry entered in the node directory.

4 Click on the **OK** push button push button of the Catalog TCP/IP Node window.

You uncatalog CPI-C node information using the DataHub **Uncatalog Node** action.

Listing and Uncataloging Node Information

Each database instance maintains one node directory. The node directory stores information about the location of remote databases the client accesses and the communications protocol used. This information is necessary to establish a connection with a remote database that was cataloged on this node.

Use the DataHub **List Node** action to display the node directory for an instance.

To display the node directory

1 Select the instance occurrence and click on the **List Node** push button. A window opens and displays the node directory for the selected instance.

2 Click on **OK** to close this window.

You can uncatalog CPI-C and TCP/IP nodes. Uncataloging a node deletes the node name from the node directory on a file system. After a remote node is uncataloged, any databases residing on the remote node can no longer be accessed from the local node.

A user must have SYSADM authority to uncatalog a remote node.

To uncatalog a node for an instance:

1 On the object tree, select the instance.

2 Click on the **Uncatalog Node** push button. The Uncatalog DB2 Node window opens.

3 From the newly opened window, type the node name and click on the **OK** push button.

Forcing Off and Quiescing Application Processes

You can force applications from the system by using the **Force App** action. You might want to force off the applications when you need to stop the database manager.

To force applications off a system:

1 On the object tree, select the instance.

2 Click on the **Force App** push button. The Force Application window opens. You can choose to force or terminate connections for all the database or for specific applications.

Attention: Do not force all Agent IDs. DataHub displays a message if you try forcing all applications.

FYI: Normally, DB2 will wait until applications have completed their operation before disconnecting. If you select **Mode Async** from the Force Application window, DB2 does not wait for application termination before disconnecting.

DB2v2 The *quiesce* command provides an administrator with a way of ensuring that all instance or database activity has been brought to a halt. You can quiesce a table space or a database.

How to display a database object type or a database: In the tasks described in the following sections, the first step you do is select or double-click on the database object type or a database. Before you can perform this step, you must display the object type or database. Here are two ways to do this:

To display a database object type along with all databases belonging to a specific instance:

1. On the object tree, double-click on the host at which the instance is installed. The Display Types Window opens.
2. From the newly opened window, select Instance and click on the **OK** push button. The window closes, and the instance object type and all DB2 instances defined to the host you double-clicked on are added to the object tree.

   If you have more than one version of DB2 (for example, DB2 Version 1 and DB2 Version 2) installed on a host, all instances for all versions are displayed in the object tree. It is recommended that you use different instance names for each version of DB2 you have installed.
3. Double-click on the instance whose databases you want to display. The database object type and all databases belonging to the instance you double-clicked on are added to the object tree.

To display a database object type along with all databases residing at a specific host:

1. On the object tree, double-click on the host. The Display Types Window opens.
2. From the newly opened window, select Database and click on the **OK** push button. The database object type and all databases residing at the host you double-clicked on are added to the object tree.

   If you display the databases this way and you have multiple instances installed for DB2, the databases displayed on the object tree include all the databases for all of the instances.

To quiesce a database or table space:

1. On the object tree, select the database.
2. Click on the **Quiesce** push button. The Quiesce Application window opens.
3. From the Quiesce Application window, specify table space or database, and then specify the names. (See online help for more information.) Then click on the **OK** push button.

If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type the user name and password and click on the **OK** push button.

Setting Monitor Switches

You can display the current settings for the database system monitor recording switches and change the settings using the **Monitor Switches** action.

To open a window to display the monitor recording switches:

1. On the object tree, select the instance occurrence.
2. Click on the **Monitor Switches** push button. The Monitor Switches window opens.
3. From the Monitor Switches window, click on the check boxes to select the information you want monitored and then click on the **OK** push button.

Displaying and Changing the Configuration of DB2 Instances

Configuration files contain parameter values that define the resources allocated to instances and individual databases. The configuration file for the instance is created when the instance is created and affects the system resources allocated to DB2 for that individual instance. Its parameters have global applicability, independent of any one database in that instance. You can change many of these parameters from the system default values to improve performance or increase capacity of the instance.

To display or change the configuration of an instance:

1. On the object tree, select the instance occurrence.
2. Click on the **Configuration** push button. A window opens and displays the parameters in the configuration file.
3. To change a parameter, select the parameter and click on the **Change** push button. Depending on the type of parameter, one of several windows opens.
4. To reset to the system default parameters, click on the **Reset** push button.
5. Click on the **OK** push button of the window that opened in step **2.**

After changing the parameters, the instance must be stopped and then started for the new parameter values to take effect.

Snapshot Monitoring

DB2v2 DB2 Version 2 snapshot monitoring provides information about an instance at a point-in-time. Use the DataHub **Snapshot** action to do DB2 Version 2 snapshot monitoring.

To do snapshot monitoring for an instance

1 Select the instance occurrence and click on the **Snapshot** action. The Snapshot window opens. This window displays a list of snapshots that you can take for the selected instance.

For descriptions of the different snapshots, see the online help for this window.

2 Click on the snapshots you want. Depending on the snapshot you choose, you might have to further specify other items (such as database names) for your snapshot.

3 When finished working with snapshots, click on the **OK** push button.

Creating DB2 Databases

When you create a new database using DataHub, DataHub creates it and automatically adds the description to the control database control tables. When you create a new database outside of DataHub and you want to manage it with DataHub, you have to use the **Add** action to add the description of the database to the control database control tables (see "Adding Database Definitions to the Control Database" on page 100).

To create a database:

1 On the object tree, select the database object type.

2 Click on the **Create** push button.

If you displayed the Database object type from an instance occurrence, the Create Database window opens.

If you displayed the Database object type directly from a host occurrence, the Select a Database Type window opens.

From the Select a Database Type window, select the database type you want and click on the **OK** push button. The Create Database window opens.

3 Specify values for the parameters in the Create Database window. (For explanations of the parameters, see online help.) Then click on the window's **OK** push button.

If you want to add out-of-the-box monitoring for the newly created database, you have to use the **Definition** action. See "Modifying DB2 Database Descriptions" on page 103 for more information.

Adding Database Definitions to the Control Database

In order for DataHub to manage an existing database, you must make the database known to DataHub by using the **Add** action.

> **Please note:** Adding a database definition does not create the database. (see "Creating DB2 Databases" on page 99)

To add a database definition:

1 On the object tree, select the Database object type.

2 Click on the **Add** push button. The Add Database window opens.

3 Specify the parameters in this window. For information on the parameters, see the online help.

If you want to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

4 Click on the **OK** push button of the Add Database window. The database and the description you wrote appear on the object tree.

Cataloging and Uncataloging Databases

You use the **Catalog DB** action to store the location and authentication information of a database in the database directory. The database can be located either on the local workstation or on a remote node.

The database directory is a file that identifies the location of one or more databases. It contains the name of the databases and their physical locations for a particular directory path. There is one of these directories for every path that contains a database. An entry is added to the directory when a database is created on the local node. However, if the database was created at another location, you need to catalog the database so it appears in the local database directory.

You also need to catalog a database when you want to give it a different alias name or to catalog a database entry that was previously deleted.

You must have SYSADM authority at the node you are cataloging on in order to catalog a database.

To catalog a database:

1 Display the Instance object type and instance object occurrences on the object tree.

2 Double-click on the instance object occurrence that you want to catalog the database for, select Database from the selection list, and click on **OK**.

The Database object type and any database object occurrences in that instance are displayed on the object tree.

3 Select the Database object type and click on the **Catalog DB** push button. The Catalog DB window opens.

4 Specify values for the parameters in the Catalog DB window. (For explanations of the parameters, see online help.) Then click on the window's **OK** push button.

Uncataloging a database deletes a database entry from the system database directory on a specific node. You must have SYSADM authority to uncatalog a database. A database is uncataloged only in the system database directory. Entries in the local database directory are deleted as part of the DROP DATABASE processing.

To uncatalog a database:

1 On the object tree, select the database.

2 Click on the **Uncatalog DB** push button. The Uncatalog DB window opens.

3 From the Uncatalog DB window, click on the **Stop Managing Database by DataHub** check box and the **OK** push button.

Cataloging and Uncataloging DCS Information

You use the **Catalog DCS** action to store information about a database in the database connection services (DCS) directory. The DCS directory is only used if DDCS/6000 is installed on your system.

This directory stores information used by DB2 to access remote databases on a host computer: the database name, its local alias, and the application that prepares requests for accessing the host.

To catalog DCS information:

1. Display the Instance object type and instance object occurrences on the object tree.

2. Double-click on the instance object occurrence that you want to catalog the database for, select Database from the selection list, and click on **OK**.

   The Database object type and any database object occurrences in that instance are displayed on the object tree.

3. Select the Database object type and click on the **Catalog DCS** push button. The Catalog DCS window opens.

4. Specify values for the parameters in the Catalog DCS window. (For explanations of the parameters, see online help.) Then click on the **OK** push button.

Uncataloging DCS information deletes the database entry from the DCS directory. You must have SYSADM authority to uncatalog DCS information.

To uncatalog DCS information for a database:

1. On the object tree, select the database.

2. Click on the **Uncatalog DCS** push button. The Uncatalog DCS window opens.

3. From the Uncatalog DCS window, enter the database alias and click on the **OK** push button.

Listing Database and DCS Information

You use the DataHub **List DB** and **List DCS** actions to list the database and DCS directories, respectively. These actions are performed on an instance occurrence.

To list the database directory for an instance occurrence

1. Select the instance occurrence and click on the **List DB** push button. The List Databases window opens.

2. Enter a database path if you want to list the databases for that path and click on **OK**. If you want to list all databases for all paths, just click on **OK**. A window opens listing the databases identified in the selected instance's database directory for the specified path.

   You can add or change a comment for a listed database.

a Select the database name and click on the **Comment** push button. The Change Database Comment window opens.

b Enter a new comment or modify the old one and click on the **OK** push button.

3 When finished with the list of databases, click on the **OK** push button.

To list the DCS directory for an instance occurrence

1 Select the instance occurrence and click on the **List DCS** push button. The List Databases window opens.

2 Enter a database path if you want to list the databases for that path and click on **OK**. If you want to list all databases for all paths, just click on **OK**. A window opens listing the databases identified in the selected instance's DCS directory for the specified path.

You can add or change a comment for a listed database.

a Select the database name and click on the **Comment** push button. The Change Database Comment window opens.

b Enter a new comment or modify the old one and click on the **OK** push button.

3 When finished with the list of databases, click on the **OK** push button.

Modifying DB2 Database Descriptions

Use the **Definition** action to change the description that appears in the object tree next to a database occurrence. You can also use this action to add out-of-the-box monitoring to a database.

To change the database description:

1 On the object tree, select the database.

2 Click on the **Definition** push button. The Change Definition window opens.

3 Change the database description. (For explanations of this window, see online help.)

If you want to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

Please note:

1. If you have previously added out-of-the-box monitoring for this database and you select **Add Standard Monitoring**, DataHub will add another set of monitors.
2. If you change the other parameters in this window and then run into problems, please verify that you have entered the correct values.

4 Click on the **OK** push button.

Restarting DB2 Databases

If an uncommitted database has been abnormally terminated, you must restart the database.

You should use the **Restart** action after an attempt to connect to a database indicates that the database must be restarted.

To restart a DB2 database:

1 On the object tree, select the database.

2 Click on the **Restart** push button.

After restarting a database, you can list all indoubt transactions by using the **Transactions** action.

To list all indoubt transactions

1 Select the database occurrence and click on the **Transactions** push button.

A window opens and displays the indoubt transactions for the database.

2 Click on **OK** to close the window.

Suspending Administration of DB2 Databases

You can suspend administration of a database by:

- Stopping the DataHub management of the database
- Removing the database from its RDBMS.

Stopping Management of Databases

When you stop management of a DB2 database, DataHub no longer manages the database. The database and its data remain in DB2.

> **Attention:** Do not stop management of the control database. DataHub will become inoperative.

You can stop managing a database at any time. To do so:

1 On the object tree, select the database you want to stop managing.

2 Click on the **Stop Managing** push button.

Dropping Databases

Using DataHub, you can physically drop a database from the RDBMS at which it resides.

> **Attention:** Do not drop the control database. DataHub will become inoperative.

To drop a database from an RDBMS:

1 On the object tree, select the database.

2 Click on the **Drop** push button.

The database is removed from the object tree.

Displaying Lock Information for DB2 Databases

The DataHub **Locks** action displays information on locks held by applications as well as internal locks.

In DB2, users should note that the types of monitored objects and quantity of monitor information you can get depends on monitor switches. The information is collected by the database manager only after a monitor switch is turned on. When the switch for lock information is turned off, only basic monitor information can be shown.

To display locks for a database:

1 On the object tree, select the database.

2 Click on the **Locks** push button. If no locks are currently held, a message stating "No locks held" is presented to the user. Otherwise, a window opens with a list of locks. Each line shows the application process associated with the lock, the type of lock, the id of the table, and other information about the lock. You can also show session information for a lock by selecting the lock and clicking on the **Info** push button.

3 Click on the **OK** push button of the window that opened in step **2.**

Monitoring Database Sessions

To display information about current database sessions, use the **Sessions** action.

To display information on sessions:

1 On the object tree, select the database.

2 Click on the **Sessions** push button. A window opens. Each row in this window refers to one active application. For each application, information such as its owner id is displayed. For explanations of the parameters, see online help.

3 If you want to terminate a session, select the row and click on the **Kill** push button from the window that opened in step **2.**

4 Click on the **OK** push button of the window that opened in step **2.**

Backing Up and Recovering DB2 Databases

You can backup a database and rebuild that database should the original database be damaged. The rebuilding of a database is called *recovery*.

Restoring a database refers to a recovery method in which the database is rebuilt using the backup copy of the database. The restore recovery method returns the database to the state it was in at the time the backup was made.

Rolling forward a database refers to a recovery method in which a database is restored from the backup and then changes to the database since the time when the backup was made are applied. The changes are applied using the database logs.

Backing Up Databases

But first: Before backing up a database, you should ensure that the database is started.

As you plan for backing up your database, you must keep in mind the recovery method to be used. You must have SYSADM or DBADM authority to back up the database.

The DB2 backup utility uses the instance name as the owner of the backup file;therefore, the instance name must have write authority to create the file in the directory specified for the backup.

To back up an database:

1 On the object tree, select the database.

2 Click on the **Backup** push button. A window opens.

3 Specify values for the parameters in the window that opened in step **2.** (For explanations of the parameters, see online help.)

> **Tip:** Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the backup process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.
>
> See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

4 Click on the **OK** push button of the window that opened in step **2.**

Restoring Databases

Before restoring a database, it is assumed that the instance is started and the database is restarted.

Restoring a database rebuilds the database to the state it was in when the backup copy was made. To apply the changes made since the backup, you must roll the database forward after restoring it. The default action using the **Restore** action is to roll the database forward.

You must have SYSADM authority to restore a database. The database must have been previously backed up.

The restore utility restores in exclusive mode. The utility will fail if any user or application is already connected to the database that will be restored.

You can restore a backup copy of a database to an existing database or a new database. For more information, see the *DB2 Administration Guide*.

To restore a database:

1 On the object tree, select the database.

2 Click on the **Restore** push button. A window opens.

3. Specify values for the parameters in the window that opened in step **2.** (For explanations of the parameters, see online help.) Then click on the window's **OK** push button.

Rolling Databases Forward

Rolling a database forward applies recent database transactions to a restored database.

Before you can roll a database forward, the following conditions must be true:

- The database has been restored successfully.
- You must have SYSADM authority.
- The database must be configured for roll-forward recovery.

To roll a database forward:

1. On the object tree, select the database.
2. Click on the **Roll Forward** push button. The Roll Forward window opens.
3. From the Roll Forward window, click on the **Roll Forward** radio button to roll the database forward.
4. Specify values for the Roll Forward window's other parameters. (For explanations of these parameters, see online help.) Then click on the window's **OK** push button.

Displaying and Changing the Configuration of DB2 Databases

The database configuration file contains parameter values that define the system resources allocated to a particular database. The file is created when the database is created, and resides where the database physically resides. Many of the parameters can be changed to improve performance or increase capacity.

To display or change configuration parameters for a database:

1. On the object tree, select the database.
2. Click on the **Configuration** push button. A window opens and displays the parameters in the configuration file.
3. To change a parameter, select the parameter and click on the **Change** push button. Depending on the type of parameter, one of several windows opens.

4. To reset to the system default parameters, click on the **Reset** push button.

5. Click on the **OK** push button of the window that opened in step **2.**

Updates to the parameters do not take effect until all applications have been disconnected from the database. Changes take effect upon the next connection.

Activating and Deactivating DB2 Version 2 Databases

You can activate and deactivate your DB2 Version 2 databases by using the DataHub actions **Activate** and **Deactivate**. These actions are valid on DB2 for AIX version 2.1.1, DB2 for HP-UX, and DB2 for Solaris.

To activate a DB2 Version 2 database

1. Select the database object occurrence on the object tree.

2. Select the **Activate** push button.

   A question box opens asking for confirmation. Select **Yes** to confirm activation.

The database is activated.

To deactivate a DB2 Version 2 database

1. Select the database object occurrence on the object tree.

2. Select the **Deactivate** push button.

   A question box opens asking for confirmation. Select **Yes** to confirm deactivation.

The database is deactivated.

Managing Histories for DB2 Version 2 Databases

To make easier the task of administering DB2 Version 2 databases, a history file for each database keeps track of the operational activities such as backups, recoveries, and table loads. In DataHub, you can perform LIST BACKUP/RECOVERY FILE and UPDATE RECOVERY FILE operations.

To perform the LIST BACKUP/RECOVERY FILE operation for a database:

1. On the object tree, select the database.
2. Click on the **List History** push button. The List History Files window opens.
3. Specify the alias for the database, and the other parameters as you want. See online help for more information.
4. Click on the **OK** push button of the List History Files window. A window opens. For explanation of this window, see the online help.
5. After you have read the information displayed in the window that just opened, click on the window's **OK** push button.

To perform the UPDATE RECOVERY HISTORY FILE operation for a database:

1. On the object tree, select the database.
2. Click on the **Update History** push button. The History File Maintenance window opens.
3. For explanations of the parameters in the History File Maintenance window, see online help. You must enter a timestamp.
4. Click on the **OK** push button of the History File Maintenance window.

Migrating Data for DB2 Version 2 Databases

In DataHub, you can migrate your database using the **Migrate** action.

To migrate data for a database:

1. Select the database and click on the **Migrate** push button. The Migrate Database window opens.
2. Enter a user name and password and click on the **OK** push button.

   If migration is successful, a message box opens confirming migration.
3. Click on **OK** to close the message box.

Creating and Using DB2 Version 2 Explain Tables

An explain tool is provided by DB2 Version 2. This tool describes the access path selection for static SQL statements in the packages stored in the system catalog tables. For more information on the explain tool, see the *Administration Guide* for DB2 Version 2.

First, you use the DataHub **Explain Tables** action to create tables to store the information used by the DB2 explain tool. Then, you use the DataHub **Explain SQL** action to use the DB2 explain tool.

To use the explain tool tables:

1 Select the database and click on the **Explain Tables** push button. The Explain Tables window opens.

In this window, you can create, purge, or drop the explain tool tables.

To create the explain tables, click on the list menu next to Explain Function and select **Create Tables**. The following tables are created:

- "explain_argument"
- "explain_instance"
- "explain_object"
- "explain_operator"
- "explain_predicate"
- "explain_statement"
- "explain_stream"

To purge the contents of these tables, select Purge Tables from Explain Function.

To drop these tables, select Drop Tables from Explain Function.

2 Click on the **OK** push button.

You use the **Explain SQL** action to explain SQL statements. Results are stored in the tables that were created with the **Explain Tables** action.

To use the explain tool for SQL statements:

1 Select the database and click on the **Explain SQL** push button. The Explain Tables DB2 Version 2.x window opens.

2 Select an explain mode from the Explain Snapshot parameter.

On Allows you to display the rows returned (if any) from your SQL statement.

Explain Does not display returned rows.

If you select On as your Explain Snapshot parameter, you can select the Display Returned Rows check box to have the rows (if any) that are returned by your SQL statement displayed in a window.

3 Enter your SQL statement in the SQL Statement parameter field.

4 Click on **OK**.

If you selected the Display Returned Rows check box, a window opens and displays the returned rows.

When DataHub is finished writing the results to the explain tables, a confirmation message is displayed in the status message line. You can then look at the contents of the tables by displaying the tables on the object tree, selecting the tables, and clicking on the **Contents** action.

Chapter 9. Managing INGRES Databases

Managing INGRES databases with DataHub involves performing three classes of tasks:

- Tasks unique to INGRES (for example, managing INGRES databases)
- Tasks that are common to the management of most database types, but that you execute differently for each type (for example, starting and stopping)
- Tasks that are common to the management of most types, and that you execute in a similar way for each type (for example, managing database access).

This chapter describes INGRES tasks that belong to the first two classes listed above. These tasks are:

- Managing INGRES installations
- Displaying the INGRES databases on a host
- Creating databases
- Adding database definitions to the control database
- Dropping databases
- Backing up and recovering databases
- Using the INGRES Verify Tool
- Generating and displaying database statistics

Other chapters describe tasks that are common to INGRES and other database types, and that you execute in a similar way for each type. These tasks are:

- Managing access to databases (see Chapter 12)
- Managing security of databases (see Chapter 13)
- Managing data and data objects (see Chapter 15)
- Using watchdogs to monitor and control database activity (see Chapter 27)

Reminder: In the object tree, database types are color-coded as follows:

Database Type	Color
DB2 Version 1	light blue
DB2 Version 2	black
INGRES	red
ORACLE	dark blue
Sybase	orange

Important: You should add the INGRES master database, iidbdb, as a DataHub managed database. If you do not add iidbdb as a managed database, you cannot perform certain actions, such as starting and shutting down INGRES databases, using DataHub.

To add iidbdb as a managed database, see "Adding Definitions for Your Existing Databases" on page 34.

Attention: To avoid damaging or corrupting the control database, be sure not to manipulate its tables directly. The tables will be automatically manipulated, as needed, when you run the install program and invoke DataHub for UNIX Operating Systems actions.

Managing INGRES Installations

If you add the INGRES master database, iidbdb, as a managed database, you can manage all the INGRES databases on the managed host as a single *INGRES installation*. An INGRES installation refers to all of the files (such as executable files or database files) that belong to an INGRES RDBMS. The value of the II_SYSTEM variable is used to uniquely identify an INGRES installation on a host.

DataHub actions such as **Startup** and **Shutdown** are applied to the INGRES installation as a whole, and act on every database in the INGRES installation.

Although it is possible to manage INGRES databases without managing the master database, some DataHub actions are only available at the INGRES installation level. Such actions include starting and shutting down INGRES databases. Managing the iidbdb database with DataHub is recommended.

Managing INGRES installations with DataHub involves:

- Identifying INGRES installations defined to a given host
- Starting and stopping INGRES installations
- Managing INGRES servers
- Managing locations for INGRES installations
- Managing locks for INGRES installations
- Managing sessions for INGRES installations

Identifying INGRES Installations

If the INGRES master database, iidbdb, is managed by DataHub, you can display the INGRES installation on the object tree.

To display the INGRES installation on the object tree

1 Double-click on the host occurrence. The Display Types window opens.

2 Select **Ingres Installation** and click on the **OK** push button. The Ingres Installation object type and any INGRES installations on that host are displayed on the object tree.

To identify what INGRES databases belong to an installation

1 Double-click on the installation occurrence. The Display Types window opens.

2 Select **Database** and click on the **OK** push button.

The Database object type and the databases belonging to this installation are displayed on the object tree.

Starting and Stopping INGRES Installations

You start and shut down INGRES installations using the DataHub **Startup** and **Shutdown** actions. All of the INGRES databases and servers in that installation are acted on.

Starting an INGRES installation

The DataHub **Startup** action starts all of the elements of an INGRES installation.

To perform the startup process, you must have a DBM connection to this INGRES installation and the user name for the connection must have the superuser permission. If you do not have a DBM connection, DataHub prompts you to log in. If you have a DBM connection that does not have the superuser permission, you must close the current DBM connection and open a new one with the superuser permission.

For information on closing and opening DBM connections, see “Working with DBM Connections” on page 13.

> **Attention:** If you startup the INGRES installation that has the control database, an implicit shut down is performed, the databases are restarted, and you lose your connection with the control database. To establish a new connection, select the **DB Connections** choice of the **File** menu and click on the **Open** push button to open a new connection.

To start up an INGRES installation on a given host:

1 Select the INGRES installation object occurrence you want to start up and click on the **Startup** push button.

DataHub issues the command to start up the INGRES installation.

2 A window opens and displays informational messages about the startup process. When the process is complete, you can click on **OK** to close this window.

Shutting Down an INGRES installation

The DataHub **Shutdown** action allows you to shut down an entire INGRES installation for reconfiguration or system shutdown. You need to log in with a user name that has the INGRES superuser permission.

> **Attention:** Do not shut down the INGRES installation that has the control database. If you shut down the INGRES installation that has the control database, DataHub will be inoperative.

To shut down an INGRES installation on a given host:

1 Select the INGRES installation occurrence you want to shut down and click on the **Shutdown** push button.

If you do not have a current DBM connection open to this INGRES installation, the Operating System Login window opens. Enter a user name that has the superuser permission and its corresponding password. Click on the **OK** push button.

2 The Shutdown INGRES Installation window opens.

3 Specify the parameters you want. For explanations of all of the parameters in this window, see the online help.

4 A window opens that shows the results of the shutdown process. Click on **OK** to close this window and return to the object tree.

Managing INGRES Servers

An INGRES server is a process for executing a request for database access. There are four types of servers: DBMS servers, star servers, communication servers, and name servers.

Displaying INGRES Servers

To display the servers for an INGRES installation

1 Double-click on the INGRES installation object occurrence, select Ingres Server from the selection list, and click on **OK**.

If you are not already connected to the INGRES installation, the Operating System Login window opens. Enter a name and password and click on **OK**.

2 The Ingres Server object type and existing INGRES server object occurrences are displayed on the object tree.

On the object tree, INGRES server object occurrences have descriptions like

```
INGRES  * 36888
STAR    * 36879
COMSVR  * 36777
IINMSRV * 33876
```

The first word is the server type.

Server type	Word in the object tree
DBMS	INGRES
Star	STAR
Communications	COMSVR
Name	IINMSRV

The second word tells which databases in the INGRES installation can be served by the server. An * means that the server can serve all databases in INGRES installation. The third word is the server id; in the preceding example, 36888 is the server id for the first server.

Starting Servers in INGRES Installations

You can use the DataHub **Start Server** action to start servers in an INGRES installation.

To start a server in an INGRES installation

1 Display the Ingres Server object type from the INGRES installation object occurrence. For information on how to do this, see "Displaying INGRES Servers" on page 116.

2 Select the Ingres Server object type and click on the **Start Server** action button. The Start INGRES Server window opens.

3 Specify the parameters you want in this window. See the online help for information on the parameters.

4 Click **OK** to begin the start process.

A DataHub window opens to show the status of the start server process. When it is finished, click on **OK** to close the status window.

Stopping Servers in an INGRES Installation

You can use the DataHub **Stop Server** action to stop a specific server in an INGRES installation. The server is stopped immediately. All open transactions are rolled back.

Only INGRES and STAR servers can be stopped.

To stop a server in an INGRES installation

Select the INGRES server object occurrence and click on the **Stop Server** action button.

Managing Locations for INGRES Installations

For an INGRES installation, a *location* points to an INGRES directory structure on a physical disk. You specify the location to point to the full UNIX path to a directory structure (such as, /u/company/ingres/...). Using different locations, you can put database files and objects in locations other than where the install defaults are. You can also store database data files, except system catalogs, in more than one location. However, checkpoint, journal, and dump files must be stored in a single location. It is recommended that you store data files in a different location and on a different disk from the checkpoint, journal, and dump files.

To display an INGRES installation's locations

1. Double-click on the INGRES installation object occurrence.
2. Select Location from the selection list and click on the **OK** push button.

The Location object type and all existing locations for the databases in the selected INGRES installation are displayed on the object tree.

You can extend the storage area for your INGRES installation by adding more locations.

To add a location for an INGRES installation

1. Select the Location object type and click on the **Locations** push button. The Locations window opens.

   You must specify the directory and name for this location. DataHub creates the directory if the Create Directories check box is selected.

   See the online help for information on all of the parameters. You can specify the types of things for which this location can be used.

2. When finished, click on **OK**.

Monitoring INGRES Installation Sessions

To display session information about INGRES installations, use the **Sessions** action.

To display information on sessions

1. Select the installation and click on the **Sessions** push button.
2. A window opens with a list of session ids. For each session id, a number of parameters are listed, such as its owner and state it is in. For descriptions of each of the parameters, see the online help.
3. When finished with this window, click on **OK**.

Displaying Lock Information for INGRES Installations

You can use the **Locks** action to display information about locks that are requested or granted for pages, databases, and tables supported by an INGRES installation.

To display locks for an INGRES installation

1. Select the installation and click on **Locks**. A window opens with a list of locks. Each line shows the user who has the lock, the session associated with the lock, and other information about the lock.

   For more details on lock information, see the online help for the Locks window.
2. When finished with this window, click on **OK**.

Identifying INGRES Databases

You can display the INGRES databases that are currently managed by DataHub in two ways: displaying them from the INGRES installation object occurrence (as described in “Identifying INGRES Installations” on page 115) or displaying all of the managed databases on a particular host.

To display INGRES databases from a particular host

1. Double-click on the host object occurrence. The Display Types window opens.
2. Select **Database** from the list and click on the **OK** push button. The Database object type and any databases defined on that host are displayed on the object tree.

Creating INGRES Databases

Each INGRES database contains the following files:

Configuration file
The configuration file contains all of the details INGRES needs to administer the database.

Data files
Data files contain user data such as tables and indexes, as well as the system catalogs.

Checkpoint files (if you checkpoint your database)
If you *checkpoint* your database, you create a static copy of the entire database as it exists at the time you take the checkpoint. These checkpoint files are used when you need to recover the database.

Journal files (if you journal your database)
Journals are dynamic records of the changes made to tables that are being journaled. You can apply these journal files to checkpoint files to recover the database to the point up to media failure.

Dump files
Dump files are records of changes to an INGRES database that occurred while you checkpoint the database. These dump files are used in recovery when the database is checkpointed online.

When you create an INGRES database, you choose the database name, whether it is a public or private database, and, if you want, the locations of the data, checkpoint, journal, and dump files.

Public and Private INGRES Databases

When you create an INGRES database, you choose whether it is a public or private one. All INGRES users can access a public database; however, they do not necessarily have access to all of its data. Access to data is controlled by privileges and authorizations.

A private database can only be accessed by its owner or users who are authorized by the INGRES superuser through the use of the INGRES accessdb utility. Unless you want to create the database on the default location or locations that already exist, you need to add a location before creating the database. For information on adding a location to an INGRES database, see "Managing Locations for INGRES Installations" on page 118.

Creating the Database

The database name must begin with an alphabetic character and can contain up to 24 alphanumeric characters, including the underscore (_)

character. While you can enter uppercase or mixed-case characters, INGRES turns them to lowercase.

To create an INGRES database

1 On the object tree, display the Database object type.

2 Select the Database object type and click on the **Create** push button.

If you displayed the Database object type directly from the host object occurrence, a window opens with a list of database types. Select **INGRES 6.4** from the list and click on the **OK** push button.

The Create Database window opens.

3 Specify the parameters you want in the window. You must specify a database name.

For detailed information on the parameters, see the online help.

4 Click on the **OK** push button to start the creation process.

A window opens and shows the progress of the creation process. You can use the **Abort** push button to stop the creation of the database.

Creating the database can take a few minutes. When the process is finished, you can close the window using the **OK** push button.

If you want to add out-of-the-box monitoring for the newly created database, you have to use the **Definition** action. See “Modifying INGRES Database Descriptions” on page 122 for more information.

Adding INGRES Database Definitions to the Control Database

A database definition must be added to the control database in order for DataHub to manage the database. If the database was created using DataHub, its definition is automatically added to the control database.

To add a database definition

1 On the object tree, select the host and display the Database object type.

2 Select the database object type and click on the **Add** push button. The Add Database window opens.

3 Fill in the parameters. See the online help for more information.

In the **Database Directory** field, you must enter the name of the directory where INGRES is installed on the host.

If you to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

4 Click on the **OK** push button. The database is displayed on the object tree.

Modifying INGRES Database Descriptions

Use the **Definition** action to change the description that appears in the object tree next to a database occurrence. You can also change the directory where INGRES is installed.

To change the database description or directory

1 Select the database and click on the **Definition** push button. The Change Definition window opens.

2 Change the parameters you want. (For explanations of this window, see online help.)

If you to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

> **Please note:**
>
> 1. If you have previously added out-of-the-box monitoring for this database and you select **Add Standard Monitoring**, DataHub will add another set of out-of-the-box monitors.
> 2. If you change the other parameters in this window and then run into problems, please verify that you have entered the correct values.

3 When finished, click on **OK**.

Modifying INGRES Journaling and Storage Structures

Journals are dynamic records of the changes made to tables that are being journaled. You can apply these journal files to checkpoint files to recover the database to the point up to media failure. Journaling means that any change to the data in the database is recorded.

Use the **Alter** action to:

- Modify journaling parameters
- Modify the database's system tables to the most appropriate storage structure for better query processing

- Optimize catalogs for the INGRES user interface products of your choice (such as VISION)

> **Tip:** You should use the **Alter** action periodically to improve database performance, especially when many tables, front-end objects, or secondary indexes are created or destroyed.

1. Select the database and click on the **Alter** push button.
2. Change the parameters you want. See the online help for more information.
3. Click on **OK** to start the alter process.

   A window opens and shows the progress of the alter process. When the process is finished, you can close the window using the **OK** push button.

Stopping Management of INGRES Databases

When you stop management of a database, DataHub no longer manages the database. The database and its data remains on the host.

> **Attention:** Do not stop management of the control database. DataHub will become inoperative. Also, do not stop management of the master database, iidbdb.

You can stop managing a database at any time. To do so:

1. On the object tree, select the database you want to stop managing.
2. Click on the **Stop Managing** push button.

   A question box opens for your confirmation.
3. Click on the **Yes** push button to confirm.

Dropping INGRES Databases

Using DataHub, you can physically drop an INGRES database from the RDBMS at which it resides.

> **Attention:** Do not drop the control database. DataHub will become inoperative. Also, do not drop the master database, iidbdb.

To drop an INGRES database from the RDBMS

1. On the object tree, select the database.
2. Click on the **Drop** push button. A question box opens asking for confirmation.
3. Click on the **Yes** push button.

Backing Up and Recovering INGRES Databases and Installations

Using the DataHub actions **Backup** and **Recover**, you can back up individual INGRES databases or entire INGRES installations. When you back up or recover an INGRES installation, all of the databases in the installation are backed up or recovered.

Backing Up INGRES Databases and Installations

But first: Before backing up an INGRES database or installation, you should ensure that it is already started.

Use the DataHub **Backup** action to create a new backup copy for the selected INGRES database or installation. If journaling is enabled for the selected database or installation, all journal entries up to the time of the backup are marked expired.

Before starting to make the backup copy, you should know the location where you are putting the checkpoint files. The DataHub **Backup** action allows you to write the backup to a tape device. For more information on specifying tape devices for backups, see the *INGRES Database Administrator's Guide*.

To back up an INGRES database or installation

1. Select the database or installation object occurrence and click on the **Backup** push button. The Backup window opens.

2. Specify the parameters in the window. For an explanation of the parameters, see online help.

   You can select whether you want the backup copy made online or offline by selecting the appropriate radio button for the **Backup Mode** parameter. If you select **Offline**, you can specify whether the backup process should wait for the databases to be free by selecting the **Wait** check box.

Suggestion: To avoid tying up your DataHub session, not using **Wait** is recommended. Alternatively, if you do not plan to continue managing the database or installation in the current DataHub session, you can select the **Wait** check box and select **Asynchronous** for the Async. Exec. field. The asynchronous process proceeds when you close the DataHub connection to the database or installation and it is not used by others.

Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the checkpointing process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.

See “Working with Finished and Asynchronous Jobs” on page 12 for information on monitoring asynchronous jobs.

Please note: When you use the **Backup** action on an INGRES installation, DataHub backs up all of the databases in the installation. To ensure there is enough space for this operation, you should remove old backup copies by selecting the Destroy Old Checkpoints check box.

For more information on making backup copies, see the <ckpdb> command in the *INGRES SQL Reference Manual*.

3. To begin the backup process, click on **OK**.

   A window opens and shows the progress of the backup process. You can use the **Abort** push button to stop the backup process.

   Backing up databases can take a few minutes. When the process is finished, you can close the window using the **OK** push button.

Recovering INGRES Databases and Installations

When you recover an INGRES database or installation, you use a backup copy of the database (or of the databases in the installation) and specify whether or not to apply the journal files to the backup copy. The journal files apply any transactions that occurred since you made the backup copy.

To recover a INGRES database or installation

1. Select the database or installation object occurrence and click on the **Recover** push button.

2. Specify the parameters in the window. See the online help for information.

3. When you are finished specifying recovery parameters, click on the **OK** push button.

Exporting and Importing INGRES Data

With the DataHub **Export** and **Import** actions you can export and import tables for an entire database, specific tables, or all of the tables owned by a specific user.

Before exporting or importing data, you need to know the directory you are exporting to or importing from.

Exporting Data

Use the DataHub **Export** action to save tables of the selected database in an operating system file. You can later import the data using the **Import** action.

To export data

1. Select the INGRES database occurrence and click on the **Export** push button. The Export INGRES Database window opens.
2. Specify the parameters you want and click on **OK**. For more information on the parameters, see the online help.

   If you do not have a current DBM connection to the INGRES installation containing the selected database, the Operating System Login window opens. Enter a user name and password and click on **OK**.

   If you selected the **Synchronous** radio button, the DataHub status window opens and displays the progress of the export process. When the process has finished, click on **OK** to close this window.

Importing Data

Importing a table is a quick way of populating the table with data. Importing is fastest when the table is empty, not journaled, not indexed, and has heap storage.

Use the DataHub **Import** action to move data from an operating system file to an INGRES database. The operating system file must have been previously created by the export utility.

To import data

1. Select the INGRES database occurrence and click on the **Import** push button. The Import INGRES Database window opens.

   For information on the parameters, see online help.

Tip: Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the checkpointing process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.

See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

2 When you are finished specifying parameters, click on **OK**.

If you do not have a current DBM connection to the INGRES installation containing the selected database, the Operating System Login window opens. Enter a user name and password and click on **OK**.

If you selected the **Asynchronous** radio button, the DataHub status window opens and displays the progress of the import process. When the process has finished, click on **OK** to close this window.

Verifying Data for INGRES Databases

INGRES provides a Verify tool for purging temporary and expired tables and performing accessibility checks. You can use the DataHub **Verify** action to invoke the INGRES Verify tool.

To run the Verify tool on an INGRES database

1 Select the database object occurrence and click on the **Verify** push button. The Verify Database window opens.

2 Select the parameters you want. See the online help for information on the parameters.

3 Unless you select the Run Silent mode, a window opens listing the results when the verification is finished.

Monitoring Database Statistics

Using the DataHub **Statistics** action, you can generate or display INGRES database statistics.

To generate or display statistics for an INGRES database

1 Select the database object occurrence and click on the **Statistics** push button. The Statistics window opens.

For an explanation of the parameters, see online help.

2. Select the statistics mode you want. To generate statistics, select the **optimizedb** toggle button. To display stored statistics, select the **statdump** toggle button.

   You can specify a file that contains the flags you want to use in the INGRES optimizedb or statdump command.

3. Select **OK** to begin the statistics process.

A window displays the progress of the statistics process (optimizdb or statdump). If you selected **optimizedb**, statistics are created and stored in the database. If you selected **statdump**, stored statistics are displayed in the window.

When the process is finished, you can close the window using the **OK** push button.

Chapter 10. Managing ORACLE Databases

Managing ORACLE databases with DataHub involves performing three classes of tasks:

- Tasks unique to ORACLE (for example, mounting ORACLE databases)
- Tasks that are common to the management of most database types, but that you execute differently for each type (for example, starting and stopping)
- Tasks that are common to the management of most types, and that you execute in a similar way for each type (for example, managing database access).

This chapter describes ORACLE tasks that belong to the first two classes listed above.

These tasks are:

- Determining what databases reside on a host
- Creating databases
- Adding databases
- Starting up and shutting down
- Mounting and opening databases
- Stopping management of databases
- Dropping databases
- Transporting ORACLE data
- Backing up and Recovering ORACLE databases

Other chapters describe tasks that are common to ORACLE and other database types, and that you execute in a similar way for each type. These tasks are:

- Managing access to databases (see Chapter 12)
- Managing security of databases (see Chapter 13)
- Managing data and data objects (see Chapter 15)
- Using watchdogs to monitor and control database activity (see Chapter 27)

Reminder: In the object tree, database types are color-coded as follows:

Database Type	**Color**
DB2 Version 1	light blue
DB2 Version 2	black
INGRES	red
ORACLE	dark blue
Sybase	orange

But first: Before using DataHub for UNIX Operating Systems to manage an ORACLE database, ensure that the initialization parameters are set correctly in the init.ora and config.ora files. For example, if you want to have the audit action recorded in the audit trail table, you need to set the AUDIT_TRAIL parameter to DB or TRUE.

Attention: To prevent any potential damages or corruption of the control database, do not directly manipulate its tables. The tables will be automatically manipulated, as needed, when you run the install program and invoke DataHub for UNIX Operating Systems actions.

FYI: Some actions on ORACLE databases (for example, **Shutdown**) require an operating system login. The first time an action of this type is performed in a DataHub session, for a particular database, the Operating System Login for Database Operation window opens and you must enter an operating system login and password. This creates a DBM connection for that database. If you perform subsequent actions that require the operating system login, DataHub will not request the login again but will use the existing connection. To see what DBM connections are active, use the **DBM Connections** choice in the **File** pull-down menu (see also "Working with DBM Connections" on page 13 for more information).

The operating system login must have the privileges to perform the action. If the operating system login does not have the necessary privileges, you must close the operating system login connection using **DBM Connections** and open a connection using a login that has the necessary privileges.

Creating Databases

When you create a new database using DataHub, DataHub creates it and automatically adds the description to the control database control tables. When you create a new database outside of DataHub and you want to manage it with DataHub, you have to use the **Add** action to add the description of the database to the control database control tables (see "Adding ORACLE Database Definitions to the Control Database" on page 132).

To create a database

1 On the object tree, display the database object type.

2 Select the database object type and click on the **Create** push button.

3 Select the database type you want from the list and click **OK**. A Create Database window opens.

4 Specify the parameters in the window. For information on the parameters, see the online help or the *ORACLE Server Administrator's Guide*.

FYI:

1. Adding rollback segments in this window does not create them. If the rollback segments do not already exist, you must create them before shutting down the database. Otherwise, unexpected results might occur. See "Managing Rollback Segments" on page 136 for information on creating rollback segments.
2. DataHub only creates the system table space. When the database is created, you have to create additional user or rollback segment table spaces and create user rollback segments. See "Creating Table Spaces" on page 205 and "Managing Rollback Segments" on page 136.

Tip: Archiving set Off is recommended at creation. After database creation, set archiving On using the **Archive Log** action push button.

5 When finished specifying parameters, click on **OK**.

After you create the database, you must make sure the rollback segments that you specified actually exist before the first shut down of this database. You also must create additional user and rollback segment table spaces and user rollback segments.

If you want to add out-of-the-box monitoring for the newly created database, you have to use the **Definition** action. See "Changing ORACLE Database Descriptions" on page 132 for more information.

Adding ORACLE Database Definitions to the Control Database

For DataHub to manage an existing database, you must make the database known to DataHub by using the **Add** push button.

> **Please note:** Adding a database definition does not create the database. (See "Creating Databases" on page 131)

To add a database definition

1. Select the database object type and click on the **Add** push button. The Add Database window opens.
2. Enter the database name, the directory, a description, and select the release (ORACLE). See the online help for more information.

   If you want to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.
3. Click on the **OK** push button. The database is displayed on the object tree.

Changing ORACLE Database Descriptions

You can change the description that appears next to an ORACLE occurrence in the object tree by using the **Definition** action.

To change a database description

1. Select the database and click on the **Definition** push button. The Change Definition window opens.
2. Change the database description. See the online help for more information.

   If you want to add out-of-the-box monitoring for this database, select the **Add Standard Monitoring** check box. See "Out-of-the-Box Monitoring" on page 293 for more information.

Please note:

1. If you have previously added out-of-the-box monitoring for this database and you select **Add Standard Monitoring**, DataHub will add another set of monitors.
2. If you change the other parameters in this window and then run into problems, please verify that you have entered the correct values.

3 Click on the **OK** push button.

Renaming Databases

Use the **Rename** action to change the global name of a database.

To rename a database

1 Select the database and click on the **Rename** push button. The Global Database Name window opens.

2 Type the new database name. The Domain name is optional.

3 When finished, click on **OK**. A message box opens confirming the change. Click on **OK** to close the message box.

Starting Up and Shutting Down Databases

To open a window from which you can start up a database on a given host

1 Select the database you want to start up and click on the **Startup** push button. The Startup window opens. From this window, fill in the following parameters

Mount

Click on the radio buttons to choose Exclusive, Parallel, or No Mount

Exclusive One instance can access the database

Parallel Multiple instances can access a single database

No Mount Starts the instance without mounting

No Mount is usually used during database creation.

Open

Click on the radio button to have the database open as part of starting up.

Restricted
Click on the radio button to have the database open but restricted. Users who have the RESTRICTED SESSION privilege can connect to the database after it is started.

Recover
Click on the radio button to perform media recovery.

Force
Click on the radio button to require a forced start of the database.

For more information, see the online help.

2. Click on the **OK** push button in the Startup window.

   If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

3. A window opens that shows the results of the start up operation. Click on **OK** to close this window and return to the object tree.

To open a window from which you can shut down a database on a given host

1. On the object tree, display the databases on the host.
2. Select the database you want to shut down and click on the **Shutdown** push button. The Shutdown window opens.
3. Select a shutdown mode:

   Abort Instantaneous shutdown. Disconnects all connected users and uncommitted transactions are not rolled back.

   Immediate Immediate shutdown. Disconnects all connected users and uncommitted transactions are rolled back.

   Normal ORACLE waits for all connected users to disconnect.

   For more information, see the online help.

4. Click on the **OK** push button in the Shutdown window.

   If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

5. A window opens that shows the results of the operation. Click on **OK** to close this window and return to the object tree.

Mounting ORACLE Databases

Some database operations, such as renaming datafiles, require a database to be started and mounted but closed. Other users cannot access the database.

Before mounting, the database should have been started with the No Mount option. See "Starting Up and Shutting Down Databases" on page 133 for more information.

To mount a database

1 Select the database occurrence and click on **Mount**. The Mount Parameters window opens.

2 Click on a radio button to select Mount Exclusive or Mount Parallel and click on **OK**.

If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

3 A window opens that shows the results of the operation. Click on **OK** to close this window and return to the object tree.

Opening Databases

When a database is mounted but not opened, you can open the database. When it is opened, the database can be accessed by one or more users.

To open a mounted database

1 Select the database and click on **Open**. The Open Parameters window opens.

2 Click on a radio button to select the Open Parameter:

None

No open parameters

Resetlogs

Use after media recovery. The current log sequence number is set to one. All redo entries in the online and archived redo log files are invalidated. After opening a database with this parameter, you need to do a complete database backup.

No Resetlogs
The log sequence number is not altered and the archived redo log files are not invalidated.

3 Click on **OK**.

If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

4 A window opens that shows the results of the operation. Click on **OK** to close this window and return to the object tree.

Managing Rollback Segments

A rollback segment is used by ORACLE to record rollback information. Rollback information is a record of those database actions that should be undone when an uncommitted transaction occurs.

Managing rollback segments involves:

- Getting information about a rollback segment on a database
- Creating and deleting
- Updating and setting online or offline

To get information about a rollback segment on a database

1 Double-click on the database, select **Rollback Segments** from the list, and click on **OK**. The Rollback Segment object type and any existing rollback segments are displayed on the object tree.

2 Select the rollback segment occurrence you want to get information on and click on **Segment Info**. The Segment Info window opens and displays information about the rollback segment.

3 When finished, click on **OK**.

To create a rollback segment

1 Select the Rollback Segment object type and click on **Create**. The Create a new Rollback Segment window opens. The following parameters must be specified:

Rollback Segment Name
Type the name of the rollback segment.

Public

Click on the radio button to specify whether the rollback segment is made available to any instance.

The other parameters refer to the storage settings of the rollback segment. You can accept the default values of the other parameters or specify values. If you want to specify the values of the storage settings, select each field and type in the values.

Table spaces must be online to create a rollback segment in that table space.

2 When finished, click on **OK**. The new rollback segment is displayed on the object tree. When a rollback segment is created, its status is set to offline.

To delete a rollback segment

1 Select the rollback segment and click on **Drop**. A question box opens to confirm dropping the rollback segment.

2 Click on **Yes**. A message box opens to confirm editing of the relevant `init.ora` files to delete the rollback segment.

3 Click on **OK**.

To alter or set a rollback segment online or offline

1 Select the rollback segment and click on **Alter**. The Alter Rollback Segment window opens.

In this window, you can change the following parameters by selecting the field and typing.

Next Extent (Bytes)

The size of the next extent.

Maximum # Extents

The maximum number of extents for the rollback segment.

In this window, you can also change the status (online or offline) of the rollback segment. To change the status, click on the Status button and select the status you want.

2 When finished, click on **OK**.

Stopping Management of ORACLE Databases

When you stop management of a database, DataHub no longer manages the database. The database and its data remains on the host. You can stop managing a database at any time. To do so:

1. On the object tree, select the database you want to stop managing.
2. Click on the **Stop Managing** push button.

Dropping Databases

Using DataHub, you can physically remove a database from the host on which it resides. The database and its data are erased from the host.

To remove a database from the RDBMS on a given host

1. On the object tree, select the host and display the database.
2. Click on the **Drop** push button. The Select Files to Delete window opens. The window displays a list of physical files (except config.ora) for the selected database.

   You can optionally remove one or more of these files by selecting them in the list.
3. Click on the **OK** push button in the Select Files to Delete window.

Displaying Lock Information

In a multi-user RDBMS, problems associated with data concurrency, integrity, and consistency are solved by using locks. For example, putting a lock on a table prevents incorrect updates and alterations of that table when the table is shared.

The **Locks** action displays information on locks held by applications as well as internal locks.

To display locks for a database

1. Select the database and click on **Locks**. A window opens and displays the locks (if any) existing on the database. Each line identifies the session and lists information about a lock, such as the type of lock. For information on this window, see the online help.

   You can also display a system information for a lock by selecting the lock and clicking on **Session Info**. For more information, see online help.

   You can terminate a particular lock by selecting the lock and clicking on **Kill**.
2. When finished, click on **OK**.

Monitoring ORACLE Database Sessions

To display information about current ORACLE database use the **Sessions** action.

You can also show the locks for each session as well as kill a session or process.

To display information on sessions

1. Select the database and click on **Sessions**.
2. A window opens with a list of session ids. For each session, a number of parameters are listed, such as its owner, status, and host name. For descriptions of each of the parameters, see the online help.

   You can show the locks for each session by selecting the session and clicking on the **Show Locks** push button.

   You can kill a session by selecting the session and clicking on **Kill**.
3. When finished, click on **OK**.

Monitoring Database Statistics

Using DataHub, you can display certain database statistics for an ORACLE database. You can display statistics for:

- Resource costs
- Reads and writes for data files
- User sessions
- Redo log

Displaying statistics for data files is discussed in "To display file input and output (File I/O)" on page 210.

Displaying Resource Costs

Use the **Resource Costs** action to display the total resource costs used in a session for an ORACLE database. The costs of the session can be limited by the value of the composite limit parameter in the user's profile.

To display the total resource cost

1. Select the database occurrence and click on **Resource Costs**. The Profile Cost window opens. In this window are listed values for certain costs for this session. See the online help for descriptions of these costs.

2 If you want to accept the values shown in the window, click on **OK**. If resource limitation enforcement is disabled, a question box opens and asks you if you want to enable it. Click on **Yes** to enable, **No** to leave as is.

3 If you want to exit the window, click on **Cancel**.

Displaying User Statistics

Using the **User Stats** action, you can display statistics from user sessions.

To display user statistics

1 Select the database occurrence and click on **User Stats**. The Select Statistics window opens; it has a scrollable list of user statistics.

2 From the list, select the statistics you want monitored and click on **OK**. The User Statistics window opens.

This window displays the selected user session statistics in a table and histogram. The table provides one line for each measured statistic.

As you look at this window, your cursor changes from an arrow to a watch as DataHub refreshes the statistics. The statistics are refreshed frequently; before selecting a row or push button, wait until DataHub finishes refreshing and your cursor is displayed as an arrow.

For more information on this window, see the online help. You can see the history for a statistic by selecting the statistic row and clicking on the **History** push button.

3 When you are finished, click on **OK**

Displaying Redo Log Statistics

Using the **Redo Stats** action, you can display statistics from the redo log.

To display redo log statistics

1 Select the database occurrence and click on **Redo Stats**. The Select Statistics window opens with a list of the statistics that can be monitored.

2 Select the statistics you want and click on **OK**. The Redo Statistics window opens.

This window displays the selected redo log statistics in a table and histogram. The table provides one line for each measured statistic.

For more information on this window, see the online help.

3 When you are finished, click on **OK**.

Transporting ORACLE Data

With the Export and Import utilities, you can move existing data in ORACLE format to and from an ORACLE database. You can export and import specific tables, data for a specific user, or the data for an entire database. The utilities get a single point-in-time image of the data and then reconstruct the images in a physical file (export) or in a database (import).

Exporting Data from ORACLE Databases

Use the DataHub **Export** action to export data from an ORACLE database to a physical file. The file must reside on the same host that the database is on.

> **FYI:** In ORACLE, export files for an entire database are owned by the database ID that performs the action.

To open a window for specifying export parameters and begin the export process

1 Select the database and click on the **Export** push button. The Export window opens.

For an explanation of the parameters in this window, see the online help.

You must enter the name of the file to contain the exported data and the Operating System name and password.

> **Tip:** You can use the **Export Mode**, **User**, and **Table** parameters to back up:
>
> - A complete database.
>
>   To proceed, select the **Full database** option of the **Export Mode** parameter.
>
> - All tables owned by one or more users.
>
>   To proceed, leave the **User** option of the **Export Mode** parameter as is (selected). **User** is the default. Next, click on **List** beside the **User** field. From the window that subsequently opens, select the user or users whose tables you want to back up.
>
> - Selected tables owned by one or more users.
>
>   To proceed, select the **Tables** option of the **Export Mode**. Next, click on **List** beside the **Table** field. From the window that subsequently opens, (1) select the user or users whose tables you want to back up, and (2) click on the **OK** push button. From the window that opens next, select the table or tables you want to back up.

Tip: Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the export process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.

See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

2 Click on **OK** to start the export process.

Importing Data to ORACLE Databases

Use the DataHub **Import** action to import data from a physical file to an ORACLE database. The file must reside on the same host that the database is on.

To open a window for specifying import parameters and begin the import process

1 Select the database and click on the **Import** push button. The Import Parameters window opens.

For an explanation of the parameters, see online help.

You must specify the name of the physical file. You can accept the default values or specify the parameter.

Tip: Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the import process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.

See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

2 Click on the **OK** push button to start the import process.

Backing Up and Recovering ORACLE Databases

Backup and recovery of databases in ORACLE involve:

- Backups of the data files that make up the ORACLE database
- The redo log (both the archived and online redo log)

- Control files (which monitor and control all of the other database files)
- Initialization and configuration parameter files

Backing Up ORACLE Databases

In general, for ORACLE databases, you can have the following types of backups:

- Full database backup. A full backup is performed when the database is shutdown (offline).
- Partial backup. In a partial backup, the data files of one or more tablespaces are backed up. The tablespaces can be online or offline.

This section assumes that you have already determined the backup and recovery strategy for your ORACLE database.

Using DataHub's **Backup** action, you can back up:

- the control file
- the parameter files
- the full database (offline)
- table spaces (online or offline)
- archived redo log files

With the **Backup** action, you can backup the database files to disk or tape. The DataHub **Backup** action uses the archiving program you specify to create an archive on disk or tape.

DataHub also lets you store the specification of your particular backup scenario; you can then schedule the execution of the actual backup. You can also alter and delete backup specifications.

To perform a backup of your ORACLE database with DataHub, you first specify the type of backup you want (for example, full database backup offline or just backup of the control file). DataHub stores this information as a *backup specification*.

To complete a backup, you then schedule it for execution. When you schedule the backup specification for execution, DataHub generates a backup script that will be executed when the backup is performed. See "Scheduling Backups for Execution" on page 146 for more information.

Working with ORACLE Backup Specifications

In the backup specification, you choose

- What to back up (for example, just the control file)
- Whether online or offline backup

- Location of the backup files

To work with DataHub backup specifications

1 Select the database object occurrence on the object tree and select the **Backup** action. A window in which you can work with backup specifications for this database opens. From this window, you can:

- Create a new backup specification
- Alter an existing backup specification
- Delete an existing backup specification
- Schedule for execution a particular backup specification

If you have backup specifications already for this database, they are listed in the table by Backup Id.

The following sections describe how to work with your backup specifications.

Creating Backup Specifications

You create a new backup specification by selecting the **Create Backup Specification** push button.

In the Backup window, for all types of backups, you must specify

- A backup description
- A directory where you want DataHub to write the output from executing the backup script
- Target destination for the backup copy
- Type of archiving program you want to use
- User name and password for the connection used to backup the files at the target destination

You also select the type of backup you want. If you select

- A Full Database Backup Offline, DataHub backs up:
  - All control files
  - All data files
  - Online redo log files
  - All specified parameter files
- A Partial Database Backup, depending on what you specify, DataHub backs up any combination of the following files:
  - The current control file
  - All data files of the specified table spaces
  - All archived redo log files
  - All specified parameter files

To create a new backup specification

1 Select the **Create Backup Specification** push button. The Backup window opens.

For a description of the parameters, see online help.

If you want the parameter files backed up, as part of a full or partial backup or by themselves, you must specify the parameter file names.

2 Specify the Backup Mode you want.

If you select Partial Database Backup, you can specify any combination of:

- Backup Control File. If you select this check box, DataHub backs up the current control file. In the Backup Control File Name field, you must specify the full name you want for the backup of the control file.
- Backup Table Space Mode. DataHub backs up all data files of the table spaces you select. When backing up online table spaces, DataHub does the marking of the beginning and end of the backups for you.
- Backup Archived Redo Log Files. If you select this check box, all the archive files at the location specified by the LOG_ARCHIVE_DEST initialization parameter are backed up.

If you select Full Database Backup Offline, upon execution of the full backup, DataHub will immediately shut down the database, restart it in restricted mode, and shut it down in normal mode.

3 Specify the required parameters, as described under "Creating Backup Specifications" on page 144.

To see the backup script that DataHub generates from the parameters you specify in the Backup window, click on the **Show Script** push button. This script is generated when you schedule your backup.

4 When you are finished specifying your backup parameters, click on the **OK** push button to store this backup specification.

Altering Backup Specifications

You can alter the parameters specified in an existing DataHub backup specification.

To alter an existing backup specification

1 In the backup specification list, select the Backup Description of the backup specification you want to change.

2 Select the **Alter Backup Specification** push button. The Backup window opens.

3 Change the parameters you want. See online help for a description of the parameters.

4 When finished, click **OK** to store the modified backup specification.

Deleting Backup Specifications

You can delete backup specifications that are no longer needed.

To delete an existing backup specification

1 In the backup specification list, select the Backup Description of the backup specification you want to delete.

2 Select the **Delete Backup Specification** push button.

The selected backup specification is deleted.

Scheduling Backups for Execution

When you schedule a backup specification for execution, DataHub generates the backup script that will be executed. Any data files that have been added to the database after you created a full backup specification are included in the backup script when the backup specification is scheduled. Any data files that are no longer part of the database when the full backup specification is scheduled are not referenced by the backup script. Any data files that have been added to a tablespace after you created a partial backup specification that includes that tablespace are included in the backup script when the partial backup specification is scheduled.

DataHub generates the backup script when you select the **Execute Backup Specification** push button.

To schedule a particular backup to be performed

1 In the backup specification list, select the Backup Description of the backup specification you want to schedule.

2 Select the **Execute Backup Specification** push button. The Specify Backup Time window opens.

3 Specify when you want the backup to be performed. You can either specify it for immediate execution or specify particular times. See the online help for information on the parameters in this window.

4 When finished, click on **OK** to complete scheduling of the backup; otherwise, click **Cancel** to close the window without scheduling the backup.

Execution of the backup script is scheduled according to the times you specified in the Specify Backup Time window. If you selected Backup Now, DataHub executes the script immediately. If you entered date and time information, DataHub:

- Saves the generated backup script.
- Adds an entry in the crontab file for the operating system user associated with the backup specification. This entry tells the cron utility when to execute the saved backup script.

Recovering ORACLE Databases

Recovery of a database involves two sequential tasks: restoring the database files from a backup copy and then recovering the database by applying the redo log files to the backup copy. You can recover your managed ORACLE databases using DataHub′s **Recover** action.

For ORACLE databases, DataHub supports the following types of recovery:

- Full database recovery.
- Partial database recovery. DataHub can recover one or more data files. If you need to restore the parameter files or control files, you must copy these files from the backup media before you begin recovery with DataHub.

The database must be closed during full or partial database recovery using DataHub. You should shutdown the database before starting the recovery. DataHub issues a startup mount command prior to recovering a database or data files. DataHub recovery is always automatic. The archived redo log files must be in the location specified by the LOG_ARCHIVE_DEST initialization parameter, the archived redo log file name format must agree with the value specified in the LOG_ARCHIVE_FORMAT initialization parameter.

If a DBM connection already exists for the database object occurrence you are recovering, the userid and password combination for this connection must have the authority to write to the database files that are being restored. If not, you must close the existing DBM connection and open one using a userid and password combination that has the correct authority. For information on opening and closing DBM connections, see “Working with DBM Connections” on page 13.

To recover an ORACLE database

1 Select the database object occurrence in the object tree and click on the **Recover** action. The Recover Database window opens.

For information on the parameters in this window, see online help.

To recover a full database:

a Select the **Full Database** toggle button.

b Specify one of the following options:

- Time based recovery
- Change based recovery
- Complete recovery

In a change-based recovery, the database is recovered to a transaction consistent state immediately prior to the specified system change number. In a time-based recovery, the database is recovered to the specified timestamp. After a change-based or time-based recovery, DataHub opens the database, resets the log sequence number, and then shuts down the database normally.

To recover individual data files:

a Deselect the **Full Database** toggle button. The **Data Files** parameter becomes available.

b Specify the data files in the **Data Files** parameter using the **Add** push button. See online help for more information. During recovery, the specified files are restored from the backup and then recovered by application of the redo log files.

2 Specify where the backup copy files are stored by selecting the **Storage Parameters** push button. The Recovery Storage window opens.

If the backup copy was made by DataHub and the backup specification still exists, you can use that specification in the recovery process. Select the **Stored Specification** push button to use the DataHub backup specification.

If the backup copy was made with another backup tool, specify the storage parameters in the Recovery Storage window. See the online help for more information.

To see the contents of the files at the specified location, click on the **Contents** push button.

After you have specified the recovery storage parameters, click on the **OK** push button to return to the Recover Database window.

3 To see the recovery script that DataHub generates when it begins the recovery process, select the **Show Script** push button.

4 When finished specifying parameters in the Recover Database window, select the **OK** push button to begin the recovery process.

Managing Redo Log Files

Redo log files store transaction information. They are used for recovering from instance or media failure.

Together, all of the redo log files for a database are called the *redo log*. The redo log records all changes made to the database.

The redo log can be composed of two parts: the online redo log and the archived redo log. The online redo log is a set of two or more online redo log files that record all committed changes made to the database. When online redo log files are filled, they can be archived before being reused. The set of archived redo log files is the archived (or offline) redo log.

Archiving the redo log is determined by the mode that the database is using: ARCHIVELOG or NOARCHIVELOG.

Managing redo log files involves:

- Creating
- Adding
- Switching
- Renaming
- Deleting

Redo log files are created when the associated database is created; see "Creating Databases" on page 131.

> **Please note:** In the DataHub window, ORACLE redo log files are called *log files* and redo log groups are called *logfile groups*.

Adding Redo Log Files

You can add redo log files to a redo log group until you reach the maximum number of redo log files allowed for that group. For more information on redo log groups, see "Managing Redo Log Groups" on page 151.

To add a redo log file

1. Double-click on the database occurrence, select **Logfile Groups**, and click on **OK**. The Logfile Group object type and any redo log group occurrences are displayed on the object tree.

2. Double-click on the redo log group occurrence. The Log File object type and the redo log file occurrences in that group are displayed on the object tree.

3. Select the Log File object type and click on **Create**. The Add Logfile window opens.

4. Type in the name of the redo log file and click on **OK**. The new redo log file is added to the object tree.

Renaming Redo Log Files

Renaming redo log files updates the entry in the database control file; the physical redo log file is not changed.

Before renaming online redo log files, make sure that the new online redo log files already exist. You might want to backup the database and copy the redo log files to their new location.

To rename a redo log file

1. In the object tree, select the redo log file occurrence you want to rename and click on **Rename**.

2. Type the new file name and click on **OK**.

   If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

After renaming or relocating a redo log file, you might want to backup the database's control file.

Deleting Redo Log Files

Make sure the redo log group associated with the log file is archived before dropping the redo log file.

To delete a redo log file

1. Select the redo log file occurrence you want to delete and click on **Drop**. A question box opens asking you to confirm dropping the file.

2. Click on **Yes**.

   If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

3 The log file is deleted.

Switching Redo Log Files

Use the **Switch Logfile** action to make the currently active redo log group inactive. You might want to use this to do online redo log maintenance operations.

> **Please note:** The **Switch Logfile** action actually switches the redo log group. Redo log files in the same redo log group have identical data, that is used primarily for data protection.

To force a log switch

1 Select the database and click **Switch Logfile**. A message box opens confirming that the database is now using the next redo log group.

2 Click on **OK**.

Managing Redo Log Groups

The online redo log of an ORACLE database usually consists of groups of online redo log files. In the DataHub window, these groups are called *logfile groups*.

Managing redo log groups involves:

- Displaying the redo log groups on a database
- Displaying the redo log files in a particular redo log group
- Creating a redo log group
- Deleting a redo log group

To display the redo log groups on a database and the redo log files in the group

1 Double-click on the database, select **Logfile Group** from the list, and click on **OK**. The Logfile Group object type and the existing redo log groups are displayed on the object tree.

2 To display the redo log files in a group:

Double-click on the group The redo log file occurrences are displayed on the object tree.

To create a redo log group

1 Select the Logfile Group object type and click on **Create**. The Add Logfile Group window opens.

You must provide values for:

Log File Type the name of the redo log file for this group.

Size (K/M) Type in a size.

Logfile Reuse Click on the radio button to specify whether or not to reuse the redo log file in this group. **No Reuse Logfile** is the default.

2 When finished, click on **OK**. A new group is added to the database and displayed on the object tree. If you double-click on the new group, the redo log file you specified in the Add Logfile Group window is displayed.

To delete a redo log group

1 Select the redo log group you want to drop and click on **Drop**. A question box opens asking for confirmation.

2 Click on **Yes**.

Archiving Redo Information

Use **Archive Log** and **No Archive Log** actions to set the archiving mode of a database to archive or no archive. If you use the archive log mode, you can recover all committed transactions in the event of a database failure. Also, the entire database can be open while you back up or recover the database. If you use the no archive log mode, you can only restore the database to the point of the most recent full backup in the event of a database failure. Also, in no archive log mode, you cannot do online table space backups.

Usually, a database is in no archive log mode at creation. After the database is created, you can change the archive mode.

To go from no archive log mode to archive log mode, you should first shutdown and backup the database.

To set archive log mode

1 Mount the database exclusively and do not open.

2 Select the database and click **Archive Log**.

If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

3 A window opens to display the output for this operation. Click on **OK** to close the window.

To set no archive log mode

1 Mount the database exclusively and do not open.

2 Select the database and click **No Archive Log**.

If there is no current DBM connection to this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privileges to perform this action and click on **OK**.

3 A window opens to display the output for this operation. Click on **OK** to close the window.

Managing ORACLE Database Links

A *database link* is a named object that describes a “path” from one database to another. They are usually used in a distributed ORACLE database environment to facilitate connections between individual databases. Database links are implicitly used when a reference is made to a global object name in a distributed database.

To display the database links for a database

Double-click on the database occurrence, select **Database Links** from the list, and click on the **OK** push button. The Database Links object type and any existing database links are displayed on the object tree.

To create a database link

1 Select the **Database Links** object type and click on **Create**. A window opens in which you can enter parameters for the link. See the online help for information on the parameters.

2 Click on **OK** when finished.

To display information about a particular database link

1 Select database link and click on the **DB Link Info** push button. A window opens that displays information such as the owner of the database link and when it was created.

2 Click on **OK** to close this window.

To delete a database link

1 Select the database link and click on the **Drop** push button. A question box opens asking for confirmation.

2 Click on **Yes** to confirm.

Chapter 11. Managing Sybase Databases

Managing Sybase databases with DataHub involves performing three classes of tasks:

- Tasks unique to Sybase (for example, dumping a Sybase database)
- Tasks that are common to the management of most database types, but that you execute differently for each type (for example, showing processes)
- Tasks that are common to the management of most types, and that you execute in a similar way for each type (for example, managing database access).

This chapter describes Sybase tasks that belong to the first two classes listed above. These tasks are:

- Determining what databases reside on a Sybase server
- Creating databases
- Altering databases
- Removing databases
- Displaying locks
- Displaying processes
- Backing up and recovering databases

Other chapters describe tasks that are common to Sybase and other database types, and that you execute in a similar way for each type. These tasks are:

- Managing access to databases (see Chapter 12)
- Managing security of databases (see Chapter 13)
- Managing data and data objects (see Chapter 15)
- Using watchdogs to monitor and control database activity (see Chapter 27)
- Adding standard monitors to databases (see “Out-of-the-Box Monitoring” on page 293)

Reminder: In the object tree, database types are color-coded as follows:

Database Type	Color
DB2 Version 1	light blue
DB2 Version 2	black
INGRES	red
ORACLE	dark blue
Sybase	orange

Attention: To avoid damaging or corrupting the control database, be sure not to manipulate its tables directly. The tables will be automatically manipulated, as needed, when you run the install program and invoke DataHub for UNIX Operating Systems actions.

Tip: If another user shuts down and restarts the Sybase server while you are using a Sybase database on that server, you might receive the following message:

SQL-statement for database Sybase returned code ′20047′;
′Unexpected EOF from SQL Server.
DBPROCESS is dead or not enabled.
DBPROCESS is dead or not enabled.′

If you receive this message, you can restore the connection between your database and the server by:

1 From the DataHub window, select the **Connections** choice in the **File** pull-down menu. The Connections window opens.

2 From the Connections window, select the Server and click on the **Open** push button. The Managed Database Connect window opens.

3 In the Managed Database Connect window, click on the **List** push button and select the Sybase server you want from the list.

4 Enter a user name and password and click on the **Connect** push button.

Creating Sybase Databases

A Sybase server uses logical devices to allocate operating system disk resources to databases. In DataHub, these devices are called *database devices*. When the Sybase server is first configured, the database device name is mapped to the actual physical disk resource on the host machine. This resource might be a disk partition or an operating system file.

Before you create a database on a particular device, that device has to exist for the Sybase server. When the Sybase server is first installed, Sybase creates a master device and loads the Sybase required databases on that device. It is not a good idea to install user databases on this master device. You should create database devices for the databases you want to create.

Managing Database Devices

A *database device* stores the objects that make up a Sybase database. Each database device or file must be prepared and defined to the Sybase server before it can be used for database storage.

Managing database devices with DataHub involves:

- Displaying the database devices on a particular Sybase server
- Displaying the databases mapped to a particular database device
- Displaying information about a particular database device.
- Creating and dropping database devices
- Making a database device a default device
- Mirroring, unmirroring, and remirroring database devices.

To see what database devices already exist for a particular Sybase server

1 Double-click on the Sybase server. The Display Types window opens.

2 Select **Database Device** from the list and click on the **OK** push button. The Database Device object type and any existing database devices are displayed on the object tree.

There should be at least one device named **d_master**. The Sybase system databases reside on this device.

To see the databases on a particular device, double-click on that device.

You can also display information about the database device, such as its size, by using the **Information** action.

Creating Database Devices

Before creating a database device, you need to know:

- The logical device name
- The physical name of the operating system file or raw partition
- The device number
- The size of the device in pages (2K blocks)

Using DataHub, you can retrieve the available devices from the Sybase server and display the values on the Device No field in the Create Database Device window.

To create a database device

1 Select the Database Device object type and click on the **Create** push button. The Create Database Device window opens. You must specify the device name, the physical name of the disk or file where this device will reside, and the size of the device.

2 Click on **OK**.

> **Please note:** If creating the database device fails while setting up the disk space, there is a chance that the file still exists and you will need to delete the file before creating the database device again.

Dropping Database Devices

Use the DataHub **Drop** action to drop a database device. Besides dropping the selected device, DataHub also deletes the physical operating system file for the device.

> **Attention:** Do not drop the d_master database device. If you do, the Sybase server will be inoperative.

To drop a database device

1 Select the database device and click on the **Drop** push button. The Operating System Login for Server Operation window opens.

2 Enter the appropriate password and click on the **OK** push button. A question box opens asking for confirmation.

3 Click on the **Yes** push button.

You must restart the Sybase server to free the logical device number that belonged to the dropped device.

Designating Default Devices

A *default device* is used by Sybase server logins for creating databases. Whenever a login creates or alters a database without specifying a database device, new disk space is allocated from the pool of default disk space.

Use the DataHub **Default** action to designate a database device as a default device or remove a device from the pool of default devices.

To put a database device in the pool of default devices

1 Select the database device and click on the **Default** push button. The Disk Default window opens.

2 Click on the Default check box to designate the device as a default device.

3 Click on the **OK** push button.

To remove a database device from the pool of default devices

1 Select the database device and click on the **Default** push button. The Disk Default window opens.

2 Click on the **OK** push button.

Mirroring, Unmirroring, and Remirroring Database Devices

Nonstop recovery in the event of media failure is provided by mirroring Sybase database devices to a separate physical disk. The database devices are "duplicated": all writes to the device are copied to a separate physical device. If one of the devices fails, the other contains an up-to-date copy of all transactions.

To mirror a database device

1 Select the database device and click on the **Mirror** push button. The Mirror Device window opens.

2 You must specify the full path name of the database mirror device that is to be your secondary device. For information on the parameters in this window, see online help.

3 Click on the **OK** push button.

Unmirror a device when you need to stop the mirroring process for hardware maintenance or to change a hardware device.

To unmirror a database device

1 Select the database device and click on the **Unmirror** push button. The Unmirror Device window opens.

2 Specify which side of the mirror to disable and whether the unmirroring process is temporary or permanent.

3 Click on the **OK** push button.

Remirror a disk to restart disk mirroring after it is stopped by failure of a mirrored device or temporarily disabled by disk unmirroring.

To remirror a database device

1. Select the database device and click on the **Remirror** push button. A question box opens for confirmation.

2. Click on the **Yes** push button.

Managing Sybase Database Segments

A Sybase database device can be divided into segments. Also, a single segment can be created to span multiple database devices.

You can use segments to control the use and storage of specific database objects. If you run out of storage space on the current device, you can extend the segment across a new device and gain more space.

You should define segments before you create objects in the database. Then, after the segment is created, you can create objects to take advantage of the segment structure.

To display the segments on a database

Double-click on the database, select **Segment**, and click on **OK**. The Segment object type and existing segments are displayed on the object tree.

To display information about a particular segment, select that segment occurrence and click on the **Information** push button. A window opens displaying information such as size of the segment and the tables on the segment.

To create a segment

1. Select the Segment object type and click on the **Create** push button. The Create Segment window opens.

2. Enter the name of the new segment and select a database device by clicking on the **List** push button and selecting from the list of devices.

   You must select the device that is used by the database for this segment.

3. When finished, click on **OK**.

For a segment to span multiple devices, you need to extend it. For example, if you run out of storage space on one device, you can extend the segment to another device.

To extend a segment

1 Select the segment and click on the **Extend** push button. The Extend Segment window opens.

2 Choose a device and click on **OK**.

To drop a segment, select the segment occurrence and click on the **Drop** push button.

Managing Thresholds

A Sybase server has a Threshold Manager which monitors the disk space that is available for Sybase databases. When a specified threshold is reached, the Threshold Manager executes a user-defined stored procedure that can perform a function such as allocate more space.

To see the information for a threshold on a segment

1 Select the segment and click on the **Threshold Info** push button. The Threshold Information window opens. This window displays information for this segment such as the name of the threshold procedure and the size of the threshold.

2 Click on **OK** to close the window.

To see the thresholds for a segment, double-click on the segment occurrence. The Threshold object type and any existing thresholds for that segment are added to the object tree.

To set a threshold for a segment, you need to specify the number of free pages left before the threshold procedure is executed and specify the name of the procedure to execute.

To create a threshold for a segment

1 Select the Threshold object type and click on **Create**. The Create Threshold window opens.

2 Enter the number of free pages, database name, procedure owner, and procedure name.

3 Click on **OK**.

To drop a threshold, select the threshold occurrence and click on the **Drop** push button.

Creating a Sybase Database

When you create a Sybase database, you need to specify the database device for the database. You can create the database on a single device, or have the database span a number of devices. You should not create a database for users on the master device.

All of the changes made to a database are recorded in a *transaction log*. By default, the transaction log for a database is kept on the same database device where the database is created. You might choose to write the transaction log to a different device so that if the disk for the database fails, you can use the transaction log to restore the database. However, when you use a device for logging transactions, you cannot use it for other purposes. You can put a number of logs on the same device.

To create a Sybase database:

1. On the object tree, double-click on the Sybase server that you want the new database connected to. The Display Types window opens.
2. From the Display Types window, select **Database** and click on the **OK** push button. The database object type and any databases defined to the server are added to the tree.
3. Select the database object type.
4. Click on the **Create** push button. The Create Database window opens.
5. Specify values for the parameters in the Create Database window. (For explanations of the parameters, and for information about how to specify values for the **Locate On** and **Log On** parameters, see online help.)

> **Options:** To allocate space for the database and its transaction log from devices of your choice, specify these devices in the **Locate On** and **Log On** boxes. Otherwise, to allocate available space from one or more default devices, leave the **Locate On** and **Log On** boxes blank.

6. Click on the **OK** push button of the Create Database window.

If you want to add out-of-the-box monitoring for the newly created database, you have to use the **Add Std Montr** action on the database. See "Out-of-the-Box Monitoring" on page 293 for more information.

Renaming Sybase Databases

Use the **Rename** action to change the global name of a database.

Attention: Renaming a Sybase database invalidates any database objects, views, stored procedures, triggers, checks, and rules that reference the database name. Anything that references the database name has to be updated and recompiled to ensure that they work with the new database name.

To rename a database

1 Select the database and click **Rename**. The Rename Database window opens.

2 Type the new database name and click on **OK**. A message box opens confirming the change. Click on **OK** to close the message box.

Altering Sybase Databases

Use the DataHub **Alter** action to increase the space allocation for a Sybase database. You can change the devices where the database is located and where the database is logged, as well as set flags for override and load. The Override option allows the same device to be used for the database and the log. The Load option is used when recreating a database for loading from a dump.

To alter a Sybase database

1 Select the Sybase database and click on the **Alter** push button. The Alter Database window opens.

2 Change the parameters you want to change. For an explanation of the parameters, see the online help.

FYI: If you do not specify Locate On and Log On parameters, the default device is used; whatever space is available on your default device is what is allocated.

3 Click on the **OK** push button when finished.

Dropping Sybase Databases

Using DataHub, you can physically remove a Sybase database from the Sybase server on which it resides. The database and its data are erased from the Sybase server.

To remove a Sybase database from a given Sybase server

1. Select the Sybase database and click on the **Drop** push button. A question box opens to confirm.
2. Click on the **Yes** push button to drop the database.

Displaying Lock Information

In a multi-user RDBMS, problems associated with data concurrency, integrity, and consistency are solved by using locks. For example, putting a lock on a table prevents incorrect updates and alterations of that table when the table is shared.

The **Locks** action displays information on all locks related to a Sybase database.

To display locks for a Sybase database

1. Select the database and click on **Locks**. A window opens with a list of locks. Each line shows the id associated with the lock, the type of lock, the id of the table, and other information about the lock. You can also show process information for a lock by selecting the lock and clicking on the **Processes** push button.

   You can kill the process associated with a lock by selecting the lock and clicking on the **Kill** push button.

   You can see table information by selecting a lock and clicking on the **Table Information** push button.

   For more details on lock information, see the online help for the Locks window.
2. When finished, click on **OK**.

Monitoring Sybase Database Processes

To display information about current Sybase database processes, use the **Processes** action.

You can also show the locks for each process as well as kill a process.

To display information on processes

1 Select the Sybase database and click on **Processes**.

2 A window opens with a list of process ids for the selected database. For each process id, a number of parameters are listed, such as its status and host name. For descriptions of each of the parameters, see the online help.

You can show the locks for each process by selecting the process and clicking on **Locks**.

You can kill a process by selecting the process and clicking on **Kill**.

3 When finished, click on **OK**.

Backing Up and Restoring Sybase Databases

Backing up and restoring Sybase databases can involve:

- Backing up both the database and transaction log
- Backing up the transaction log only
- Truncating the transaction log
- Loading the database and transaction log

The following sections describe the DataHub actions you use for backing up and restoring Sybase databases.

Backing Up Sybase Databases

You want to back up your databases periodically so that you can restore them in case of system or media failure.

You can back up Sybase databases while they are in use. Users do not have to log off. The Sybase Backup Server handles copying the transaction logs at the end of creating the backup.

You can use dump devices to back up databases and transaction logs, or specify a physical device for the dump. For information on dump devices, see "Managing Dump Devices" on page 84.

The DataHub **Dump Database** action makes a copy of the entire database, including both the data and the transaction log. This action does not truncate the log.

To back up a Sybase database

1 Display the database on the object tree, select the database and click on **Dump Database**. The Dump Database window opens.

2 Specify the parameters for the backup. You must specify a stripe device. See the online help for more information.

> **Tip:** Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the backup process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.
>
> See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

3 Click on **OK** when finished.

Backing Up Sybase Transactions

A *transaction log* is the record of all changes made to a Sybase database. You can back up the transaction log in order to apply actions to a backup copy of a database when the actions were made after the backup database was created. Use the DataHub **Dump Trans** action to make a backup copy of the transaction log and remove the inactive portion of the log.

You can dump the transaction log while users are connected to the database.

To back up the transaction log for a database

1 Select the database and click on the **Dump Trans** push button. The Dump Trans window opens.

2 Specify the parameters for the backup. You must specify a stripe device. See the online help for more information.

> **Tip:** Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the backup process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.
>
> See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

3 Click on the **OK** push button to start dumping the transaction log.

If the transaction log becomes filled, you can truncate the transaction log to clear space for recording new transactions. You can truncate the transaction log while users are connected to the database.

Truncating the transaction log does not make a backup copy of the transaction log.

To truncate the transaction log

1 Select the database and click on **Trunc Trans**. The Truncate Transaction window opens.

2 Specify the Truncate Option you want. For further description of the options, see the online help or the *Sybase SQL Server Reference Manual*

> **Caution:** The Truncate Only option removes the inactive part of the log without making a backup copy.
>
> The No Log option removes the inactive part of the log without making a backup copy and without recording the procedure in the transaction log.

3 Click on **OK**.

Restoring Sybase Databases

If your database is corrupted and marked suspect by the Sybase server, you need to restore it. To restore a Sybase database that has been corrupted, you must:

- Drop the database
- Recreate the database
- Load the database from the dump device
- Load the transaction log from the dump device

If the database is not marked suspect by the Sybase server, it is accessible by users and you can load the database and transaction log from the dump device without first dropping the database.

Loading the transaction log applies to the backup copy all of the actions made to the database since it was backed up.

While the database and transaction logs are being loaded, users cannot access the database. To load transaction logs, the DBA must have exclusive use of the database.

To load the backup copy of the database

1 Select the database and click on the **Load Database** push button. The Load Database window opens.

2 Specify the parameters for loading the database. You must specify a stripe device. See the online help for more information.

> **Tip:** Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the load process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.
>
> See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

3 Click on **OK** when finished.

For Sybase 11: After loading a backup copy of a user database, Sybase automatically puts the database into offline mode. To bring the database online, you must issue the following command:

online database *database_name*

You can do this from DataHub by selecting the **Options** menu, selecting **Query**, entering the command in the Query window, and then selecting the **Execute** push button. The database is brought online.

To load the transaction log for the database

1 Select the database and click on the **Load Trans** push button. The Load Transaction window opens.

2 Specify the parameters for loading the transaction log. You must specify a stripe device. See the online help for more information.

> **Tip:** Use the **Async. Exec.** parameter to indicate whether the DataHub for UNIX Operating Systems window and DataHub should be available for you to manage other hosts, databases, and host and database objects during the load process. If you want the DataHub window to be available, select the **Asynchronous** option. If you want the DataHub window to be unavailable, select the **Synchronous** option.
>
> See "Working with Finished and Asynchronous Jobs" on page 12 for information on monitoring asynchronous jobs.

3 Click on **OK** when finished.

Displaying Information on Sybase Databases

You can display information such as creation date and size of Sybase databases by using the **Information** action.

Changing Database Ownership

A system administrator might want to create a Sybase database and then give ownership of the database to another user after completing some initial work on the database.

You can change the owner of a database by using the DataHub **Change Owner** action.

> **Attention:** Do not change the owner of the sybsystemprocs database.

To change database ownership

1 Select the database occurrence and click on the **Change Owner** push button. The Change Owner window opens.

2 Enter the name of the new owner, or click on the **List** button to select a name from the list of server logins.

3 Select the Transfer Aliases check box if you want to transfer aliases and their permissions to the new database owner.

4 When finished, click on the **OK** push button.

Managing Database Options

You use database options to customize and fine tune user applications on the database.

> **Tip:** You can only change options for user databases. You cannot change options for the master database.

To display or change those Sybase database options you can set

1 Select the database occurrence and click on the **Database Option** push button. The Database Option window opens.

The database options you can set are listed in this window. If an option is currently set, the check box next to the option is already selected.

To change the settings for an option, click on the check box next to the option.

2 When finished, click on the **OK** push button.

Database Security

Chapter 12. Managing Access to Databases

Using DataHub, you can manage users′ access to your managed databases. You can also manage the privileges and authorizations of those users.

Managing Database Users

The user of a database is someone who can access that database. Accessing a database usually means logging into the database. However, the fact that a user can log into a database does not necessarily mean that user can access the data on the database. Accessing the data depends on what authorizations and privileges that user name has.

> **Please note:**
>
> For INGRES databases, when a user is defined to one database, it is defined to the entire INGRES installation.

Managing database users can involve:

- Finding out the users defined on a database
- Adding new users
- Ing Ora Updating users
- Ora Granting roles to users
- Syb Displaying information about users
- Syb Assigning aliases for users
- Managing privileges and authorizations for users
- Dropping users

Managing privileges and authorizations for users is discussed in Chapter 13, "Managing Security of Databases" on page 181. The following sections discuss the other tasks.

Identifying Database and INGRES Installation Users

To find out the users defined on a database or INGRES installation, double click on

DB2 Ora Syb The database occurrence
Ing The INGRES installation occurrence

select **User** from the list, and click on **OK**. The User object type and any user occurrences defined on that database or INGRES installation are displayed on the object tree.

Creating New Database Users

The conditions on new users for a database vary slightly for different database types.

- DB2 You must choose the name of the user from the list of operating system user names.
- Ing You can choose any name you want as a user of an INGRES installation. When a user is defined to an INGRES installation, the user has access to all public databases in the installation.
- Ora You can choose any name you want as a user of an ORACLE database.
- Syb You must choose the name of the user from the list of login names on the Sybase server that has the database.

To designate a person as a new user of a database

- DB2 Ora Ing

  1 On the object tree, double-click on

  - DB2 Ora The database occurrence
  - Ing The INGRES installation occurrence

  The Display Types Window opens.

  2 From the Display Types window, select **User** and click on the **OK** push button. The User object type and any users of the database you double-clicked on are added to the object tree.

  3 On the object tree, select the User object type.

  4 Click on the **Create** push button.

  - DB2 The Create User window opens.
  - Ing The Create User window opens.
  - Ora The Create_User_Ora7_Popup window opens.

  5 Then:

  - DB2 From the Create User window, select a user and click on the **OK** push button.
  - Ing Specify values for the parameters in the newly opened window. Note that the parameters include one for the name of a new user and an optional one for the name of the group to which you want to assign the user. If the **Default** group is not specified, the user created is not associated with any group. (For more information about the parameters, see online help.) Then click on the window's **OK** push button.

– Ora Specify values for the parameters in the Create_User_Ora7_Popup window.

  You can specify Identification External to have the database verify user access through the operating system, instead of by database password.

  For explanations of the parameters in this window, see online help.

  Then click on the window's **OK** push button.

- Syb

  1. On the object tree, double-click on the Sybase server that supports the database you want to provide a user for. The Display Types Window opens.
  2. From the Display Types window, select **Database** and click on the **OK** push button. The database object type and any databases supported by the Sybase server you double-clicked on are added to the object tree.
  3. Double-click on the database you want to provide a user for. The Display Types window opens again.
  4. From the Display Types window, select **User** and click on the **OK** push button. The User object type and any users of the database you double-clicked on are added to the object tree.
  5. On the object tree, select the User object type under the Sybase database occurrence.
  6. Click on the **Create** push button. The Create User window opens.
  7. Specify values for the parameters in the Create User window. Note that the parameters include one for the name of a new user and one for the name of the group to which you want to assign the user. (For more information about the parameters, see online help.) Then click on the **OK** push button.

Updating Users

Ing Ora For users of ORACLE and INGRES databases, you can update information such as their default profile (ORACLE) or default group (INGRES).

To update a user:

1. Select the user occurrence on the object tree and click on
   - Ing **Update**

- Ora **Alter**

  A window opens in which you can update user information.

2 Change the parameters you want to change and click on **OK**.

Granting Roles to ORACLE Users

Ora For users of ORACLE databases, you can grant roles to them that give the user all the privileges and authorizations associated with those roles. For a discussion of ORACLE roles, see "Administering ORACLE Roles" on page 182.

To grant roles to an ORACLE user occurrence

1 Select the user occurrence and click on the **Role** push button. The Select One or More window opens.

2 Select the roles you want to grant to this user and click on the **OK** push button.

A default role is one that is automatically enabled for a user when the user creates an ORACLE database session. You can assign a user zero or more default roles. Any role directly granted to the user can potentially be a default role; an indirectly granted role cannot be a default role.

To grant a default role to an ORACLE user occurrence

1 Select the user occurrence and click on the **Default Role** push button. The Select One or More window opens.

2 Select the roles you want for the default roles for this user and click on the **OK** push button.

Sybase Aliases for Sybase Database Users

Syb In a Sybase database, using aliases you can map logins to database user names. This provides better control for the system administrator since each alias user name is using its own selectable unit ID (SUID).

To display the aliases for a Sybase database user

1 Double-click on the user occurrence displayed from a Sybase database occurrence. The Display Types window opens.

2 Select **Alias** from the list and click on the **OK** push button. The Alias object type and any existing aliases for that user name are displayed on the object tree.

To create an alias for a user

1 Select the Alias object type and click on the **Create** push button. The Create Alias window opens.

2 Enter a login name to map to this user, or click on the **List** push button to select from a list of logins.

3 When finished, click on the **OK** push button.

To drop an alias for a user

1 Select the alias occurrence and click on the **Drop** push button. A question box opens asking for confirmation.

2 Click on the **Yes** push button to confirm.

Displaying Information about Users

For a Sybase database, you can display information about a user such as the user's group name. To display this information, select the user occurrence (under a Sybase database occurrence or group occurrence) and click on the **Information** push button.

Dropping Users

If you want to remove a user from a database or INGRES installation, select the user occurrence and click on the **Drop** push button.

Managing Database Groups

A *group* is a logical organization of users. Groups allow you to assign permissions for a number of users at once. When you specify privileges and authorizations for a group, they are specified for all the users in that group.

You can have groups for DB2 Version 2, INGRES, and Sybase databases.

> **Please note:** For INGRES databases, when a group is defined for the INGRES installation, it is defined to all of the databases in the installation.

Managing database groups can involve:

- Displaying the groups defined to a database
- Creating and deleting groups
- Adding and deleting users to and from a group
- Assigning privileges and authorizations to a group

Assigning privileges and authorizations to a group is discussed in Chapter 13, "Managing Security of Databases" on page 181.

Displaying the Groups for a Database or INGRES Installation

To display the groups defined on a database or INGRES installation,

1 Double-click on

- DB2v2 Syb The database occurrence
- Ing The INGRES installation

The Display Types window opens.

2 Select **Group** from the list and click on the **OK** push button. The Group object type and any groups defined to this database or INGRES installation are displayed on the object tree.

Tip: Syb With DataHub, you can display under a user occurrence, the Group object type and one group to which that user belongs.

To display the Group object type under a user occurrence, double-click on the user occurrence, select **Group** from the list in the Display Types window, and click on the **OK** push button.

The Group object type and a group to which the user has been assigned are displayed on the object tree. If the user has not been explicitly assigned to a named group, the public group is displayed under the Group object type. If the user has been assigned to a named group, that group occurrence is displayed under the Group object type.

Creating and Deleting a Group

For DB2 Version 2, you must choose the name of your new group from a list of the operating system groups. The privileges and authorizations associated with that particular group name are stipulated for your new group.

To create a group

1 Select the Group object type and click on the **Create** push button.

DB2v2 The Select a Group to Add window opens. Select the group you want and click on the **OK** push button.

Ing Syb The Create Group window opens. Enter a name and click on the **OK** push button.

Please note: Ing For INGRES installations, the name of the group must be unique among all identifiers for users, groups, and roles in the INGRES installation.

To delete a group, select the group and click on **Drop**.

Working with Users in Groups

This section explains:

- Ing Syb How to display users belonging to a group
- DB2v2 How to add a user to a group; how to delete a user from a group.
- Ing One way to add a user to a group (for the other way, see "To designate a person as a new user of a database" on page 173); how to delete a user from a group.
- Syb How to transfer a user from one group to another. For information about how to add a user to a group, see "To designate a person as a new user of a database" on page 173. For information about how to delete a user from a group, see "Dropping Users" on page 176.

Displaying Users in INGRES or Sybase Groups

Ing Syb You can display on the object tree those users that are in a particular group.

To display the users in a particular group

1 Double-click on the particular group object occurrence.

Ing The Select One or More list opens. Select User from the list and select the **OK** push button.

2 The User object type and any user object occurrences in the group are displayed on the object tree.

Working with Users in DB2 Version 2 Groups

DB2v2 To add or delete a user from a DB2 Version 2 group, you must add or delete the user from the corresponding operating system group. See your system administrator for information.

Working with Users in INGRES Groups

Ing To add users to an INGRES group, use the **Add User** action; to delete users from an INGRES group, use the **Delete User** action.

To add users to an INGRES group

1 Select the group object occurrence and click on the **Add User** push button.

The Select One or More window opens with a list of valid users.

2 Select those users you want to add to the group and click on **OK**. The users are added to the group.

To delete users from an INGRES group

1 Select the group object occurrence and click on the **Delete User** push button.

The Select One or More window opens with a list of users currently in the group.

2 Select those users you want to delete from the group and click on **OK**. The users are deleted from the group.

Working with Users in Sybase Groups

FYI:

1. Syb You assign a Sybase user to a group when you first give the user access to a database. For instructions, see "To designate a person as a new user of a database" on page 173. Later, as shown below, you can transfer the user to another group.
2. Each Sybase user belongs to the public group. The only way to completely remove a user from all groups, including the public group, is to remove the user from the database. For instructions, see "Dropping Users" on page 176.
3. Each Sybase user can belong to one named group (a group other than the public group) in addition to the public group for a given Sybase database.

1 On the object tree, double-click on the database. The Display Types Window opens.

2 From the Display Types window, select **Group** and click on the **OK** push button. The group object type and any groups of users of the database you double-clicked on are added to the object tree.

3 Transfer a user to another group from the user's current group

- To transfer a user to the public group:

a On the object tree, double-click on the group to which the user currently belongs. The user object type and any users belonging to the group are added to the object tree.

b Under the group occurrence, select the user occurrence you want to transfer.

c Click on the **Drop** push button. This removes the user from the current group. Because a user must be belong to a group, Sybase assigns the user to the public group.

> **FYI:** If the user currently belongs to the public group, the **Drop** action on the user object occurrence has no net effect. The user remains in the public group.

- To transfer a user from one group (for example, “grp1”) to another group (for example, “grp2”), do one of these:

  – Begin with the new group (“grp2”)

    a On the object tree, double-click on the group (“grp2”) to which you want to transfer the user. The user object type and any users belonging to the group are added to the object tree.

    b Select the user object type.

    c Click on the **Add** push button. The Select User window opens with the list of users defined on this database that are not already in the group (“grp2”).

    d Select the user′s name and click on the **OK** push button.

    Because Sybase users can belong to only one group, the user is transferred from “grp1” to “grp2”.

  – Begin with the user:

    a On the object tree, display the User object type and the user occurrences for the database.

    b Select the user.

    c Click on the **Change Group** push button.

    d Specify the group name (“grp2”) in the newly opened window. (For explanations of the parameters, see online help.) Then click on the window′s **OK** push button.

Chapter 13. Managing Security of Databases

With DataHub, you can manage database security by:

- Administering privileges
- Administering authorizations
- Ing Ora Administering roles
- Ora Managing profiles
- Displaying lock information

In DataHub, the term *privilege* refers to the right to access a database or database objects. Examples of some privileges for the database types are:

- Syb Ing Ora CREATE TABLE
- Ora CREATE SESSION
- DB2 CONNECT

The term *authorization* refers to the right to access a specific database object (for example, the table named “test1” owned by user “testmgr”) in a specific manner (for example, SELECT or INSERT on that table).

Administering Privileges

In DataHub, a *privilege* is a right to access a database or database objects. For example, you can grant a user the privilege to create a table or the privilege to select rows from another user’s table.

You can grant privileges to

- Users
- DB2v2 Ing Syb Groups
- Ing Ora Roles

For information on roles for various RDBMS types, see the corresponding section as listed:

RDBMS	Section
INGRES	“Administering INGRES Roles” on page 183
ORACLE	“Administering ORACLE Roles” on page 182
Sybase server	“Server Roles” on page 81

To grant or revoke privileges to users, groups, or roles

1 On the object tree, display the object (user, group, or role) that you want to grant or revoke privileges for.

2 Select the object occurrence and

- Ing Ora Syb Click on the **Privilege** push button.
- DB2 Click on the **DB Authorities** push button.

The window that opens depend on the type of database the object is on. Any privileges already granted to the object occurrence are shown in the window.

For an explanation of the window, see the online help.

Administering ORACLE Roles

In ORACLE, a *role* is a named set of privileges. A role can combine a set of privileges and even other roles. When you grant or revoke a role for a user, you are granting or revoking all the privileges contained within that role.

You can assign a particular role to a user in order to control the user's capability for accessing tables and other database objects or for executing queries.

Using DataHub, you can:

- List ORACLE roles and users assigned to them
- Create, alter, and delete ORACLE roles
- Grant and revoke system privileges for an ORACLE role
- Grant and revoke object privileges for an ORACLE role
- Grant and revoke ORACLE roles for particular users

See "Administering Privileges" on page 181 for information on granting and revoking system privileges to ORACLE roles; see "Administering Authorizations" on page 184 for information on granting and revoking object privileges to ORACLE roles.

To display ORACLE roles and the users assigned to a particular role

1 Double-click on the database displayed on the object tree, select **Role** from the list, and click on **OK**. The Role object type and object occurrences are displayed on the object tree.

2 Double-click on the role you are interested in, select **User** from the list, and click on **OK**. The User object type and the users assigned to that role are displayed on the object tree.

To create an ORACLE role

1 Select the Role object type and click on **Create**. A window opens.

Enter the name of the role and specify whether identification is by password or not required. If by password, enter a password to be used to enable the role.

2 Click on **OK**. The new role is added to the control database.

When you have created a new role, initially no privileges are associated with that role. When the privileges are set, you can grant the role to users or other roles.

To alter the definition of an ORACLE role

1 Select the role occurrence and click on **Alter**. A window opens.

2 Change the parameters you want to update and click on **OK**.

To delete an ORACLE role

1 Select the role occurrence and click on **Drop**. A question box opens asking for confirmation.

2 Click on **Yes**.

To grant or revoke ORACLE roles for particular users

1 Select the role occurrence and click on **Grantee**. The Display Types window opens. In this window is listed all of the users on that database. The users which already have this role are highlighted.

- To grant this role to one or more users or roles

  Select one or more users or roles in the list and click on **OK**.

- To revoke this role for one or more users or roles

  Select one or more highlighted users or roles in the list and click on **OK**.

Administering INGRES Roles

In INGRES, roles are used to associate privileges with INGRES applications.

Using DataHub, you can:

- List the INGRES roles defined in an INGRES installation
- Create, alter, and drop INGRES roles
- Grant and revoke privileges for an INGRES role
- Grant and revoke authorizations for an INGRES role

For information on granting and revoking INGRES installation privileges for INGRES roles, see "Administering Privileges" on page 181.

For information on granting and revoking authorizations for INGRES roles, see "Administering Authorizations" on page 184.

To display INGRES roles in an INGRES installation

1 Double-click on the installation displayed on the object tree, select **Role** from the list, and click on **OK**. The Role object type and object occurrences are displayed on the object tree.

To create an INGRES role

1 Select the Role object type and click on **Create**. The Create Role window opens.

Enter the name of the role and specify whether identification is by password or not required. If by password, enter a password to be used to enable the role.

2 Click on **OK**. The new role is added to the installation.

To change the definition of an INGRES role

1 Select the role occurrence and click on **Alter**. A window opens.

2 Change the parameters you want to update and click on **OK**.

To delete an INGRES role

1 Select the role occurrence and click on **Drop**. A question box opens asking for confirmation.

2 Click on **Yes**.

Administering Authorizations

An authorization is the permission to access a particular database object in a specific manner. For example, you can authorize a user permission to insert into your table named "MyTable".

With DataHub, you can manage authorizations by either

- Granting permission for a user, group, database role, or Sybase server role to use a particular object, or
- Granting permission on an object occurrence to be used by a particular user, group, database role, or Sybase server role.

Granting Authorizations for Users, Groups, Database Roles, and Sybase Roles

The steps involved in granting authorizations for users, groups, database roles, and Sybase server roles are similar for all those object types. First, the steps are described for occurrences of the above object types, and then an example of granting authorization for a DB2 Version 2 user follows.

To grant authorizations for users, groups, database roles or Sybase server roles

1 Select one or more object occurrences for which you want to grant authorizations. The objects must be of the same object type (for example, all users).

2 Select the **Authorization** push button. A window opens and displays the objects these object occurrences are already authorized to use, if any.

The window has two areas. The first area is the list of objects the selected object occurrences already have permissions to use. The other area is a set of columns headed by the name of the object occurrences. Under each name is a representation of the types of authorizations that object occurrence has for the object. See the online help for more information on these window areas.

3 To add authorizations for a database object, select the name of the object occurrence in the second area and click on the **Add** push button.

A window opens in which you can choose a database object and specify authorizations for that object. See the online help for an explanation of the parameters in this window. When finished adding authorizations, click on the **OK** push button.

4 To alter authorizations for a database object, select the cell that has the authorizations for that object and click on

DB2 Ora The **Alter** push button.
Ing Syb The **Change** push button.

A window opens in which you can change the authorizations for the database object. Change the authorizations you want and click on **OK** when finished.

5 To drop authorizations for a database object, select the cell that has the authorizations for that object and click on the **Drop** push button. The authorizations are removed.

6 When finished with the first window, click on **OK**.

Example: For example, you have a DB2 Version 2 user named "tester1" to which you want to authorize to use three tables ("test1", "test2", and "test3") that are owned by the user "testmgr". You want the user to be able to select, insert, and update the table. The user has no current authorizations.

1. Select the "tester1" user and click on **Authorization**. A window opens that has a column headed by Table/View and a a column headed by

tester1. Because the user has no current authorizations, the areas under the column headings are blank.

2. Select **tester1** and click on **Add**. The Grant Table Privileges window opens.

3. Add a table by clicking on **Add**. A window opens.

   Enter test% in the Name field (to get the three tables "test1", "test2", "test3") and click on the **OK** push button. A window opens and displays a list of object occurrences that fit this filter. The format of the list is "object name - owner name - object type", where "object type" can be "table" or "view".

   For our example, the window shows

   ```
   test0 - testmgr - view
   test1 - testmgr - table
   test2 - testmgr - table
   test3 - testmgr - table
   test4 - testmgr - table
   testa - testmgr - view
   testb - testmgr - view
   ```

   These are all of the tables and views owned by "testmgr" that fit the filter criteria.

   Select "test1", "test2", "test3" and click on the **OK** push button.

4. Select the check boxes next to Select, Insert, and Update in the Grant Table Privileges window and click on the **OK** push button. The window closes.

   In the first window, the tables are listed in the Table column, and the authorizations are listed in the "tester1" column.

5. Click on **OK** to close the window.

Suppose you want to remove authorization for updating the "test3" table.

1. Select the "tester1" user and click on **Authorization**. The window opens. This time, the three tables are shown under the Table/View column.

2. Select the cell containing the authorizations for "test3" and click on the **Alter** push button. The Alter Table Privileges window opens.

3. Click on the check box for Update so that it is not filled in, then click on the **OK** push button. The authorizations for "test3" are updated in the first window.

4. Click on **OK** to close the window.

Granting Authorizations on Specific Database Object Occurrences

The objects you can grant authorizations on are

- Tables
- Views
- DB2 Ing Indexes
- Ing Ora Syb Stored procedures
- DB2 Ora Packages
- Ing Events
- Ing Rules

To grant authorizations on one or more specific database objects

1 Select the database objects. The objects must be of the same object type (for example, all tables).

2 Click on the **Authorization** push button. A window opens and displays the grantees who are already authorized to use the selected objects, if any.

The window has two areas. The first area is the list of grantees who are already authorized to use the selected objects. The other area is a set of columns headed by the names of the objects you selected. Under each name is a representation of the types of authorizations the grantee has for the object. See the online help for more information on these window areas.

3 To grant an authorization for a grantee, select the name of the object in the second area and click on the

- DB2 Ing Syb **Add** push button
- Ora **Grant User** push button to grant to users
- Ora **Grant Role** push button to grant to roles

4 To change an authorization for a grantee that is listed in the first area, select the cell with the authorizations and click on the

- DB2 Ing Syb **Change** push button
- Ora **Alter** push button

5 To drop an authorization for a grantee that is listed in the first area, select the cell with the authorizations and click on the **Drop** push button.

6 When finished, click on **OK**.

Example: For example, you have three tables (“Test10”, “Test11”, and “Test12”) and you want to authorize the user “Tester1” to select, insert, and update on these tables. The user has no current authorizations on these tables.

1. Display the three tables on the object tree and select all three.
2. Click on the **Authorization** push button. A window opens that has a Grantee column and three columns headed by the names of the tables.
3. Select “Test10” and click on the **Add** push button. The Set Grants For User window opens.

   Enter Tester1 in the User/Group Name field.

   Click on the check boxes next to Select, Insert, and Update.

   Click on the **OK** push button when finished. A row is added in the first window with “Tester1” and the set of authorizations for table “Test10”.
4. Repeat using the **Add** push button for tables “Test11” and “Test12”.
5. Click on the **OK** push button when finished.

Types of Authorizations

Consult the SQL reference manual for the RDBMS you are using to find the types of authorizations allowed for the database objects in that RDBMS.

> **Please note:** For INGRES databases, if a user is granted permission to use a view or procedure, then the user can do so without needing permission on the underlying tables or views.

Managing ORACLE Profiles

An ORACLE profile is a named set of resource limits for databases and instances. Users are assigned profiles to control their resource limits. You can assign different profiles to individual users; a default profile is assigned to all users that are not explicitly assigned a profile.

Using DataHub, you can

- List the profiles on a database
- List the users who have been assigned a particular profile
- Create, alter, and delete profiles
- Assign a profile to a user

To list the profiles on a database

Double-click on the database occurrence, select **Profile** from the list, and click on **OK**. The Profile object type and any existing profiles are added to the object tree.

To list the users who have been assigned a particular profile

Double-click on the profile occurrence, select **User** from the list, and click on **OK**. The User object type and the users who have been assigned that profile are added to the object tree.

To create a profile

1 Select the Profile object type and click on **Create** from the list, and click on **OK**. The Create Profile window opens.

2 Type in the name of the profile in the **Profile Name** parameter.

3 Click on the radio buttons under each resource parameter to specify one of these three values:

- Unlimited. This places no restriction on the use of the resource.
- Default. This is the value in the default profile. It is selected by default.
- Limit. This places a limit on the use of the resource.

4 When finished, click on **OK**.

To alter a profile

1 Select the profile occurrence and click on **Alter**. The Alter Profile window opens.

2 Change the resource limits you want to change and click on **OK**.

To delete a profile

1 Select the profile occurrence and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**. The profile is removed from the object tree.

To assign a profile to a user

1 Select the user occurrence in the object tree and click on **Update**. The Update User window opens.

2 Click on the **List** push button next to the **Profile** parameter. The Select Profile window opens.

3 Select a profile from the list and click on **OK**.

4 Click on **OK** in the Update User window.

Chapter 14. Monitoring and Auditing for Managed Databases

DB2v2 DataHub V.2 event monitors are ways to keep track of database activity over a period of time. For example, administrators can use event monitors to gather information on system thresholds, deadlocks, or transaction complete conditions. You can manage your DB2 Version 2 event monitors using DataHub.

Ora Syb Auditing lets you monitor and record specific RDBMS activity to help you detect suspicious use of the system. You can manage auditing for Sybase and ORACLE using DataHub.

Auditing for Sybase is discussed in "Managing Auditing for Sybase" on page 193; auditing for ORACLE is discussed in "Managing Auditing for ORACLE" on page 197.

Managing Events for DB2 Version 2

A DB2 Version 2 event is an occurrence of significance to a task, for example, the completion of an asynchronous operation such as an input/output (I/O) operation. You create a DB2 Version 2 event monitor to collect information related to transient events as they occur. For example, information about deadlocks is collected by an event monitor when the deadlock occurs, before it is resolved and the information disappears.

Using DataHub, you can display the event monitors that currently exist on a database, create event monitors to monitor specific events, activate or deactivate event monitors, and drop event monitors. DataHub calls the object type associated with DB2 Version 2 event monitors the Event object type.

To display the DB2 Version 2 event monitors that exist on a database

1 Double-click on the database. The Display Types window opens.

2 Select Event from the list and click on the **OK** push button. The Event object type and the event monitor occurrences for this database are displayed on the object tree.

You define an event monitor to record certain events that occur when using the database. The definition also specifies where the database should record the events.

To create an event monitor

1 Select the Event object type and click on the **Create** push button. The Create DB2v2 Event window opens.

For information on the parameters in this window, see the online help.

2 Specify the name of the event monitor, where to write the event information (Event Target), the path name, and event file options (if the target is a file or set of files). You can specify the event file options by entering options in the Event Options field, or clicking on the **List** push button. If you click on the **List** push button, the Create Event File Options window opens and you can specify the options using the parameters in this window. See online help for information about the parameters in the Create Event File Options window.

3 To define the type of event to record, click on the **Add** push button in the Event Monitored field. The Define Event Condition window opens.

4 Specify the type of event to record in the Monitor Name field by either entering it in the field or clicking on the **List** push button to select from the list of types (such as "deadlocks").

5 Specify the conditions for this monitor using the Boolean parameters (And/or, Not, Operator) in this window to specify the event SQL statement. For information on the parameters, see the online help.

You can define a filter that determines which connections cause a CONNECTION, STATEMENT, or TRANSACTION event to occur by selecting a Condition Subject (Application ID, Auth ID, Application Name), entering a comparison string, and clicking on the **Commit Selected Where Clause** push button. The WHERE clause is displayed in the Event SQL field. You can edit the SQL statement in the field.

6 Click on the **OK** push button in the Define Event Condition window.

The event monitored you defined is displayed in the Event Monitored window.

7 Define another event type for this monitor by clicking on **Add**; change or drop a displayed event type by selecting the event and clicking on the **Change** or **Drop** push button.

8 Specify when to start the event monitor by selecting one of the radio buttons for the Start Event parameter.

9 Click on the **OK** push button in the Create DB2v2 Event window to commit your specifications.

To activate or deactivate an event monitor

1. Select the event monitor occurrence and click on the **Set Event** push button. The Set Event window opens.

2. Select the state (activate or deactivate) you want by clicking on the **State** push button.

3. Click on the **OK** push button to set the event monitor.

An event monitor must be deactivated before you can drop it.

To drop an event monitor

1. Select the event monitor occurrence and click on the **Drop** push button. A question box opens asking for confirmation.

2. Click on **OK** to confirm deleting the event monitor.

Managing Auditing for Sybase

Sybase auditing lets you audit at a Sybase server, Sybase database, user, or object level. For detailed information on Sybase auditing, see the *Sybase SQL Server System Administration Guide*.

With DataHub, at the Sybase server level, you can:

- Display the current status of auditing options
- Display the audit records for each auditing option
- Enable or disable different auditing options

for the Sybase server and for Sybase server logins.

Auditing options for Sybase server logins is discussed in "Auditing Logins" on page 194.

See "Managing Auditing Options for Sybase Databases" on page 195 for using DataHub to manage Sybase database auditing; see "Auditing for Sybase Tables and Views" on page 196 for using DataHub to manage auditing for tables and views on Sybase databases.

Managing Auditing Options for Sybase Servers

In DataHub, use the **Audit** action to work with auditing options for a Sybase server.

To open a window to work with auditing options for a Sybase server

1. Select the Sybase server and click on the **Audit** push button. The Audit window opens.

This window displays a table with the available Sybase server auditing options and the current settings for the audit options.

Each row in the table has:
- Name of the audit option (for example, “logins”)
- Current value of the option (for example, “on”)

For a description of the available auditing options, see the online help or the *SQL Server Administration Guide*.

In this window you can change settings for audit options, as well as show the audit trail for a particular audit option.

2 To change the settings for an audit option:

a. Select the audit option and click on the **Change** push button. The Change Audit Option window opens.

b. Set the audit option value using the selection button next to the Audit Option Value field.

   For information on this window, see the online help.

3 To show the audit trail for an audit option:

   Select the audit option and click on the **Show** push button. The Show Audit Data window opens.

   This window shows the audit records for the selected audit option. For an explanation of the fields in this window, see online help.

4 When finished in the Audit window, click on the **OK** push button.

Auditing Logins

In DataHub, to audit an individual login, use the **Audit** action on the particular login occurrence. You can display the audit options for a particular login, change the audit options, and show the audit trail.

To display the audit options for a Sybase login:

1 Select the Login and click on the **Audit** push button.

The Audit window opens. This window displays the audit options for Sybase server logins and their current settings. In each row is the name of an audit option (for example, “table”) and the current value of the option (for example, “fail”).

In this window, you can set audit options to audit a Sybase server login’s attempts to access tables or views in any database in this server, or to audit the text of a login’s command batches. You can also display the status of table, view, or command text auditing.

To change the audit options for a Sybase login:

a. Select the audit option row and click on the **Change** push button. The Change Audit Option window opens.

You can set the audit option values by selecting the button next to the Audit Option Value field.

b. Click on the **OK** push button when finished.

To show the audit trail for an audit option for a Sybase login:

a. Select the audit option row and click on the **Show** push button. The Show Audit Data window opens.

   This window shows the audit records for the selected server audit options. For an explanation of the fields in this window, see online help.

b. Click on the **OK** push button.

2 When finished with the Audit window, click on the **OK** push button.

Managing Auditing Options for Sybase Databases

With DataHub, at the Sybase database level, you can:

- Display the current status of auditing options
- Display the audit records for each auditing option
- Enable or disable different auditing options

To open a window to work with auditing options for a Sybase database

1 Select the Sybase database and click on the **Audit** push button. The Audit window opens.

This window displays a table with the available Sybase database auditing options and the current settings for the audit options.

Each row in the table has:

- Name of the audit option (for example, “drop”)
- Current value of the option (for example, “ok”, “off”, “both”, “fail”)

For a description of the available auditing options, see the online help or the *SQL Server Administration Guide*.

In this window you can change settings for audit options, as well as show the audit trail for a particular audit option.

2 To change the settings for an audit option:

a. Select the audit option and click on the **Change** push button. The Change Audit Option window opens.

b. Set the audit option value using the selection button next to the Audit Option Value field.

   For information on this window, see the online help.

3 To show the audit trail for an audit option:

Select the audit option and click on the **Show** push button. The Show Audit Data window opens.

This window shows the audit records for the selected audit option. For an explanation of the fields in this window, see online help.

4 When finished in the Audit window, click on the **OK** push button.

Auditing for Sybase Tables and Views

With DataHub, for Sybase table and view auditing, you can:

- Display the current status of auditing options
- Display the audit records for each auditing option
- Enable or disable different auditing options

In DataHub, use the **Audit** action to work with auditing options for a Sybase table or view.

To open a window to work with auditing options for a Sybase table or view

1 Select the Sybase table or view occurrence and click on the **Audit** push button. The Audit window opens.

This window displays a table with the available auditing options for the object occurrence you selected and the current settings for the audit options.

Each row in the table has:

- Name of the audit option (for example, "delete")
- Current value of the option (for example, "ok", "off", "both", "fail")

For a description of the available auditing options, see the online help or the *SQL Server Administration Guide*.

In this window you can change settings for audit options, as well as show the audit trail for a particular audit option.

2 To change the settings for an audit option:

a. Select the audit option and click on the **Change** push button. The Change Audit Option window opens.

b. Set the audit option value using the selection button next to the Audit Option Value field.

For information on this window, see the online help.

3 To show the audit trail for an audit option:

Select the audit option and click on the **Show** push button. The Show Audit Data window opens.

This window shows the audit records for the selected audit option. For an explanation of the fields in this window, see online help.

4 When finished in the Audit window, click on the **OK** push button.

Managing Auditing for ORACLE

ORACLE lets you monitor and record the actions of database users, and lets you investigate specific database activities. For example, you can audit certain tables and determine when they are updated or altered and by which user.

But first:

1. Before using DataHub for UNIX Operating Systems to audit database usage, be sure that the AUDIT_TRAIL parameter in your parameter initialization files is set to DB or TRUE.
2. Before you can audit database usage, you must run certain SQL scripts. To find out which scripts these are, see *Oracle Server Administrator's Guide*.

DataHub provides two types of auditing:

Object auditing
Monitors and records all kinds of database actions on a specific database object. Object auditing applies to all database users.

Statement auditing
Includes both SQL statement auditing and system privilege auditing.

- SQL statement auditing monitors and records specific SQL statements based on the type of statement. For example, audit any SQL statement on an index records each CREATE, ALTER, and DROP INDEX statement.
- System privilege auditing monitors and records the use of a particular system privilege, such as CREATE ANY INDEX.

Statement auditing can be set to apply to specific database users.

For both types of auditing, you can audit according to:

- Success

  Audit only successful SQL statements
  Audit only unsuccessful SQL statements
  Audit both successful and unsuccessful SQL statements

- By session or by access

  By session records only one audit trail per user and object per session (audit trail/user and object/session)

By access records an audit trail for each statement that is executed (audit trail/statement)

Object Auditing

When you audit by object, you can audit specific data manipulation language (DML) statements for selected objects.

To see what objects are currently audited

1 Select the database occurrence and click on **Audit Objects**. A window opens with a scrollable list of objects, listed alphabetically by owner name. The window also shows the SQL commands which can be audited.

For each listed SQL command, the auditing type is displayed; abbreviations are used, separated by a slash. The letter to the left of the slash indicates auditing by session (S) or by access (A). The letter to the right of the slash indicates auditing by successful operations (S), unsuccessful operations (U), or all operations (U). A dash (–) indicates that no auditing is done for this command.

For example, an S/A under the insert command indicates that you are monitoring by session (S) and that only unsuccessful operations (U) are monitored.

From this window, you can:

- Add auditing for an object
- Change auditing for an object
- Stop auditing for an object
- Display successful and failed attempts for each SQL statement
- Display the number of sessions in which each SQL statement occurred

2 When you are finished setting auditing for objects, click on **OK** to return to the object tree.

To add auditing for an object

1 In the window for audited objects click on **Add**. The Add Audited Object window opens. In this window, you name an object and specify the options for auditing.

For an explanation of the parameters in this window, see online help.

2 When finished specifying options, click on **OK**. The object is added to the list of audited objects.

To change auditing for an object

1 In the window for audited objects select the object row and click on **Change**. The Alter Audited Object window opens. In this window are listed the auditing choices (successful/not successful; sessions/access) for the selected object.

2 Use the **Add**, **Change**, and **Drop** push buttons to alter your auditing options.

  See the online help for more information.

3 To complete your changes, click on the **OK** push button.

To drop auditing for an object

1 In the window for audited objects select the object row and click on **Delete**. The object is removed from the list.

To display auditing by access for an object

1 In the window for audited objects select the object row and click on **Access**. The Access window displays the number of successful and failed attempts for each action (SQL) listed. The displayed values represent every attempt for all actions during the current audit session.

  For more information about a particular action and the user who accessed the object, select the SQL action and click on **Detail**. A window displays information about the user. See the online help for more information.

2 When finished, click on **OK** to close the window.

To display auditing by session for an object

1 In the window for audited objects select the object row and click on **Session**. The Sessions window displays the number of successful and failed attempts for each action (SQL) listed. The displayed values represent one occurrence per action (SQL) per session, or occurrence/SQL/session. For example, if five sessions ALTER an object twenty times for each session and all times succeed, the number 5 is displayed under the Success column (for the five sessions).

  For more information about a particular action and the user who accessed the object, select the SQL action and click on **Detail**. A window displays information about the user. See the online help for more information.

2 When finished, click on **OK** to close the window.

Statement Auditing

In DataHub, you manage both SQL Statement auditing and system privilege auditing using the **Audit Statement** action.

When you audit by statement, you collect information about the use of a specified SQL statement or system privilege. For example, if you audit the SQL statements on INDEX, the three SQL statements involving indexes, CREATE, ALTER, and DROP, are audited.

You can specify statement auditing for just one user or for a number of users.

To see what SQL statements or system privileges are currently audited

1 Select the database occurrence and click on **Audit Statement**. A window opens with a scrollable list of user names, audit options for each user name, and two columns showing options (by session or by access) for the success and failure of the audit.

From this window, you can:

- Add auditing for an SQL statement or system privilege
- Stop auditing for a statement
- Display successful and failed attempts for each audited statement

2 When you are finished setting auditing for statements, click on **OK** to return to the object tree.

To add auditing for an SQL statement or system privilege

1 In the window for audited statements click on

- The **SQL Statement** push button for SQL statement auditing
- The **System privilege** push button for system privilege auditing.

A window opens. In this window, you add one or more statements and specify options for auditing and the users.

Statements
Shows the audited statements.

To add a statement, click on **Add**, select one or more statements from the scrollable list, and then click on **OK**.

To drop a statement, select the statement and then click on **Drop**.

Users

Shows the user or users whose activities are monitored.

To add a user, click on **Add**, select one or more users from the scrollable list, and then click on **OK**.

To drop a user from monitoring, select the statement and then click on **Drop**.

Whenever

Click on the radio buttons to select the results that are tracked (Successful, Unsuccessful, or Always).

By

Click on the radio buttons to select the option (Access or Session) you want.

2 When finished specifying options, click on **OK**. The statement audit is added to the list.

To drop auditing for a statement

1 In the window for audited statements select the statement row and click on **Delete**. The statement is removed from the list.

To display results for a statement auditing

1 In the window for audited statements select the statement row and click on **Show**. A window opens that lists by username the objects which were affected by the audited statement. For details about the statement, select the statement and click on **Detail**. A window displays information about the user. See the online help for more information.

2 When finished, click on **OK** to close the window.

Managing Database Objects

Chapter 15. Managing Data and Database Objects

A database is a collection of data with a given structure for accepting, storing, and providing, on demand, data for multiple users. Besides managing a database as a whole unit, you also manage the data and database objects that together make up the database.

Managing data and database objects involves:

- DB2v2 Ora Managing storage of data structures using objects such as table spaces
- Managing data objects such as tables and views
- Maintaining data integrity
- Maintaining data security

Managing Storage of Data Structures

Managing storage of data structures involves:

- DB2v2 Ora Managing table spaces
- Ora Managing data files

A database can be partitioned into separately managed parts called *table spaces* (ORACLE calls these *tablespaces*). A table space allows you to assign the location of a database and table directly onto media storage devices or *containers*. Using table spaces, you can group related logical structures together; for example, table spaces commonly group all of a application's objects to simplify certain administrative operations.

Managing Table Spaces

Some of the tasks with table spaces you can do using DataHub are:

- Creating and dropping table spaces
- DB2v2 Altering table spaces
- Ora Updating table spaces
- DB2v2 Adding and replacing comments in the catalog
- DB2v2 Displaying the containers for the table space
- Ora Displaying how much free space table spaces have
- Ora Setting table spaces online or offline (ORACLE)
- Ora Marking begin and end backup points for table spaces

Creating Table Spaces

Use the DataHub **Create** action to create a table space.

To create a table space for a database

1. Double-click on the database occurrence, select **Table Space** and click on the **OK** push button. The Table Space object type appears, along with a list of the existing table spaces for that database.
2. Select the Table Space object type and click on the **Create** push button. The Create Table Space window opens.
3. Fill in the parameters. Some already have default values. You can either accept the defaults or select new values.

   For detailed information on the parameters, see the online help.
4. Click **OK** when finished.

Dropping Table Spaces

Once a table space is dropped, the data in that table space is deleted. Also, immediately before and after dropping a table space from a database, back up the database completely. This is strongly recommended so that you can recover the database if you mistakenly drop a table space, or if the database experiences a problem in the future after the table space has been dropped.

You cannot drop a table space if a table stores at least one of its parts in it and one or more of its parts in another table space.

DB2v2 When you drop a table space, you delete all data in that table space, remove the catalog entries, and all existing database objects with dependencies on the table space are either dropped or marked invalid.

To drop a table space

Select the table space and click on the **Drop** push button.

Ora For ORACLE, a window opens with parameters to specify.

Include Contents
: Select whether to include the contents (**Yes**) or not (**No**). If a table space contains objects, you must select **Yes**.

Cascade Constraints
: Select whether to cascade constraints or not. If tables outside the table space contain referential constraints to tables in this table space, you must select **Yes**.

Remove Datafiles
Select whether or not to remove the data files from the operating system.

When finished, click on **OK**. If there is no current DBM connection for this database, the Operating System Login for Database Operation window opens. Type in a login and password that has the privilege to perform this action, and click on the **OK** push button.

DB2v2 For DB2 Version 2, a question box opens asking for confirmation. Click on the **OK** push button.

Ora When you drop a table space, only the file pointers in the control files of the associated database are dropped; the data files that constituted the dropped table space continue to exist. To free previously used disk space, delete the data files of the dropped table space.

Altering DB2 Version 2 Table Spaces

Use the DataHub **Alter** action to add a container to a DB2 Version 2 table space or modify the PREFETCHSIZE, OVERHEAD, and TRANSFERRATE settings.

To alter a table space

1. Select the table space occurrence and click on the **Alter** push button. The Alter Table Space window opens.

   Change the settings you want.

2. When finished, click on the **OK** push button.

Altering ORACLE Table Spaces

Use the DataHub **Alter** action to update storage parameters for an ORACLE table space.

To alter an ORACLE table space

1. Select the table space and click on the **Alter** push button. A window with the table space storage parameters opens. Change a parameter by selecting the parameter, deleting the current value, and typing in a new value.

   For detailed information on the parameters, see the online help.
2. When finished, click on **OK**. A message box opens to confirm changes. Click on **OK** to close the box.

Managing Comments for Table Spaces

DB2v2 You can create, change, and replace comments for a DB2 Version 2 table space by using the DataHub **Comment** action.

To create, change, or replace a comment

1. Select the table space occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

   If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2. To add a new comment, enter it in the New Comment field.
3. To change an existing comment, edit the comment in the New Comment field.
4. To replace an existing comment, delete the comment in the New Comment field and enter a new one.
5. When finished, click on the **OK** push button.

Displaying the Containers for a Table Space

DB2v2 You can use the **Containers** action push button to display information about the containers, such as table space id and container type, of a DB2 Version 2 table space. Select the table space occurrence and click on **Containers**.

Displaying Free Space in a Table Space

Ora You can display the free space for an ORACLE table space using the **Free Space** action push button. Select the table space occurrence and click on the **Free Space** push button.

Setting Table Spaces Online and Offline

Ora Setting an ORACLE table space online makes the objects within the table space available to users. Setting a table space offline makes it unavailable.

To set an ORACLE table space online

1. Display the table space on the object tree.
2. Select the table space occurrence and click on the **Online** push button.

To set an ORACLE table space offline

1. Display the table space on the object tree.
2. Select the table space occurrence and click on the **Offline** push button.

In the window that opens, select the offline priority: Normal, Temporary, or Immediate.

3 Click on **OK**

Marking Begin and End Backup Points for ORACLE Table Spaces

Use the **Begin Backup** and **End Backup** actions to mark the beginning and the end of a table space backup. It is important to do **Begin Backup** prior to backing up the physical datafile for the table space to yield consistent data for recovery, if needed later.

You might also want to query the backup status of the table space datafiles by using the **Query** choice on the **Options** pull-down menu and entering the command

```
select file#,status FROM v_$backup
```

The **Begin Backup** and **End Backup** actions send messages to ORACLE to alter the start point of the table space.

Before a backup can be made of the contents of a table space, you have to mark the beginning of the backup and the end of the backup.

> **But first:** Before beginning the backup of an online table space, the database must be in ARCHIVELOG mode. To determine if a database is in this mode, you might check your parameter initialization files or the dynamic performance table SYS.V_$DATABASE. If you find that the database is in NOARCHIVELOG mode, you can switch to ARCHIVELOG mode by clicking on the **Archive Log** push button.

To mark the beginning of a table space backup

1 Select the table space and click on the **Begin Backup** action. A message box opens to confirm the start of the backup is marked.

2 Click on the **OK** push button to close the message box.

To mark the end of a table space backup

1 Select the table space and click on the **End Backup** action. A message box opens to confirm the end of the backup is marked.

2 Click on the **OK** push button to close the message box.

Managing ORACLE Data Files

Every ORACLE table space has one or more physical *data files*. An ORACLE database's data files contain all the database data.

Some of the tasks that managing ORACLE data files involves are:

- Creating, renaming, and recovering data files
- Setting data files online or offline
- Displaying file i/o information

The following text explains how to open the windows on which you can perform these tasks.

To create, rename, or recover data files for an ORACLE database

1 On the DataHub window, double-click on the database, select **Datafiles** from the list, and click on **OK**. The Datafiles object type appears, along with a list of the database's existing data files.

2 Do one of the following actions:

- To create a data file for the database
  - **a** Select the data file object type and click on **Create**. The Add a Datafile window opens.
  - **b** Specify the parameters in the window. For an explanation of the parameters, see the online help.
  - **c** When finished, click on **OK**. A message box opens confirming the addition of the new data file.
- To rename a data file, you must first set the data file or table space offline (see "To set an ORACLE data file online" on page 210).

  Select the data file and click on the **Rename** push button. In the window that opens, type the new file name and click on **OK**.

  If there is no current DBM connection for this database, the Operating System Login for Database Operation window opens. Type in a user name and password that has the privilege to perform this action, and click on the **OK** push button.

- To recover a data file, you must first set the database to archive mode.

  Select the data file and click on the **Recover** push button.

To set an ORACLE data file online

1 On the DataHub window, select the data file.
2 Click on the **Online** push button.

If media recovery is not necessary, the data file is set online.

If media recovery is necessary, the data file is not set online and a message box opens to let you know. Click on **OK** to close the message box. You must recover the data file and then set it online.

To set an ORACLE data file offline

1 On the DataHub window, select the data file.
2 Click on the **Offline** push button.

If the table space if online, a question box opens asking you to confirm setting the data file offline.
3 Click on **Yes** if you want to continue.

If the database is running in NOARCHIVELOG mode, a question box opens asking you to confirm setting the data file offline.
4 Click on **Yes** if you want to continue. The data file is set offline.

To display file input and output (File I/O)

Select one or more data files and click on the **File I/O** push button. The File I/O window opens. You might have to enlarge the window to see all of the columns. The columns list information about the reads and writes for the data file in each row.

To see a graph of the history of a particular file, select the file name in the first area of the File I/O window and click on the **History** push button. A window opens and displays the history on a graph. You can print this graph by using the **Print** push button.

Chapter 16. Managing Tables

A *table* consists of data arranged in rows and columns. Tables are the basic unit of storing data in a database.

Displaying Current Information about Tables

You can obtain several kinds of current information about tables such as what tables are stored in a particular database and what columns a table has.

To list the tables stored in a given database

1. Double-click on the database in the object tree, select Table from the list, and click on **OK**. The Filter Table window opens.
2. Enter a filter criterion or leave the fields blank to see all the tables, and click on the **OK** push button. The Table object type and any existing tables are displayed on the object tree.

Use the DataHub **Contents** action to see the data stored in the table.

To open a window that displays the contents of a table

1. Select the table on the object tree.
2. Click on the **Contents** push button. A window opens and displays the contents of the table.
3. Click **OK** to close the contents window.

To display the list of column definitions for a table

1. Select the table on the object tree.
2. Click on the **Column** push button.

Gathering Statistics for Tables

DB2 Ora Syb You can gather statistics for DB2, ORACLE, and Sybase tables.

DB2 For DB2 databases, the database manager maintains statistics about the physical characteristics of tables and indexes and uses them to determine the best way to access the data. DataHub uses the RUNSTATS utility to update these statistics in the system catalog tables. For information on these statistics, see the *Administration Guide* for your DB2 installation.

Syb For Sybase databases, the **Statistics** action returns a list of indexes on a single table. For information on the Sybase sp_statistics command, see the *Sybase SQL Server Quick Reference Guide* for your Sybase installation.

Ora For ORACLE databases, DataHub uses ANALYZE TABLE to gather statistics about the physical storage characteristics of a table. The statistics are stored in the data dictionary. For information on the statistics you can gather on an ORACLE table, see the *Administrator's Guide* for your ORACLE installation.

To gather statistics for a table

1 Select the table on the DataHub window and click on the
- Syb **Statistics** push button
- DB2 Ora **Runstats** push button

2 A window opens.

- Syb The window displays information about the table. For detailed explanation on this window, see the online help.
- DB2v1 A window opens where you can specify whether to update and show statistics, only update statistics, or only show existing statistics. You can also specify whether to write the new statistics or not.

  For information on the options, see the online help.

  Click on the **OK** push button. If you selected Show Statistics or Update and Show Statistics, a window opens with statistics on the table. To close this window, click on the **OK** push button.
- DB2v2 The Runstats window opens. Specify the parameters you want and click on **OK**. For an explanation of the parameters, see the online help.

  If you checked Statistics and Reorgchk or Reorgchk Only, the Runstats window opens with statistics on the table.
- Ora The Estimate or Exact Statistics window opens. Select the one you want and click on **OK**. The Analyze Table opens and displays the statistics about the object used by the optimizer.

3 Click on **OK** to close the window.

Displaying Segment Information for Tables

Ora You can display information about segments allocated for ORACLE tables.

To obtain information about segments allocated for a table

1 Select the table on the DataHub window.
2 Click on the **Segment Info** push button. The Segment Info window opens. For an explanation of the parameters, see the online help.

Displaying Information and Index Information for Tables

Syb You can display information such as the owner of a table and what index keys it has by using the **Information** and **Index Info** actions on a Sybase table.

Displaying Space Information for Tables

Syb You can display the amount of space used by a table and estimate the amount of space needed for a Sybase table by using the **Space Used** and **Estimate Space** push buttons.

For an explanation of the parameters in the Estimate Space window, see the online help.

Creating and Dropping Tables

Use the DataHub **Create** action to create a table.

To create a table

1 Select the Table object type and the **Create** push button. The Create Table window opens.

2 In the Create Table window, specify the parameters for the table. You must specify a table name and column definitions or attributes for the table.

If you displayed the Table object type directly from a host object occurrence (instead of under a particular user), you must also specify a table owner.

For an explanation of the parameters and how to specify column definitions or attributes, see the online help. You can add, change, or drop columns or attributes by using the **Add**, **Change**, or **Drop** push buttons in the:

DB2v1 Ing Ora Syb Columns field

DB2v2 Column Definitions field

Ora You can specify whether this table is used in a cluster by selecting the **In Cluster** toggle button and selecting **Cluster Specification**. For more information see online help.

3 When finished, click on the **OK** push button.

The table is displayed on the object tree.

To drop a table

1 From the object tree, select the table you want to drop.
2 Click on the **Drop** push button. A question box appears, asking if you want to drop the table.
3 Click on the **Yes** push button. If you have sufficient privileges for this table, the table is dropped and the object tree is automatically refreshed to indicate the drop.

Updating Tables

Ing Ora You can change definitions for ORACLE and INGRES tables by using the DataHub **Update** and **Alter** actions.

To update a table

1 From the object tree, select the table you want to update.
2 Click on the
   - Ing **Update** push button
   - Ora **Alter** push button

   A window opens. In this window, you can change certain parameters for the table.
3 Change those fields in the window you want to update to new parameter values. See the online help for explanations of the parameters.
4 Click on the **OK** push button. If you have sufficient privileges for this table, it is updated.

Changing Table Definitions

You can modify the definition of an existing table using the **Column** action. You can add new columns, reorder and delete columns, and change the definitions of the columns. For INGRES, ORACLE, and Sybase tables, you can also convert the data in columns to another data type.

A column change can be a complex operation. The operations carried out by changing columns depend on the types of changes to be made and the RDBMS type. Be aware that changing a column might take some time, and that multiple SQL statements might result from an apparently simple change. For example, changing a column name might mean recreating the table, converting column values, recreating the index, creating primary keys, and copying object privileges. If a change column operation can be carried out using a simple operation (depending on RDBMS type), DataHub uses the appropriate SQL commands.

Certain rules apply to changing columns. See the online help for Columns Window for information.

To modify the definition of an existing table

1 Select the table object occurrence and click on **Column**. The Columns window opens. The columns are listed in a scrollable list. Use the scroll bars to see the attributes for each column.

2 Do one of the following actions:

- Do you want to add a new column? If so:

  a Click on **Add**. The Add Column window opens. For descriptions of the parameters, see the online help.

  b Click on **OK** when finished.

- Do you want to change a column data type? If so:

  a Select the column name and click on **Change**. The Change Columns window opens. Change the attributes of the column you selected.

  b When finished, click on **OK**.

> **Caution:** If there is data in the columns that you are changing, truncation of data might occur, depending on the data type of the column.

- Do you want to delete a column? If so:

  Select the column name and click on **Drop**.

> **Caution:** When you drop a column, the data in that column is also dropped.

- Ing Syb Ora Do you want to convert the data in a column to another data type? If so:

  a Before performing the conversion for data in a column, use the **Change** push button to change the data type of the column to the new data type (see above for a description of changing a column).

  b Select the column name of the column you want to convert and click on **Convert**. If the data type can be converted, a window opens in which you can specify the format for the conversion.

  For information on this window, see online help.

  Ing For information on type conversion functions, see the *INGRES SQL Reference Manual*.

  Syb For information on data conversion formats, see the *Sybase SQL Server Reference Manual*.

  Ora For information on the ORACLE conversion symbols, see the *ORACLE SQL Reference Manual*.

- Do you want to reorder the columns in the table? If so,

  a Select the column name of the column you want to move and click on **Reorder**.

  b In the window that opens, select the column to come before the column you selected to reorder and click on **OK**.

Caution: When there are application programs that are using this table, you might not want to reorder the columns. In this situation, reordering could cause problems for the host variables in the programs.

3 When finished modifying columns, click on **OK**.

If you changed or dropped a column, a question box opens asking to recreate the table through dropping. When you specify to change or drop a column, DataHub creates a temporary table. Then, when you click on **OK**, DataHub recreates the table with the new columns and drops the temporary table. If you select **No**, the temporary table is dropped and the original table is unchanged.

Click on **Yes** in the question box.

Reorganizing Tables

You reorganize a table to rearrange the data into a physical sequence according to a specified index. This can provide faster access to data and improve performance.

DB2 Ing Ora You can reorganize DB2, INGRES, and ORACLE tables by selecting the table occurrence and clicking on the **Reorganize** push button.

Truncating Sybase Tables

Truncating a Sybase table removes all rows from the table.

To truncate a Sybase table

1 Select the table and click on the **Truncate** push button. A question box opens asking for confirmation.

2 Click on the **Yes** push button to confirm truncating the table.

Renaming Tables

Syb Ora You can rename Sybase tables and ORACLE tables by selecting the table occurrence and clicking on the **Rename** push button.

> **Please note:** For DB2 databases, while a table cannot be renamed directly, it can be dropped and recreated with a different name.

Managing Comments for DB2 Version 2 Tables

DB2v2 You can create, change, and replace comments for a DB2 Version 2 table by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the table occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

> **FYI:** In DB2 Version 1, export files for a database are owned by the user ID that owns the instance that manages the database.

Copying Tables

With DataHub, you can copy one or more tables from one database to another. The databases do not have to be the same type; for example, you can copy an ORACLE table to a DB2 database. However, because different RDBMSs have different column types and support different kinds of table constraints, there are some characteristics of copying tables you should consider first.

From one database to another database of the same type

In this case, you can copy

- Table contents and column definitions
- Indexes
- DB2 Ora Syb Column and table constraints
- Ing Integrities
- DB2 Ora Syb Primary key

Authorizations are not copied.

From one database to a database of a different type

In this case, you can copy

- Table contents and column definitions
- Indexes
- DB2 Ora Syb Primary key

Authorizations are not copied.

Table 5 on page 219 shows what is copied from one type of database to another.

Table 5. Copying options when you copy tables between databases

From	To DB2 V.1	To DB2 V.2	To ORACLE	To Sybase	To INGRES
DB2 V.1	Contents and column definitions Indexes Primary key Column and table constraints	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes
DB2 V.2	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key Column and table constraints	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes
ORACLE	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key Column and table constraints	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes
Sybase	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key Column and table constraints	Contents and column definitions Indexes
INGRES	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Primary key	Contents and column definitions Indexes Integrities

Different RDBMSs support different data types for columns. When you copy tables from one database type to another with DataHub, column types are converted. See Appendix F, “Data Type Conversions For Copying Tables” on page 358 for information on how column types are converted when copying tables using DataHub.

To copy tables from one database to another

1 Select the tables you want to copy.

2 Using the rightmost mouse button (assuming your mouse is set for a right-handed user), click and hold down the button. The cursor will change to a different shape.

3 Move the cursor over to the icon for the target database.

4 Release the rightmost mouse button.

If you are not already connected to the target database, the Database Login window opens. Type your name and password and click on **OK**.

5 The Copy Table window opens. Specify:

Table Owner
You must specify an owner that exists at the target database.

Table Name
You can enter a new name for the table at the target database, or use the original name.

Copy Indexes
Click on the check box to specify copying indexes. Default is no copying.

Copy Constraints
Click on the check box to specify copying constraints. Default is no copying.

6 Click on **OK**.

If you are copying more than one table, a Copy Table window will open for each table.

DataHub uses the \tmp file system for temporary working storage when copying tables. Make sure you have sufficient free space in \tmp for DataHub to unload the tables you want to copy.

Chapter 17. Managing Views

A view allows you to look at certain portions of one or more tables in your database. A view can also be used to look at information referenced from other views, or a combination of both tables and views.

Managing views can involve:

- Obtaining current information about existing views
- Creating views
- Updating and dropping views
- Syb Renaming views
- DB2v2 Managing comments for a view

The following subsections discuss these tasks.

Obtaining Current Information on Views

You can obtain several kinds of current information about views. Examples are:

- Which views are stored for a given object
- Syb History and storage information about a view
- The view definition

The following text explains how to find the kinds of information listed above.

To display the list of views on a particular object occurrence

1 Display the object occurrence on the object tree. Database, table, and view occurrences can have views.
2 Double-click on the object occurrence, select View from the list, and click on **OK**. The Enter filter window opens.
3 Enter a filter criterion or leave the fields blank to see all the views, and click on the **OK** push button. The View object type and a list of views defined on the object occurrence are displayed on the object tree.

To display the contents of a view

1 From the DataHub for UNIX window select the view you want to see.
2 Click on the **Contents** push button. The contents of the view appear.

To display information about a Sybase view

Select the view you want information on and click on the **Information** push button. The Information window opens and displays information such as when it was created and the column definition of the view.

Creating Views

You can derive a view from a single object (a table or another view), or from multiple objects (tables or views). In a DB2 environment, the multiple objects must all reside in the same database. In Sybase environments, they can reside either in the same or different databases.

In DataHub's Create View Application window, you first build the query that comprises the subselect part of the view definition; then you complete the CREATE VIEW statement by specifying the view name, altering column names (if you want), and other parameters.

The query that defines a view typically consists of the following clauses:

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING

The following example is used throughout this section to illustrate the steps needed to create a view.

Example of a query you could use in a CREATE VIEW statement: Here, we have a table of employees, that has columns empname, empno, salary, deptno, deptname, division. These columns are the employee name, employee number, employee salary, department number, and division.

We want a view that shows those departments in the HR division, that have an average departmental salary greater than $15,000. We also want the view contents grouped by department number.

Example for creating a view

```
SELECT deptno, deptname, avg(salary)
  FROM employees
    WHERE division = 'HR'
  GROUP BY deptno
  HAVING avg(sal) > 15000
```

To create a view in DataHub:

1 Define the query's FROM clause by selecting in the object tree those object occurrences that you want the view based on. In this section, these object occurrences are called *base objects*

2 From the base objects, choose those columns that you want to use to derive the columns in the view. These are the columns that follow the word SELECT. In this section, these columns are called *base columns*. You can also map view columns to system variables. In this section, such columns are called *system variable columns*.

3 Optional: Work with the columns in the SELECT clause. You can select operations or functions you want to apply to a column's values, delete a column from the list, edit a column in the list, or add the column to the HAVING clause.

4 Optional: Add WHERE clauses

5 Optional: Add JOIN conditions

6 Optional: Add HAVING clauses

7 Complete the view's definition and have DataHub create the view.

Selecting the Base Objects

To select the base objects

1 Display on the object tree those object occurrences that you want your view derived from.

2 Select the object occurrences.

Please note: Ora A privilege to select a base object in another user's schema must be granted explicitly. It cannot be granted through a role.

3 Select the **Create View** push button. The Create View Application window (Figure 6 on page 224) opens.

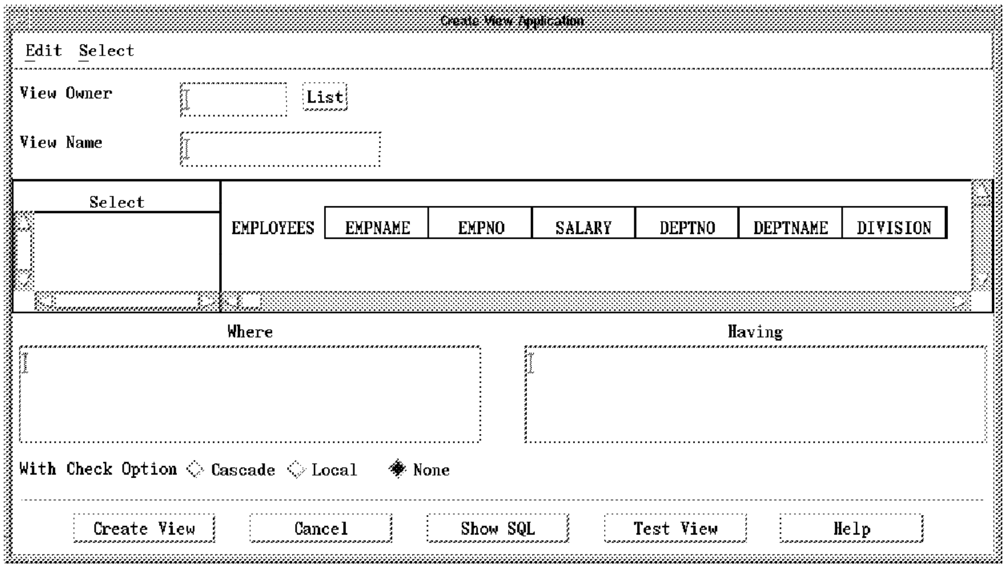


Figure 6. Create View Application window

The Create View Application window has these sections:

- A menu bar at the top.
- Fields for specifying the view name and owner.
- A **Select** box used to list the columns that you choose.
- A space containing a schematic of the base objects you selected. This space is the *base object area*.
- A **Where** box for writing WHERE clauses.
- A **Having** box for writing HAVING clauses.
- A set of push buttons at the bottom of the window.

You define your view in this window.

By selecting the appropriate push button, you can

- Show the SQL for the view definition as it is currently defined in the window.
- Test the current view definition. See “Testing Your View” on page 235 for more information.
- Directly edit SQL for the CREATE VIEW statement defined by the parameters in the window. See “Completing the View Definition” on page 235 for more information.

At any time as you are defining your view in the Create View Application window, you can show the SQL or test your view.

Choosing Columns for the SELECT Clause

You can use the mouse in the Create View Application window to choose the columns you want in the SELECT clause.

To choose base columns

1 Position the cursor over the first column you want to add for the SELECT statement and hold down the right-hand mouse button. A menu opens (see Figure 7).

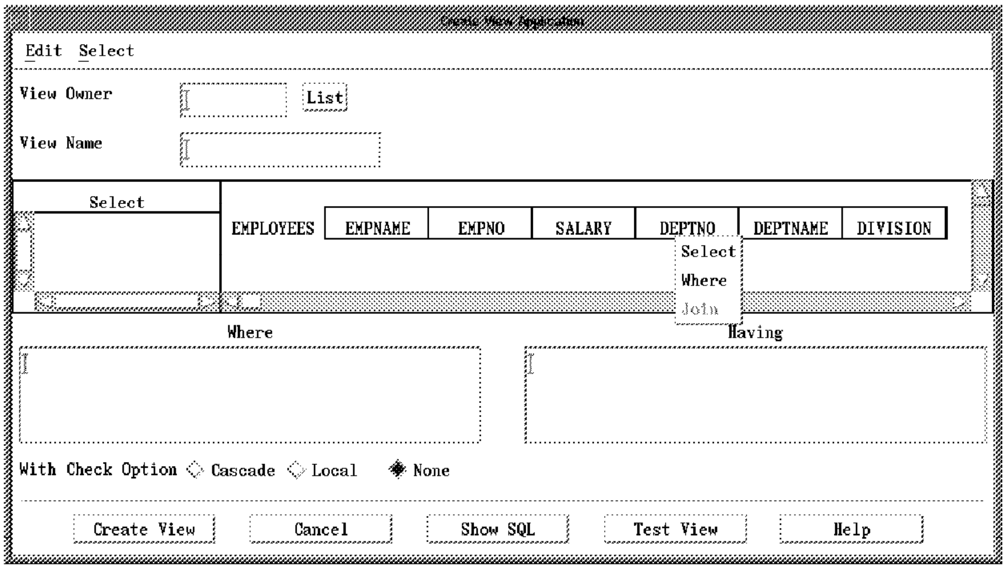


Figure 7. Base column menu

2 Holding down the right-hand mouse button, select the **Select** choice from the menu. The column is added to the Select listbox (see Figure 8 on page 226).

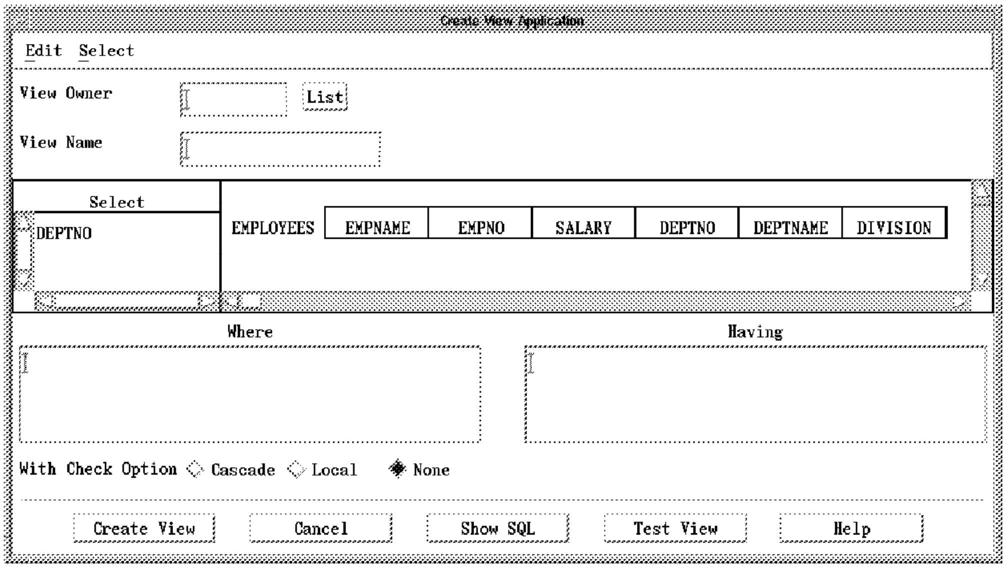


Figure 8. Base column added to SELECT list box

Please note: At any time, you can see if your query will return values correctly by using the **Test View** push button. When you select **Test View**, a window opens and displays the values that result from your subselect as currently defined in the Create View Application window.

3 *If you want view columns that map to system variables:* select the **Select** menu at the top of the window and select a system variable in that menu. The name of the view column that maps to that system variable is added in the Select section of the window.

4 Add the rest of the columns you want for the SELECT clause.

Working with Columns in the SELECT Clause

There are a number of actions you can perform on columns in the SELECT list box. You can:

- Rearrange their order. The order the columns are listed in the SELECT list box is the order they will appear in the SELECT clause of the CREATE VIEW statement.
- Delete them from the SELECT clause. You can delete a single column, or delete all of the columns at once.
- Apply functions or operators to them.

- Edit the columns.
- Add columns to a HAVING clause. See "Adding HAVING clauses" on page 234 for information on adding columns to a HAVING clause.

To rearrange the order of the SELECTed columns

1 Position the cursor over the column you want to move and hold down mouse button 1.

2 Drag the column on top of the column before which you want to place the dragged column.

3 Release the mouse button. Now, the column you dragged is above the other column in the SELECT list.

To delete all of the columns in the SELECT list box

1 Click on the **Select** menu at the top of the window

2 Select **Delete All**.

All of the columns are removed from the SELECT list box.

To delete, apply functions or operators to, or edit a column

1 In the Select list box, position the cursor over the column you want to work with and hold down the right-hand mouse button. A menu opens (see Figure 9 on page 228).

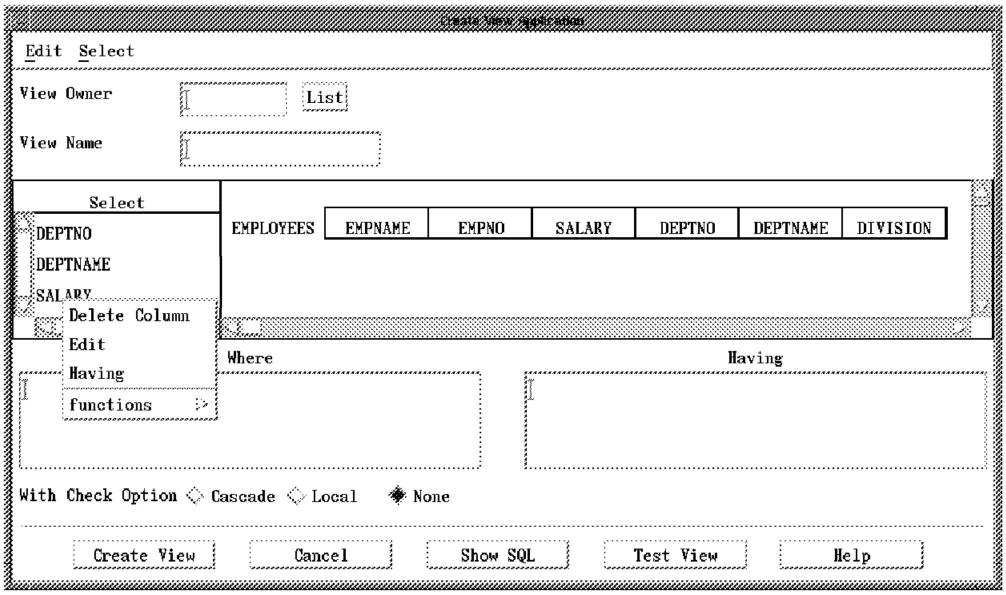


Figure 9. Menu for SELECTed columns

2 Depending on what you want to do to the column, select the appropriate menu choice while holding down the right-hand mouse button.

- *To remove the column from the SELECT clause:* select **Delete Column**.
- *To apply an operation or function to the column:* hold down the mouse button and position the cursor over **functions**. A menu opens that lists those functions and operators that can be applied to that column. The functions that are displayed are appropriate to the selected column's data type. For example, if the data type is NUMBER, functions such as AVG, MAXIMUM, and MINIMUM will be displayed, along with operators for addition, subtraction, multiplication, and division.

  Select the function or operator you want to apply to the column. When you do this, depending on the function or operation, a window might open. See the online help for more information on any displayed windows.
- *To edit a column:* select **Edit**. The Edit Attribute window opens. For information on valid modifications you can make, see the online help for the Edit Attribute window.

In our example on page 222, we want the average salary of each department. So, we open the menu for the EMPLOYEES.SALARY column, select **functions**, and select **Avg** from the list. The SELECT list box is updated to show that we have applied the Avg function to the column. If we wanted to change this, we could use the **Edit** choice on the pop-up menu to delete the avg text in front of the column name.

GROUP BY clauses:

When you include these in a SELECT clause—

- Columns qualified by an aggregate built-in function
- Columns containing repeated values

—DataHub automatically creates a GROUP BY clause that references those columns with repeated values. The resulting view shows information that the function returns for each set of repeated values.

In our example on page 222, we want the average departmental salary from the employees table. The department number (DEPTNO) and department name (DEPTNAME) are also in the SELECT list box. We apply the AVG function to the EMPLOYEE.SALARY column. DataHub automatically generates a GROUP BY clause that references DEPTNO and DEPTNAME. (To see this clause, select the **Show SQL** push button.)

If there are three departments with DEPTNO=30, 40, and 50, then the resulting view will show the average salary for the three departments, grouping them by number. For example,

DEPTNO	DEPTNAME	SALARY
30	Benefits	20000
40	Hiring	50000
50	EEO	22000

If you have a GROUP BY clause, you can qualify what groups to have in the view by using a HAVING clause. For more information, see "Adding HAVING clauses" on page 234.

Adding Where clauses

You can write WHERE clauses in the Create View Application window in these ways:

- Use the mouse to write WHERE clauses and generate logical operators. Instructions follow in the next section.
- Type it directly in the **Where** box of the View Application window. Be sure to click on the box first.

- Type it at the Create View window. See "Completing the View Definition" on page 235 for more information.
- Paste it in the **Where** box using text copied from the **Having** box. If you already have a HAVING clause in the **Having** box and you want to use this clause as a model for the WHERE clause, you can use your workstation's copy-and-paste operation to copy the HAVING clause to the **Where** box.
- Add a join. A *join* is a special type condition in a WHERE clause. Joins are discussed in "Adding Join Conditions" on page 231.

To include a base column in a WHERE clause

1 Position the cursor over the column you want to add for the WHERE clause and hold down the right-hand mouse button. A menu opens (see Figure 7 on page 225).

2 Holding down the right-hand mouse button, select the **Where** choice from the menu.

A window opens and displays a set of operators.

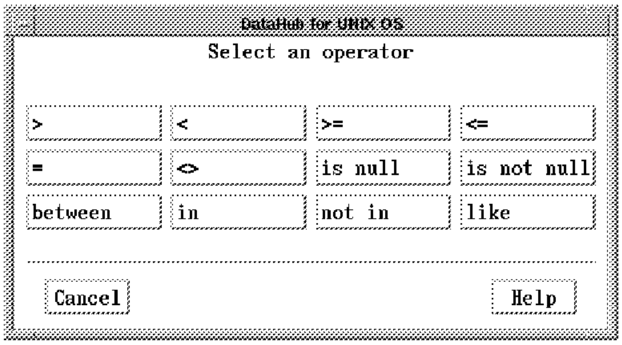


Figure 10. Select an Operator Window

3 From the Select an Operator window, click on the box that contains the operator you want to have in your condition. The Operand window (Figure 11 on page 231) opens.

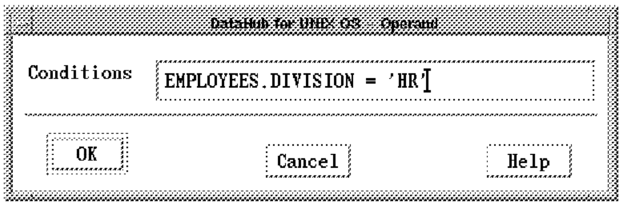


Figure 11. Operand Window

The Operand window shows the first part of the condition for your WHERE clause: the name of the base column you selected and the operator you clicked on in the Select an Operator window.

> **Exception:** If you clicked on the box containing `is null` or the box containing `is not null`, the Operand window does not open. Instead, the completed condition appears in the **Where** box.

4 From the Operand window, complete the condition and click on the **OK** push button. The condition appears in the **Where** box in the Create View Application window.

5 If you want logical operators in your WHERE clause,

- **a** In the **Where** box, move the mouse cursor to the place where an operator should be inserted. An I-beam cursor is displayed at the place.
- **b** Select the **Edit** pull-down menu of the Create View Application window. The menu choices are **And**, **Or**, and **Not**.
- **c** Select the operator you want. The appropriate syntax is added to the condition in the Where box.

In our example (see page 222), the WHERE clause is `WHERE employees.division = 'HR'`.

Adding Join Conditions

A join condition is a condition where two tables or views are brought together and compared; rows from one are selected when columns from that one match columns (according to a specified condition) from the other one.

For example, this join returns the employee name and the name of the manager of the department the employee belongs to. The employees table contains the name of employees and their department number; the departments table contains the department number and the name of the department manager.

```
SELECT empname, manager
  FROM employees, departments
  WHERE employees.deptno = departments.deptno
```

Because the data is in two different tables, the join condition allows you to get the data according to the condition employees.deptno = departments.deptno.

> **Tip:** There are three ways to write a join condition:
>
> - Type it in the **Where** box of the Create View Application window. Be sure to select the Where box before entering text.
> - Type it in the Create View window. See "Completing the View Definition" on page 235 for more information.
> - Use the mouse to create the join in the Create View Application window. See the following instructions.

There are two ways to join two different base objects by using the mouse.

- By dragging and dropping one base column onto the other, or
- Selecting one base column and using the **Join** option on the menu for the other base column.

You might want to use the second method if your base objects have many columns.

To join base columns by dragging and dropping base columns

1 In the base object area, position the cursor over one of the base columns you want to join and hold down mouse button1.

2 Drag the column onto the other base column.

An arrow representing the join appears. A square covers part of the arrow's shaft (see Figure 12 on page 233).

To join base columns by using menus

1 Select one of the base columns you want to join.

2 Position the cursor over the other base column and hold down the right-hand mouse button. A menu opens.

3 Select **Join** from the open menu. An arrow representing the join appears. A square covers part of the arrow's shaft (see Figure 12 on page 233).

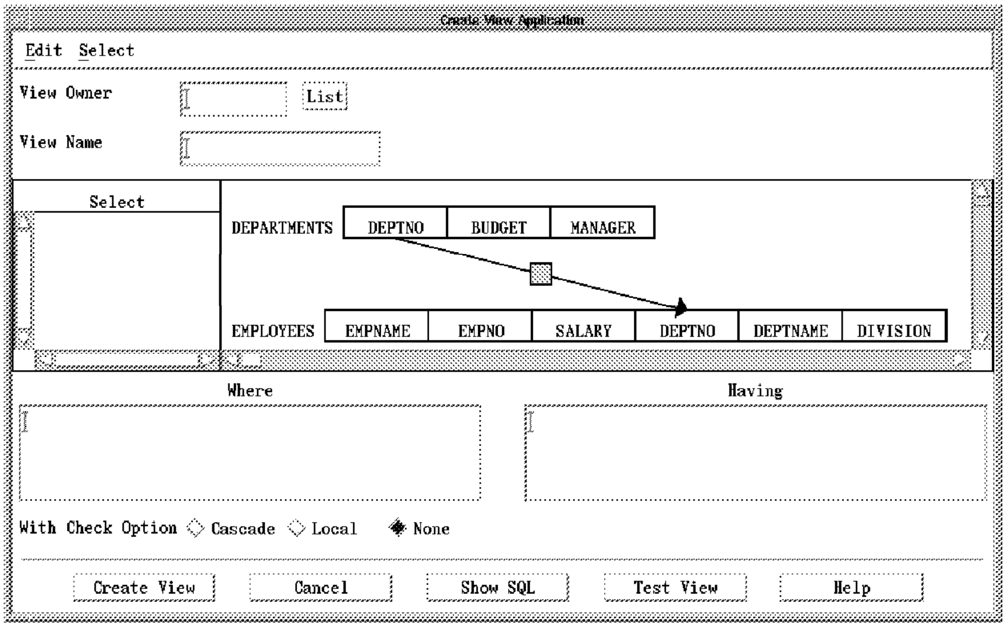


Figure 12. Create View Application window with a join condition

Tip: To see the resulting WHERE condition, select the **Show SQL** push button.

Once you have joined the base objects using one of the two methods, the operator in the WHERE condition is an equals sign. If you want to change the operator, or any other part of the condition, you can do so by editing the SQL for the final view; see page 236 for more information.

You can also join an object to itself; this is called a *self-join*.

To join an object to itself

1 Position the cursor over the name of the base object and hold down the right-hand mouse button. A menu opens.

2 Select **Self-Join** and release the mouse button. A second image of the base object is displayed in the base object area.

3 Join the base columns by one of the two methods as previously described.

To delete a join condition:

1 Position the cursor over the square on the shaft of the arrow that represents the join and hold down the right-hand mouse button.

A menu opens.

2 Select **Delete Relation** from the menu.

The join relation is deleted.

Adding HAVING clauses

You can write a HAVING clause in these ways:

- Type it in the **Having** box of the Create View Application window. Be sure to select the Having box before entering text.
- Type it in the Create View window. See "Completing the View Definition" on page 235 for more information.
- Paste it in the **Having** box using text copied from the **Where** box. If you already have a WHERE clause in the **Where** box and you want to use this clause as a model for the HAVING clause, you can use your workstation's copy-and-paste operation to copy the WHERE clause to the **Having** box.
- If DataHub generated a GROUP BY clause as described in 229, you can use the mouse to write an associated HAVING clause. See the following instructions.

To add a HAVING clause by using the mouse

1 In the Select list box, position the cursor over the column you want to work with and hold down the right-hand mouse button. A menu opens (see Figure 9 on page 228).

2 Select **Having** from the menu. The Select an Operator window (Figure 10 on page 230) opens.

3 From the Select an Operator window, click on the box that contains the operator you want to have in your condition. The Operand window (Figure 11 on page 231) opens. The Operand window shows the first part of the condition for your HAVING clause: the name of the base column you selected and the operator you clicked on in the Select an Operator window.

Exception: If you clicked on the box containing `is null` or the box containing `is not null`, the Operand window does not open. Instead, the completed condition appears in the **Having** box.

4 From the Operand window, complete the condition and click on the **OK** push button. The condition appears in the **Having** box in the Create View Application window.

5 If you want logical operators in your HAVING clause,

   a In the **Having** box, move the mouse cursor to the place where an operator should be inserted. An I-beam cursor is displayed at the place.

   b Select the **Edit** pull-down menu of the Create View Application window. The menu choices are **And**, **Or**, and **Not**.

   c Select the operator you want. The appropriate syntax is added to the condition in the Having box.

Testing Your View

At any point while you are constructing your query, you can test the results of the query by using **Test View** push button.

To test your view

1 Select the **Test View** push button.

   A window opens that shows the results of the view as it is currently defined.

2 Click on **OK** to close the results window and return to the Create View Application window.

Completing the View Definition

Once you are finished building your query in the Create View Application window, you complete the view definition so that DataHub can begin creating the view.

To complete the information in the Create View statement

1 In the Create View Application window, specify appropriate values for the parameters at the top of the window. See the online help for information on the parameters.

2 Optional: Specify the With Check Option. See the online help for more information.

3 Select the **Create View** push button. The Create View window opens (see Figure 13 on page 236).

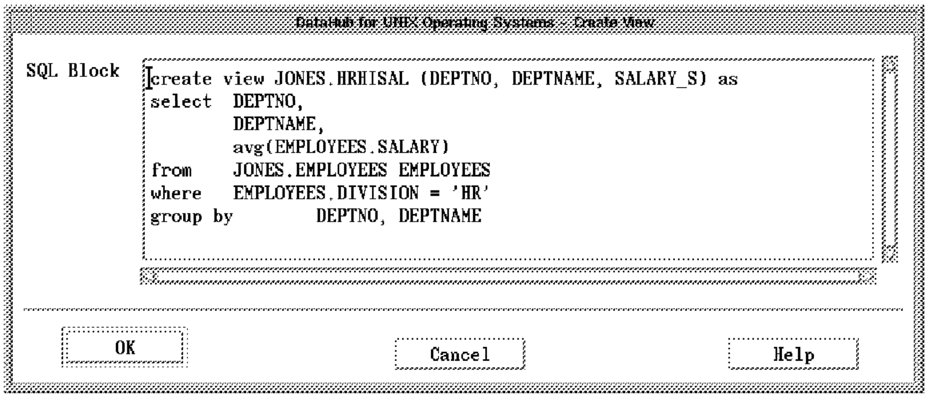


Figure 13. Create View window

This window presents an SQL Block that contains the SQL for the view as currently defined by your choices in the Create View Application window. Note that the current names of the view's columns appear in the SQL. To change the names of the columns, you must edit the SQL.

4 *To edit the SQL:*

a Click within the SQL Block to make it active. The cursor changes to an I-beam.

b Position the cursor at the point where you want to edit the SQL. You can insert or delete text.

5 When finished, click on **OK**.

DataHub creates the view.

Updating View Definitions

Use the DataHub **Update** action to update the definition of a view.

To update a view definition

1 From the DataHub for UNIX window, select the particular view you want to update.

2 Click on the **Update** push button. The Update View window opens. This window allows you to change the definition of the view.

For information about this window, see the online help.

3 When you have made your changes, commit the changes by clicking on the **OK** push button.

Dropping Views

Use the DataHub **Drop** action to drop a view.

To drop a view

1 From the DataHub for UNIX window, select the particular view you want to drop.

2 Click on the **Drop** push button. A question box will appear asking if you want to drop the view. From this window

3 Click on the **Yes** push button. The view is dropped, and the list in the DataHub for UNIX window is automatically refreshed to indicate the drop.

Renaming Sybase Views

Syb You can rename a Sybase view by selecting the view occurrence and clicking on the **Rename** push button.

Managing Comments for DB2 Version 2 Views

DB2v2 You can create, change, and replace comments for a DB2 Version 2 view by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the view occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Chapter 18. Transferring DB2 Data

DB2 Transferring DB2 data involves importing and exporting data. For DB2 Version 2, you can also transfer data by loading. This chapter discusses importing and exporting DB2 data, and loading DB2 Version 2 data. For information on importing and exporting for other RDBMS types, see

- Ing "Exporting and Importing INGRES Data" on page 126.
- Ora "Transporting ORACLE Data" on page 141.

Importing DB2 Data

The DB2 IMPORT utility inserts data from an input file into a table or view. If the table or view receiving the imported data already contains data, you can either replace or append the existing data with the data in the file.

You can use the IMPORT utility to copy data created by a separate application, such as a spreadsheet program or another database program, into a table or view.

FYI: If the existing table is a parent table that contains a primary key that is referenced by a foreign key in a dependent table, the parent table's data cannot be replaced, only added to.

Use the DataHub **Import** action to invoke the IMPORT utility.

To import data into a table or view

1 Select the table or view into which you want to import the data and click on the **Import** push button. A window opens.

2 Specify the parameters for importing the table or view. For an explanation of the parameters, see online help.

You must specify:

Import File Enter the path and name of the operating system file that stores the data for importing.

Import File Type
Click on the right-hand button and select the format of the data in the import file. This format can be Integrated Exchange Format, Work-Sheet Format, Non-delimited ASCII Format, or Delimited ASCII Format.

For an explanation of the formats, see online help or the *DB2 Administration Guide*.

Import Mode
Specify whether the data in the input file is to be inserted into the table or view, or whether it is to update, replace, or append existing data in the table or view.

Click on the **Show** push button to open a window that displays the SQL command for the import process.

Click on the Show Messages File check box to have the messages from the export process shown in a window.

3 When finished, click on the **OK** push button.

If you selected the Show Messages File check box, a window opens with the results of the import process.

Exporting DB2 Tables or Views

The DB2 Version 2 EXPORT utility exports data from a database into an operating system file. The output file has the format specified by the data format parameter.

You can use the EXPORT utility to copy data from a table or view to be used in another database or by a spreadsheet program.

Use the DataHub **Export** action to invoke the EXPORT utility.

To export data into an operating system file

1 Select the table or view from which you want to export the data and click on the **Export** push button. A window opens.

2 Specify the parameters for exporting the table or view. For an explanation of the parameters, see online help.

You must specify:

Output File Enter the path and name of the operating system file that stores the exported data.

Export File Type
Click on the right-hand button and select the format of the data for the file. This format can be Integrated Exchange Format, Work-Sheet Format, or Delimited ASCII Format.

For an explanation of the formats, see online help or the *DB2 Administration Guide*.

You can edit the select statement by clicking in the rectangular area and entering text.

> **But first:** Select the **Async. Exec.** parameter to indicate you want the DataHub for UNIX Operating Systems window to be available for you to perform other tasks during the export process. Select the **Synchronous** option if you want to wait on this Export window until the export process is finished.
>
> If you choose Asynchronous, you cannot choose to Show Messages File.

Click on the **Show** push button to open a window that displays the SQL command generated by DataHub for the export process.

Click on the Show Messages File check box to have the messages from the export process shown in a window.

3 When finished, click on the **OK** push button.

If you selected the Show Messages File check box, a window opens with the results of the export process.

> **FYI:** In DB2 Version 1, export files for a database are owned by the user ID that owns the instance that manages the database.

Loading DB2 Version 2 Data

The DB2 Version 2 LOAD utility is a high-speed alternative to the IMPORT utility. LOAD is intended for either bulk loading of new tables or appending large amounts of data to existing tables.

The table and any indexes on the table must already exist. If the table receiving the data already contains data, you can replace or append the existing data with the data in the file.

> **FYI:** If the existing table is a parent table that contains a primary key that is referenced by a foreign key in a dependent table, the parent table's data cannot be replaced, only added to.

All data types can be loaded by the LOAD utility including large objects (LOBs) and user-defined types (UDTs). See the *DB2 Administration Guide* for restrictions on the use of the LOAD utility.

Use the DataHub **Load** action to invoke the LOAD utility.

To load data into a table

1. Select the table into which you want to load data and click on the **Load** push button. The Load Table window opens.
2. Specify the parameters for loading the table. For an explanation of the parameters, see online help.

   You must specify:

   Load From: Click on the **Add** push button to add file names. The Specify File/Directory Name window opens.

   Enter the path and file name and click on the **OK** push button. When finished adding file names, click on the **Cancel** push button.

   Use the **Change** and **Drop** push buttons to change or drop the files in the Load From area. Select the file name and click on the appropriate push button.

   File Type Click on the right-hand button and select the format of the data for the file. For an explanation of the formats, see online help or the *DB2 Administration Guide*.

   Mode Click on the push button to specify how the data in the input file is to be put into the table.
3. When finished, click on the **OK** push button.

Chapter 19. Managing Indexes

An *index* is typically used to get more efficient access to rows in a table. Unique indexes ensure that no two rows of a table are the same. An *index key* is a column or an ordered collection of columns on which an index is defined.

Managing indexes involves:

- Obtaining current information about existing indexes
- Creating indexes
- Dropping indexes
- Ing Ora Updating indexes
- Syb Renaming indexes
- DB2v2 Managing comments for indexes

The following subsections discuss these tasks for table indexes. For ORACLE, you can also display the indexes for a cluster; this is discussed in "Obtaining Information about ORACLE Clusters" on page 247.

Obtaining Current Information on Indexes

You can obtain several kinds of current information about indexes. Examples are:

- What indexes exist for a table
- DB2v2 Definitions for DB2 Version 2 indexes
- DB2v1 Statistics information for DB2 Version 1 indexes
- Ora Segment information for ORACLE indexes
- Syb Statistics information for Sybase indexes

The following text explains how to find the kinds of information listed above.

To display the indexes for a table

1. Select the table on the object tree.
2. Click on the **Display** push button. The Display Types window opens.
3. Select **Index** and click on the **OK** push button. The Index object type and a list of indexes for that table appear under the table.

To show the definition of a DB2 Version 2 index

1. Select the index occurrence and click on the **Show** push button. The Index Information window opens and displays information about the index. It shows the column names for the index, whether there is a unique rule, and the order (ascending or descending) for the index.

For more information about this window, see the online help.

2 Click on **OK** to close the Index Information window.

DB2 For DB2 databases, the database manager maintains statistics about the physical characteristics of tables and indexes and uses them to determine the best way to access the data. DataHub uses the RUNSTATS utility to update these statistics in the system catalog tables. For information on these statistics, see the *Administration Guide* for your DB2 installation.

DB2v2 To use RUNSTATS on a DB2 Version 2 index, use the DataHub action **Runstats** on the table that has the index. See "Gathering Statistics for Tables" on page 211 for more information.

To use RUNSTATS on a DB2 Version 1 index

1 Select the index occurrence and click on the **Runstats** push button. A window opens where you can specify whether to update and show statistics, only update statistics, or only show existing statistics. You can also specify whether to write the new statistics or not.

For information on the options, see the online help.

2 Click on the **OK** push button. If you selected Show Statistics or Update and Show Statistics, a window opens with statistics on the index.

3 To close this window, click on the **OK** push button.

To open a window containing segment information for an ORACLE index

Ora On the object tree, select the ORACLE index and click on the **Segment Info** push button. The Segment Info window opens.

To display statistics information for a Sybase index

Syb Select the Sybase index and click on the **Statistics** push button.

Creating Indexes

Using the DataHub **Create** action on the Index object type, you can create indexes for your tables. You do not need to know the names of the table columns before creating the index; if you click on the **Add** push button in the Create Index window, DataHub will display the columns and you can select them from a list.

To create a table index

1 Display the index object type and current list of indexes, if any exist, for the table you want to create an index for. See "Obtaining Current Information on Indexes" on page 242 for how to display indexes.

2 Select the Index object type and click on the **Create** push button. The Create Index window opens.

3 Fill in the various parameters for the index you want to create. For an explanation of the parameters, see the online help.

To display a list of the names of the columns in the table, click on the **Add** push button in the Column Names field. You can select the columns for your index from the list of columns.

When finished, click on the **OK** push button to commit your choices. The index you just created appears in the list of indexes for the table.

Dropping Indexes

Use the DataHub **Drop** action to drop an index.

To drop an index

1 On the object tree, select the index.
2 Click on the **Drop** push button. A question box appears asking if you want to drop the index. From this window
3 Click on the **Yes** push button. If you have sufficient privileges for this index, the index is dropped and the object tree is automatically refreshed.

Updating INGRES and ORACLE Indexes

Ing Ora You can update INGRES and ORACLE indexes by selecting the index occurrence and clicking on the **Update** push button.

For an INGRES index, the **Update** action lets you:

- Change the storage structure for the index
- Move the index to new locations
- Reorganize the index to use additional locations

For an ORACLE index, the **Update** action lets you change the storage parameters for the index.

To update an index

1 Select the index occurrence and click on the **Update** push button. The Change Index window opens.

2 Change the parameters you want. For an explanation of the parameters, see online help.

3 When finished, click on the **OK** push button.

Renaming Sybase Indexes

Use the DataHub **Rename** action to rename a Sybase index.

To rename a Sybase index

1 Select the index occurrence and click on the **Rename** push button.

2 Enter a name and click on **OK**.

Managing Comments for Indexes

DB2v2 You can create, change, and replace comments for a DB2 Version 2 index by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the index occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Chapter 20. Managing Sequences for ORACLE Tables

Ora A sequence is a database object from which multiple users can generate unique integers. You can use sequences to automatically generate values for parameters such as primary key values and archive log file names.

Managing sequences involves:

- Listing the sequences on a database
- Creating, altering, and deleting a sequence

To see the contents of a sequence, use the **Alter** action.

To list the sequences on a database

1 Double-click on the database on the object tree, select **Sequences** from the list, and click on **OK**. The Filter window opens.

2 Specify a filter to display only the sequences that satisfy that filter or leave the fields blank. Click on **OK**. The Sequences object type and any existing sequences are added to the object tree.

To create a sequence

1 Select the Sequences object type and click on **Create**. The Sequences window opens. Specify the parameters in the window. See the online help for explanations of the parameters.

2 When finished, click on **OK**.

To alter a sequence

1 Select the sequence and click on **Alter**. The Sequences window opens.

2 Modify the parameters you want to change. For descriptions, see "To create a sequence" or the online help.

3 Click on **OK** when finished.

To delete a sequence

1 Select the sequence and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**.

Chapter 21. Managing ORACLE Clusters

Ora A *cluster* is an ORACLE schema object that contains one or more tables that have one or more columns in common. Rows of tables that share the same values in these common columns are physically stored together within the database.

Clusters provide more control over the physical storage of rows within the database and can reduce both the time it takes to access the tables and the space needed to store the tables.

Managing ORACLE clusters involves:

- Obtaining information about the clusters on a particular database
- Creating clusters
- Deleting clusters
- Altering clusters

Obtaining Information about ORACLE Clusters

Using DataHub, you can list the clusters on an ORACLE database, and show segment information for those clusters. You can also list the tables and indexes for a particular cluster.

To display the clusters on an ORACLE database

1 Ora Double-click on the database, select **Cluster** from the list for object types, and click on **OK**. The Filter window opens.

2 Specify a filter criterion or not and click on **OK**. The Cluster object type and any existing clusters are displayed on the object tree.

To display segment information for a cluster

Select the cluster occurrence and click on the **Segment Info** push button. The Segment Info window opens and displays information about the cluster.

To display the tables for a particular cluster

Double-click on the cluster occurrence, select Table, and click on **OK**. The Table object type and the tables (if any) for this cluster are displayed on the object tree.

To display the indexes for a particular cluster

Double-click on the cluster occurrence, select Index, and click on **OK**. The Index object type and the indexes (if any) for this cluster are displayed on the object tree.

Creating ORACLE Clusters

Once a cluster is created, tables can be created in the cluster. However, before any rows can be inserted into the clustered tables, you must create a cluster index.

A *cluster key* is the column, or group of columns, that the clustered tables have in common. The columns of the cluster key are specified when creating the cluster. The same columns are subsequently specified when creating every table added to the cluster.

To create a cluster

1. Select the Cluster object type and click on **Create**. The Define Cluster window opens.
2. Specify the parameters for the cluster. For an explanation of the parameters, see the online help.
3. When finished, click on the **OK** push button.

Creating Tables in Clusters

After you create the cluster, you can create tables in that cluster by displaying the Table object type under the cluster, selecting the Table object type, and clicking on the **Create** push button. See "Creating and Dropping Tables" on page 213 for more information on creating a table.

For each column specified as part of the cluster key, every table created in the cluster must have a column that matches the size and type of the column in the cluster key.

Creating Indexes in Clusters

Before you can insert rows into the cluster tables, you must create a cluster index. The cluster index contains an entry for each cluster key value.

Create a cluster index by displaying the Index object type under the cluster, selecting the Index object type, and clicking on the **Create** push button.

Deleting ORACLE Clusters

Use the DataHub **Drop** action to remove a cluster. You must remove all the objects stored in the cluster before dropping it.

To delete an ORACLE cluster

1 Select the cluster occurrence and click on the **Drop** push button. A question box opens asking for confirmation.

2 Click on **Yes** to drop the cluster.

Altering ORACLE Clusters

You can alter the name, owner, and certain storage parameters for a cluster occurrence by selecting it and then clicking on the **Alter** push button.

To alter an ORACLE cluster

1 Select the cluster occurrence and click on the **Alter** push button. The Alter Cluster window opens.

2 Enter new values for the parameters you want to change. For information on the parameters in this window, see the online help.

When you change a parameter, you can see the SQL command for altering the cluster by clicking on the **Show** push button. A window opens and displays the SQL command. Click on **OK** to close the window.

3 When you have entered the new values in the Alter Cluster window, click on the **OK** push button to commit the changes.

Chapter 22. Maintaining Data Integrity

Data integrity is important in any database management system. It refers to the accuracy of the values within database tables. Data integrity can be ensured through the use of

- DB2 Syb Ora Primary keys
- DB2 Syb Ora Foreign keys
- Syb DB2v2 Ora Checks
- DB2v2 Constraints
- Ing Integrity constraints
- Syb Sybase Rules
- Syb Defaults
- Syb DB2v2 User-defined data types
- DB2v2 Syb Ora Triggers
- Ing Rules and Events
- Ora Synonyms
- DB2v2 Aliases

Managing Primary Keys

DB2 Syb Ora The *primary key* of a table is a column or set of columns that are defined to be constrained to have unique and non-null values. This constraint ensures unique identification of the table's rows.

Primary keys are used to maintain the integrity of the database and for quick access to rows; indexes for primary keys are automatically created.

Managing primary keys involves:

- DB2 Syb Ora Creating and dropping primary keys
- Ora Enable and disable primary keys
- DB2v2 Managing comments for a primary key

Creating and Dropping Primary Keys

You can create a primary key either at table creation time or, for an existing table, using the DataHub **Create** action.

To create a primary key for an existing table

1. Display the table for which you want to create a primary key and display the Primary Key object type on the object tree.
2. Select the Primary Key object type and click on the **Create** push button. A window opens in which you can specify a primary key. For an explanation of the parameters, see the online help.

Ora To enable the primary key at creation, select the Enable parameter.

3 When finished, click on **OK**.

To drop a primary key

1 Select the primary key and click on **Drop**. A question box opens asking for confirmation.

2 Click on **Yes**.

Enabling and Disabling ORACLE Primary Keys

Ora You can enable or disable an ORACLE primary key by selecting the primary key occurrence and clicking on the **Enable** or **Disable** push button.

A message in the status line reports on success or failure.

Managing Comments for DB2 Version 2 Primary Keys

DB2v2 You can create, change, and replace comments for a DB2 Version 2 primary key by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the primary key occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Managing Foreign Keys

DB2 Syb Ora The *foreign key* of a table is a column or set of columns in the table whose values are required to match at least one foreign key value of a row of its parent table.

Managing foreign keys involves

- DB2 Syb Ora Creating and dropping

- Syb Renaming
- DB2v2 Managing comments

Creating and Dropping Foreign Keys

You can create a foreign key either at table creation time or, for an existing table, using the DataHub **Create** action.

To create a foreign key

1. Display the table for which you want to create a foreign key and display the Foreign Key object type on the object tree.
2. Select the Foreign Key object type and click on the **Create** push button. A window opens in which you can create a foreign key.
3. Specify the parameters in the window. For explanations of the parameters, see the online help.
4. When finished, click on **OK**.

To drop a foreign key

1. Select the foreign key and click on **Drop**. A question box opens asking for confirmation.
2. Click on **Yes**.

Renaming a Foreign Key

Syb You can rename a Sybase foreign key by selecting the foreign key and clicking on the **Rename** push button.

Managing Comments for DB2 Version 2 Foreign Keys

DB2v2 You can create, change, and replace comments for a DB2 Version 2 foreign key by using the DataHub **Comment** action.

To create, change, or replace a comment

1. Select the foreign key occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

   If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2. To add a new comment, enter it in the New Comment field.
3. To change an existing comment, edit the comment in the New Comment field.
4. To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Managing Checks

Syb DB2v2 Ora A check is an integrity constraint on the values that exist in the columns of a table. Checks are generally used to enforce business rules not covered by key uniqueness or referential integrity constraints. This includes specifying a range or domain that valid entries must exist in. For example, you can define checks on the EMPLOYEE table that specify that the job of an employee can only be one of ‘Sales’, ‘Mgr’, or ‘Clerk’. Each entry in the job column must exist in this domain to be valid.

Managing checks involves:

- Listing the existing checks for a table
- Creating and deleting checks
- Syb Renaming a check
- Ora Updating a check
- DB2v2 Managing comments for a check

To list the checks for a table

1 Display the table on the object tree (see “Displaying Current Information about Tables” on page 211). Double-click on the table, select **Check** from the list, and click on **OK**. The Check object type and any existing checks are added to the object tree.

You can create a check either at table creation time or, for an existing table, using the DataHub **Create** action.

To create a check

1 Select the Check object type and click on **Create**. The Table Check window opens.

2 Specify the Constraint Name and the conditions for the check. For explanations of the window parameters, see the online help.

3 When finished, click on **OK**.

To delete a check

1 Select the check and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**.

Updating Checks

Ora You can update an ORACLE check by selecting the check occurrence and clicking on the **Update** push button.

Renaming Checks

Syb You can rename a Sybase check by selecting the check occurrence and clicking on the **Rename** push button.

Managing Comments for DB2 Version 2 Checks

DB2v2 You can create, change, and replace comments for a DB2 Version 2 check by using the DataHub **Comment** action.

To create, change, or replace a comment

1. Select the check occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

   If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2. To add a new comment, enter it in the New Comment field.
3. To change an existing comment, edit the comment in the New Comment field.
4. To replace an existing comment, delete the comment in the New Comment field and enter a new one.
5. When finished, click on the **OK** push button.

Managing Constraints

DB2v2 With DataHub, you can display at once all the constraints (such as primary keys, foreign keys, checks) that exist on a DB2 V.2 database.

For each constraint, you can create or change a comment, or drop the constraint by using the **Comment** and **Drop** push buttons.

You can also set all the constraints for a table at once by using the **Set Constraints** action.

To display all the constraints on the database, double-click on the database, select **Constraint** from the list, and click on the **OK** push button. All of the constraints that exist in that database are displayed on the object tree.

To set the constraints for a DB2 Version 2 table

1 Select the DB2 Version 2 table occurrence and click on the **Set Constraints** push button. The Set Constraint Options window opens.

2 Specify the parameters you want. For explanation of the parameters, see the online help.

Managing INGRES Constraints

Ing For INGRES databases, integrity constraints are used for checking data entries for valid values. Integrities can be defined for particular columns in tables by the table owner.

Tips:

When INGRES creates an integrity constraint for a table column, INGRES checks to see that the column's rows satisfy the condition specified in the constraint. If all rows satisfy the condition, the constraint is accepted. Otherwise, it is rejected. Be aware that:

- If the table has many rows, the checking might take some time. As a result, you might not get control back immediately after you click on the **OK** push button (step 4 below).
- INGRES does not treat null as a value. Therefore, if a row is null, the constraint will be rejected. For example, take the following constraint:

  create integrity on "table1" is col1 <= 5

  This constraint would be rejected if "table1" has one or more rows in which column col1 is null. For more information, see your INGRES documentation.

To display existing integrity constraints on an INGRES table

1 Display the table on the object tree.

2 Double-click on the table, select **Integrity** and click on **OK**. The Integrity object type and any integrities already defined are displayed on the object tree.

To create integrity constraint on an INGRES table

1 Display the Integrity object type on the object tree from a table occurrence.

2 Select the Integrity object type and click on the **Create** push button. The Create Integrity window opens.

3 Specify the parameters in the Create Integrity window. For detailed descriptions of the parameters, see online help.

4 Click on **OK** when finished.

To update an integrity constraint

1 Select the integrity object occurrence and click on the **Update** push button. A window opens that has the SQL statement that defines the integrity constraint.

2 Change focus to the field that displays the SQL statement by clicking on that field. Alter the SQL statement.

3 Click on **OK** when finished.

To remove an integrity constraint

1 Select the integrity constraint and click on the **Drop** push button.

Managing Sybase Rules

Syb A Sybase rule ensures data integrity at the column level of a table. A rule is an expression that defines the format or content of the data for a specific column or user-defined data type.

For example, if DEPT_NUMBER is the column of a table, and all department numbers are of the form X99 where X is any letter A through Z and 99 is any two-digit number, then you could set a rule on that column based on the expression:

```
@deptno like [A-Z][0-9][0-9]
```

which says to allow any letter A through Z as the first item, and any number 0 through 9 as the second and third items. The value to be used in evaluating the expression is @deptno. This value does not have to match the column name; the rule uses the column to which it is bound.

With rules, you can incorporate your business requirements into your database.

To display existing rules on a Sybase database

1 Double-click on the database occurrence. The Display Types window opens.

2 Select Rule and click on the **OK** push button. A filter window opens.

If you want, enter an owner or name.

3. Click on **OK**. The Rule object type and any rule occurrences on this database are displayed on the object tree.

To use a rule, you must first define it and then bind it. You can bind a rule to one or more columns of a table, or to a user-defined data type.

To define a rule on a database

1. Display the Rule object type on the object tree from a database occurrence.
2. Select the Rule object type and click on the **Create** push button. The Create Rule window opens. Enter the name of the owner, the rule name, and the conditional expression for the rule.
3. Click on **OK** when finished.

For examples of rules, see your Sybase documentation.

To make the rule active on a table, you have to bind it.

To bind a rule

1. Select the rule occurrence and click on **Bind**. The Choose Bindtype window opens.
2. Select the bindtype you want and click on **OK**.
   - If you selected the Bind to Column bindtype, the Bind Rule to Column window opens.

     Enter the table and column names and click on **OK**.
   - If you selected the Bind to User-defined Datatype, the Bind Rule to Data Type window opens.

     Enter the data type owner and data type name in the fields, or select from a list by using the **List** push buttons. Click on the Future Only check box if you want to specify rule use on this type for the future only.

   For information on parameters in these windows, see the online help.

To update the SQL expression that defines the rule, you must unbind it first from the table or data type. To unbind a rule, you must display the rule for the particular table or data type.

To unbind a rule

1 Display the table or data type on the object tree.

2 Double-click on the table or data type occurrence, select **Rule** from the list, and click on **OK**.

3 Select the rule occurrence and click on the **Unbind** push button.

For a rule on a data type, the Unbind Rule from Data Type window opens. Specify Future Only by clicking on the check box and then click on the **OK** push button.

For a rule on a table, a question box opens asking for confirmation. Click on **OK** to confirm.

To rename a rule

Select the rule occurrence and click on the **Rename** push button.

To show information about a rule

Select the rule occurrence and click on the **Information** push button.

You must unbind a rule before you can update the SQL expression that defines it.

To update the SQL expression for a rule

Select the rule and click on the **Update** push button.

To drop a rule

Select the rule occurrence and click on the **Drop** push button.

To display the tables or data types where a rule is used

Double-click on the rule occurrence.

Managing Defaults

Syb Defaults are values which are inserted into rows in place of a null when a value is not provided. Defaults provide data integrity by ensuring there are no null values in a row. If you insert a specific value for the column, you override the default.

Defaults are useful when almost all of the entries in a table have a column value in common. You could set a default so that you do not have to enter the value explicitly every time.

For example, if your table consists of a list of customers and most of your customers are from Boston, you could have a default that sets the column for the city as

″Boston″

and override the default by explicitly setting the value for those customers who are not from Boston.

To display existing defaults on a Sybase database

1 Double-click on the database occurrence. The Display Types window opens.

2 Select Default and click on the **OK** push button. A filter window opens.

If you want, enter an owner or name.

3 Click on **OK**. The Default object type and default occurrences on this database are displayed on the object tree.

You must first define the default and then bind it to take effect. A default does not set the value for a column which has been defined as not null when an explicit null was inserted. If no value is defined to be inserted and the column does not allow nulls, the default value is inserted.

To define a default on a database

1 Display the Default object type on the object tree.

2 Select the Default object type and click on the **Create** push button. The Create Default window opens. Enter the name of the owner, the default name, and the conditional expression for the default.

3 Click on **OK** when finished.

To bind a default

1 Select the default occurrence and click on the **Bind** push button. The Choose Bindtype window opens.

2 Select the type you want and click on **OK**.

- If you selected the Bind to Column bindtype, the Bind Default to Column window opens.

  Enter the table and column names and click on **OK**.

- If you selected the Bind to User-defined Datatype, the Bind Default to Data Type window opens.

  Enter the data type owner and data type name in the fields, or select from a list by using the **List** push buttons. Click on the

Future Only check box if you want to specify rule use on this type for the future only.

For information on parameters in these windows, see the online help.

To unbind a default, you must display the default for the particular table or user-defined data type to which it is bound.

To unbind a default

1 Display the table or user-defined data type occurrence on the object tree.

2 Double-click on the table or user-defined type occurrence select **Default** from the list, and click on **OK**.

3 Select the default and click on the **Unbind** push button.

- If you selected a data type, the Unbind Default from Data Type window opens. If you want Future Only, select the check box. Click on the **OK** push button.
- If you selected a table, a question box opens asking for confirmation. Click on the **OK** push button to confirm unbinding.

To rename a default

Select the default occurrence and click on the **Rename** push button.

To drop a default

Select the default occurrence and click on the **Drop** push button.

You must unbind a default before you can update the SQL expression that defines it.

To update the SQL expression that defines a default

Select the default occurrence and click on the **Update** push button.

To display the tables or data types where a default is used

Double-click on the default occurrence.

Managing User-Defined Data Types

User-defined data types are created by combining values and rules with the existing database data types. These user-defined data types are useful in applications and tables where you consistently use the same rules and defaults. You can create a unique data type that imposes the rules and defaults.

For example, you can create a “zipcode” data type for use in a table of addresses using the rule

```
@zip like '[0-9][0-9][0-9][0-9][0-9]'
```

and binding this rule to a data type defined as “zipcode”.

DB2v2 User-defined data types ensure data integrity at the point of data entry. Any data value that is added to a column with a user-defined function defined is automatically checked for validity.

To display the data types on a Sybase or DB2 Version 2 database

1. Double-click on the database occurrence. The Display Types window opens.
2. Select Data type and click on the **OK** push button. The Data Type object type and any data type occurrences on this database are displayed on the object tree.

To define a data type for a database

1. Display the Data Type object type on the object tree.
2. Select the Data Type object type and click on **Create**. The Create Data Type window opens.
3. Enter the data type owner name, the name of the type, the data type for this unique data type, and other parameters you want to specify. For explanations of the parameters, see the online help.
4. Click on **OK** when finished.

After a data type is created, it will appear in the list of available data types to be used when you create the column definitions for a table.

You can bind rules and defaults to a user-defined data type by choosing the Bind to User defined Datatype when you bind the rule or default. You must ensure the rule or default expression is compatible with the data type.

To rename a Sybase data type

Syb Select the data type occurrence and click on the **Rename** push button.

To drop a data type

Syb DB2v2 Select the data type occurrence and click on the **Drop** push button.

To create or change a comment for a DB2 Version 2 data type

DB2v2 Select the data type occurrence and click on the **Comment** push button.

To display the rules or defaults used in a Sybase data type

Syb Double-click on the data type occurrence.

To display the functions that a DB2 Version 2 data type depends on

DB2v2 Double-click on the data type occurrence.

See the online help for more information on these actions.

Managing Triggers

DB2 Syb Ora A *trigger* defines a set of actions that are executed or *fired* when the associated table is modified by an insert, update, or delete action. Triggers are only defined on tables; however, if a view is modified by an insert, update, or delete action, the triggers of the view's tables are fired.

A trigger can be used to:

- Validate input data
- Generate a value for a newly inserted row
- Read from other tables for cross-referencing purposes
- Write to other tables for audit-trail purposes

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.

Managing triggers can involve:

- Obtaining current information about existing triggers
- Creating and dropping triggers
- Syb Ora Updating triggers
- Syb Renaming triggers
- DB2v2 Creating comments for triggers

Obtaining Current Information about Existing Triggers

You can obtain several kinds of current information about triggers on DB2 Version 2, ORACLE, and Sybase databases. Examples are:

- What triggers are stored in a given database
- What triggers are stored for a given table
- Syb General information about the trigger (such as when it was created)

To display the list of triggers on a given database

1 Select the database and click on the **Display** push button.

2 In the Display Types window, select **Trigger** and click on **OK**.

Syb Ora A filter window opens. If you want to enter a filter for the trigger display, type in an owner name or trigger name and click on **OK**. Otherwise, click on **OK** to display all triggers on the database.

3 The Trigger object type and any existing triggers appear on the object tree.

Syb For a Sybase trigger, you can display information such as when the trigger was created by selecting the trigger occurrence and clicking on the **Information** push button.

Creating and Deleting Triggers

Use the DataHub **Create** action to create a trigger.

To create a trigger

1 Select the Trigger object type and click on the **Create** push button. A window opens in which you can define the trigger.
2 Enter values for the parameters in the window. For explanations of the parameters, see the online help.
3 Click on **OK** when finished.

To delete a trigger

1 Select the trigger occurrence and click on **Drop**. A question box opens asking for confirmation.

2 Click on **Yes**.

By selecting trigger occurrences, you can perform the following actions on triggers by using DataHub:

- Syb Ora Change the parameters for a trigger by using the **Update** action.
- Syb Rename a trigger by using the **Rename** push button.
- DB2v2 Create a comment for a trigger by using the **Comment** push button.

For each action, select the trigger and then click on the appropriate push button.

Managing INGRES Rules

Ing Similar to a trigger, an INGRES rule invokes an INGRES database procedure when the associated table is modified by an insert, update, or delete action. When the table is modified, the rule is fired and the procedure is executed. INGRES rules can be used to implement integrity constraints.

Using DataHub, you can:

List the INGRES rules on an INGRES database
Create, update, and drop INGRES rules

To list the rules on an INGRES database

1 Double-click on the database occurrence in the object tree. The Display Types window opens.

2 Select **Rule** from the list and click on the **OK** push button. The Rule object type and any existing rules are displayed on the object tree.

Before creating an INGRES rule, the INGRES database procedure that the rule references must exist.

To create an INGRES rule

1 Select the Rule object type and click on the **Create** push button. The Define New Rule window opens.

2 Specify the parameters in the window. For descriptions of the parameters, see online help.

3 Click on the **OK** push button when finished.

To update an INGRES rule

Select the INGRES rule occurrence and click on the **Update** push button.

To delete an INGRES rule

Select the INGRES rule occurrence and click on the **Drop** push button.

Managing INGRES Database Events

Ing In general, an *event* is an occurrence of significance to a task. In an INGRES RDBMS, event “alarms” notify programs and applications when an event has occurred. These event alarms are called INGRES *database events*.

INGRES database events can be created and deleted, and they can be raised to signal that an event occurred. A process can receive an event if it has registered to be notified for that particular event. The raise and register operations are protected by authorizations.

Using DataHub, you can:

- List database events existing on an INGRES database
- Create and delete INGRES database events
- Authorize users to raise or register events

To list the database events on an INGRES database

1. Double-click on the database occurrence in the object tree. The Display Types window opens.
2. Select **Event** from the list and click on the **OK** push button. The Event object type and any existing database events on the database are displayed on the object tree.

To create an INGRES database event

1. Select the Event object type and click on the **Create** push button. The Create Event window opens.
2. Specify the parameters in the window. For descriptions of the parameters, see online help.
3. Click on the **OK** push button when finished.

To delete an INGRES database event

Select the INGRES database event occurrence and click on the **Drop** push button.

To grant, change, or remove authorizations for users to raise and register INGRES events

1 Select one or more event occurrences and click on the **Authorization** push button. A window opens in which you can work with authorizations for the database events.

2 To work with the authorizations for a particular database event, select the database event in the first area of the window and click on the appropriate push button.

For an explanation of the window areas and push buttons, see the online help.

3 Click on **OK** to return to the object tree.

Managing ORACLE Synonyms

Ora An ORACLE synonym is an alias for any table, view, snapshot, sequence, user, procedure, functions, or package. Typically, you use synonyms for reasons of security or convenience. For example, you can hide the real name and owner of an object by using a synonym; or refer to a view that has a complicated naming scheme by using a short synonym.

Managing ORACLE synonyms involves:

- Displaying the synonyms on a database
- Displaying the synonyms for a particular object
- Displaying the object a particular synonym refers to
- Creating and dropping synonyms

The following subsections discuss these tasks.

Displaying Synonyms

You can display all the synonyms on a database, or a selected subset. You can also display the synonyms for a particular object, or conversely, the object that is referenced by a particular synonym.

To display the synonyms on a database

Double-click on the database occurrence, select **Synonym** from the list, and click on **OK**. The Filter window opens. Either,

- Enter a filter and click on **OK**
- Click on **OK**

The Synonym object type and all of the synonyms that exist on the database are added to the object tree.

You can also display the synonyms for occurrences of these object types:

Function
Package
Procedure
Sequence
Snapshot
Synonym
Table
User
View

To display the synonyms for an object occurrence

Double-click on the object occurrence, select Synonym from the list, and click on **OK**.

The Synonym object type and any synonyms for that object are added to the object tree.

To display the object referred to by a particular synonym

Double-click on the synonym occurrence, select the object type you want from the list, and click on **OK**.

The object type and the particular object occurrence are added to the object tree.

Creating and Deleting Synonyms

Use the DataHub **Create** action to create a synonym.

To create a synonym

1. Display the Synonym object type and current list of synonyms, if any exist, for the object for which you want to create a synonym. See "Displaying Synonyms" on page 266 for the different ways of displaying synonyms.
2. Select the Synonym object type.
3. Click on the **Create** push button. The Create Synonym window opens.
4. Specify the parameters in the window. For explanations of the parameters, see the online help.
5. When finished, click on **OK**.

To delete a synonym:

1. Select the synonym you want to drop.
2. Click on the **Drop** push button. A question box opens to confirm your choice.

3 Click on the **Yes** push button.

Managing DB2 Version 2 Aliases

DB2v2 A DB2 Version 2 alias is an alternative name for an object or a database. Aliases can be given to databases, tables, views, and other aliases. Typically, you use aliases for reasons of security or convenience. For example, you can hide the real name and owner of an object by using a alias; or refer to a view that has a complicated naming scheme by using a short alias.

You can display, create, add comments to, and delete aliases.

To display the aliases that exist on a database

1 Double-click on the database, select **Alias** from the list, and click on **OK**. The Filter window opens.

2 If you want, enter a filter criteria.

3 Click on **OK** in the Filter window.

The Alias object type and any aliases on that database are displayed on the object tree.

To display the aliases for an object (table, view, or other alias)

1 Double-click on the object, select Defined Alias from the list, and click on **OK**. The Filter window opens.

2 If you want, enter a filter criteria.

3 Click on **OK** in the Filter window.

The Defined Alias object type and any aliases defined for that object are displayed on the object tree.

To create an alias for a table, view, or other alias

1 Select the Alias object type and click on the **Create** push button. The Create Alias window opens.

2 Specify the parameters and click on the **OK** push button. For explanations of the parameters, see the online help.

To add a comment to an alias

1 Select the alias occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

2 Enter a comment in the New Comment window and click on **OK** when finished.

To delete an alias

1 Select the alias and click on the **Drop** push button. A message box will open asking for confirmation.

2 Click on **Yes**. The alias is deleted.

Managing ORACLE Snapshots

Ora An ORACLE snapshot is a read-only copy of a table or data from a combination of more than one table. By replicating one or more tables from a remote database, a snapshot provides a local representation of the remote data. Snapshots can be periodically refreshed to reflect the up-to-date state of the table or tables. The remote database can have a corresponding snapshot log used for tracking the change activities for the master table of the snapshot at the local database.

Managing snapshots involves:

- Obtaining information about existing snapshots
- Creating, altering, and deleting snapshots

Managing snapshot logs involves:

- Obtaining information about existing snapshot logs
- Creating, altering, and deleting snapshot logs

Obtaining Information about Snapshots

You can list the snapshots on a database as well as display information about a particular snapshot.

To list the snapshots on a database

Double-click on the database, select **Snapshot** from the list, and click on **OK**. The Snapshot object type and any snapshot occurrences that exist on the database are added to the object tree.

To display information about a particular snapshot

1 Select the snapshot occurrence and click on **Show Snapshot**. The Snapshot Information window opens. For descriptions of the fields, see the online help.

2 When finished, click on **OK**.

Creating, Altering, and Deleting Snapshots

Tips: If you are creating a simple snapshot, it is more efficient to create the snapshot log before the snapshot. See "Creating, Altering, and Deleting Snapshot Logs" on page 271.

To create a snapshot

1 Select the Snapshot object type and click on **Add**. The Create Snapshot window opens.

2 Specify the parameters in the window. For information on the parameters, see online help. Use the **Show** push button to show the snapshot as it is currently defined by your query.

3 Click on **OK**.

To alter a snapshot

1 Select the snapshot occurrence and click on **Alter**. A window opens in which you can update parameters for the snapshot.

2 When finished, click on **OK**.

To delete a snapshot

1 Select the snapshot and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**. The snapshot is deleted. The snapshot log remains on the database.

Obtaining Information about Snapshot Logs

You can list the snapshot logs on a database as well as display information about a particular snapshot log.

To list the snapshot logs on a database

Double-click on the database, select **Snapshot Log** from the list, and click on **OK**. The Snapshot Log object type and any snapshot logs that exist on the database are added to the object tree.

To display information about a particular snapshot log

1 Select the snapshot log and click on **Show Snapshot Log**. The Snapshot Information window opens. For descriptions of the fields, see the online help.

2 Click on **OK** when finished.

Creating, Altering, and Deleting Snapshot Logs

A snapshot log can help decrease the amount of processing and time needed to refresh a simple snapshot. Snapshot logs cannot be used with complex snapshots.

A snapshot log is associated with one master table; if multiple simple snapshots are based on the same master table, they all use the same snapshot log.

You should create the snapshot log in the same database as the master table.

To create a snapshot log

1 Select the Snapshot object type and click on **Add**. The Create Snapshot Log window opens. Specify these parameters:

Master Table
Type the name of the master table

Table Owner
Type the name of the table owner

Storage... push button
Click on this button to set storage settings.

Usage... push button
Click on this button to set usage settings.

2 Click on **OK** when finished.

To alter a snapshot log

1 Select the snapshot log and click on **Alter**. A window opens in which you can update the storage and usage settings by clicking on the **Storage...** and **Usage...** push buttons.

2 When finished, click on **OK**.

To delete a snapshot log

1 Select the snapshot log and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**. The snapshot log is deleted.

Chapter 23. Managing DataHub Applications

Managing applications can be difficult even in a small client-server environment. Several applications can use a single database, and maintaining or making changes in this database has effects on all applications that use the database.

In a large client-server environment, application management becomes even more difficult. Applications might use more than one database, on different hosts in the network, and the dependencies between applications and databases can be large. In large organizations, where changes on databases are constantly occurring, you want to know how these changes affect your applications.

DataHub provides support for maintaining those tables and views that belong to applications. You can specify tables and views as belonging to DataHub applications, where the tables and views are located in different databases and different types of databases on different hosts.

In DataHub, *applications* are defined sets of related tables and views. To distinguish them from the type of applications that consist of programs, this book calls them *DataHub applications*.

DataHub applications make it possible to determine how a contemplated change in, or removal of, one table or view would affect tables and views related to it. With this information, you can decide whether to take the contemplated action, and whether to adjust the related tables and views if you do take it. DataHub applications also allow a convenient way to define additional views on any tables and views within a DataHub application.

A DataHub application can consist of tables or views that are stored in one or more databases.

The databases can be of one or more types residing on one or more hosts. The hosts can also be of one or more types.

Managing DataHub applications involves:

- Defining DataHub applications
- Displaying DataHub applications for a host occurrence
- Displaying the applications objects for a DataHub application
- Defining the relationships between the tables and views of a DataHub application
- Displaying DataHub applications in the form of Bachman diagrams

- Assessing the impact that a modification or deletion of one table or view in an application would have on the other tables and views and acting to prevent or adjust to the impact.

Defining DataHub Applications

You can define a DataHub application using tables or views from multiple databases, even databases that reside on different hosts.

To define a DataHub application

1 Display on the object tree those tables and views you want to use in the DataHub application. They can be from the same database, different databases on the same host, or different databases on different hosts. They can even be from different types of RDBMSs.

2 Select the tables and views you want to use. To select more than one object, hold down the **Ctrl** key while clicking on the objects.

3 Click on the **Define Appl** push button. The Define Application window appears.

4 Type in the name of the application and click on the **OK** push button. The Select Hosts For Application window appears.

5 Select the host or hosts you want as the application server or servers for this application and click on **OK**. A window pops up asking you whether you want to copy the foreign keys of tables in the DataHub application.

6 Click on the **Yes** push button if you want the relationships between tables to be made known to the DataHub application. Otherwise, click on the **No** push button.

The DataHub application is defined.

To display the DataHub applications on a host

1 Double-click on the host.

2 Select **Applications** from the list and click on **OK**.

The Applications object type and any DataHub applications on the host are added to the object tree.

The tables and views that make up a DataHub application are called *application objects*.

To display the application objects for a DataHub application occurrence

Double-click on the DataHub application occurrence. The Application Objects object type and the application objects for this DataHub application occurrence are displayed on the object tree.

Defining DataHub Application Relationships

You can define relationships between columns of DataHub application objects by using the **Create Relation** action. Columns must have the same data type. The relation is not a foreign key constraint; however, you can use it to represent such a constraint.

To define relationships between the objects in a DataHub application

1. Double-click on the application. The objects used by the application are displayed under the Application Objects object type in the object tree.
2. Select the objects for which you want to establish a relationship.
3. Click on the **Create Relation** push button. A window appears showing the selected objects and any previously defined relationships.

   Increase the window size to see all of the table columns. You can scroll horizontally and vertically using the scroll arrows.

In this window, you can add or delete relationships between the objects in the DataHub application.

To add DataHub application relationships

1. Click and hold down mouse button 1 on the first column you want to relate to another.
2. Drag the mouse arrow to the second column and release the mouse button.
3. Release the mouse button. When the mouse button is released, a line appears between the two columns.
4. Click on **OK**.

You can join two separate relationships to create one relationship over two columns.

To join two separate relations

1. Select the colored boxes in the lines for the relationships you want to join. The boxes will change color.

2 Click on the **Join** push button. The lines change to a single relationship line, with branches to the separate columns.

3 Click on the **OK** push button.

To delete a DataHub application relationship

1 Click on the colored square in the middle of the line between the two related columns.

2 Click on the **Delete** push button. The line disappears and the relationship is deleted.

3 Click on **OK**.

Displaying Bachman Diagrams

With DataHub, you can display your DataHub applications and the relationships between the application objects graphically in a Bachman diagram.

In a Bachman diagram, each of the application objects is represented by a rectangular box. The name of the object is given inside the box. If the object has a relationship with another object, there is a relationship line between the two.

You can display your DataHub applications and the relationships between the tables as Bachman diagrams.

To display DataHub applications as Bachman diagrams

1 Select the DataHub application in the object tree.

2 Click on the **Bachman** push button.

A window appears showing the Bachman diagram for the DataHub application.

Managing Application Objects

You can perform a number of DataHub actions on the application objects for a DataHub application. The number of actions available depends on what type of object (table or view) and what type of database the application object resides on.

To see the DataHub actions you can perform on an application objects object occurrence, select the application object occurrence. The push buttons for the available actions are shown in the Action window.

For information on the available actions, see

- "Administering Authorizations" on page 184 for the **Authorization** action
- Chapter 16, "Managing Tables" on page 211 if the application object is a table
- Chapter 17, "Managing Views" on page 221 if the application object is a view

To drop an application object

1 Select the application objects object occurrence and click on the **Drop** push button. A question box opens asking for confirmation.

2 Click on the **Yes** push button to confirm drop.

You can define an application on application objects; select the application objects and click on the **Define Appl** push button. See "Defining DataHub Applications" on page 273 for more information.

Updating and Deleting DataHub Applications

Use the **Update** action to change the name of a DataHub application.

To change the name of a DataHub application

1 Select the application occurrence and click on the **Update** push button. The Update Application window opens.

2 Enter a name and click on the **OK** push button.

Use the **Drop** action to drop a DataHub application. You can drop the application from all of the hosts it is defined on, or the host from which you displayed the application.

To delete a DataHub application

1 Select the application occurrence and click on the **Drop** push button. A question box opens asking if you want to delete the application from all hosts or just the current host occurrence.

2 Click on the selection you want.

Assessing the Impact of Deleting DataHub Application Objects

When a user attempts to drop a table or view on which a DataHub application is defined, DataHub displays the names of the applications that will get affected by the **Drop** action. You can then display the applications and application objects that are currently defined to determine the effect of dropping the table or view on the existing applications.

In addition, in a distributed database application environment, the DataHub application might need to access tables and views residing on various hosts on the network. Defining entity relationships for these distributed database application environments can be fulfilled through the use of DataHub applications because you can use tables and views across various RDBMSs and across different hosts.

Using DataHub's **Display** action, you can easily identify the DataHub applications defined on that host. Then, displaying the objects used in the DataHub applications, you can see which databases are involved in those DataHub applications.

Chapter 24. Stored Procedures

Ing Syb Ora With DataHub, you can manage the stored procedures in your database.

A *stored procedure* is a block of procedural constructs and embedded SQL statements that is stored in a database and can be called by name. A stored procedure can have both input and output arguments.

Stored procedures can help enforce data security. For example, you can authorize users to use a procedure to update a table, but not authorize them to access the table itself. The users can execute the procedure, but cannot arbitrarily manipulate the data in any other way.

Managing stored procedures involves:

- Ing Syb Ora Listing the existing procedures on a database
- Ing Syb Ora Creating and deleting procedures
- Syb Ora Updating procedures
- Ing Directly execute procedures
- Syb Rename procedures
- Ing Ora Syb Authorizing use of procedures

For information on authorizing use of procedures, see "Granting Authorizations on Specific Database Object Occurrences" on page 187.

To display the procedures that exist on a database, double-click on the database, select **Procedure** from the list, and click on the **OK** push button. The Procedure object type and any existing procedures are displayed on the object tree.

To create a stored procedure

> **But first:** Before you can create stored procedures in an ORACLE environment, you must run certain SQL scripts. To find out which scripts these are, see *Oracle Server Administrator's Guide.*

1. Select the Procedure object type and click on the **Create** push button. A window opens in which you can define the procedure.
2. Specify a name and owner for the procedure, and specify the procedure block. For explanations of the parameters in the window, see the online help.

Important: In a Sybase procedure, the first character of each parameter must be an "at sign" (@).

3 When finished, click on the **OK** push button.

To delete a stored procedure

Select the procedure occurrence and click on the **Drop** push button.

To update a stored procedure

Ora Syb Select the procedure occurrence and click on the **Update** push button. Alter the parameters you wish to change.

Caution: For Sybase procedures, object authorizations are lost after an update to a procedure.

Syb You can rename a Sybase procedure by selecting the procedure occurrence and clicking on the **Rename** push button.

Caution: If you rename Sybase stored procedures, the name of the procedure changes, but the body of the procedure does not. If you later update the procedure, you will see the old procedure name in the SQL block.

When you rename objects in Sybase, any references to the object by other objects will be invalidated if they still reference the old object name.

Ing You can directly execute an INGRES procedure by selecting the procedure object occurrence and clicking on the **Execute** action button.

Chapter 25. Functions

Ora DB2v2 With DataHub, you can manage DB2 Version 2 and ORACLE functions.

A *function* is a program or piece of a program that can be called. For DB2 Version 2, functions are written using a data manipulation language (DML), have zero or more input values called arguments, and return a single value. For ORACLE, a function is a PL/SQL block that contains procedural constructs and embedded SQL statements that can have input arguments only.

Managing functions includes:

- Listing existing functions on a database
- Creating and deleting functions
- DB2v2 Managing comments for functions
- Ora Updating functions definitions

To list the existing functions on a database

Double-click on the database, select **Function** from the list, and click on **OK**. The Function object type and any existing function occurrences are added to the object tree.

To create a function

> **But first:** Before you can create functions in an ORACLE environment, you must run certain SQL scripts. To find out which scripts these are, see *Oracle Server Administrator's Guide*.

1 Select the Function object type and click on **Create**. A window opens in which you can define the function.

2 Specify the parameters in the window. For an explanation of the parameters, see the online help.

3 When finished, click on **OK**.

To delete a function

1 Select the function occurrence and click on **Drop**. A question box opens for confirmation.

2 Click on **Yes**.

DB2v2 You can create, change, and replace comments for a DB2 Version 2 function by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the function occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

  If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Ora You can update an ORACLE function definition by selecting the function occurrence and clicking on the **Update** push button.

Chapter 26. Packages

DB2 Ora With DataHub, you can manage your database packages.

DB2 For DB2, a *package* is an object that includes all the information needed to process specific SQL statements from a single source file.

Ora For ORACLE, a package is a method of encapsulating and storing related procedures, functions, variables, and other package constructs as a unit in the database.

Managing packages involves:

- Listing the existing packages on a database
- Ora Creating packages
- Dropping packages
- DB2v2 Managing comments for packages
- Ora Updating packages
- DB2v2 Rebinding packages

To list the existing packages on a database

Double-click on the database, select **Package** from the list, and click on **OK**. The Package object type and any existing package occurrences are displayed on the object tree.

Ora

To create a package

But first: Before you can create packages in an ORACLE environment, you must run certain SQL scripts. To find out which scripts these are, see *Oracle Server Administrator's Guide.*

1 Select the Package object type and click on **Create**. A window opens in which you define the package. Specify the parameters in the window. For an explanation of the parameters, see the online help.

2 When finished, click on **OK**.

To delete a package

1 Select the package and click on **Drop**. A question box opens for confirmation.

2 Click on **OK**.

Ora You can update the definition of an ORACLE package by selecting the package occurrence and clicking on the **Update** push button.

DB2v2 You can create, change, and replace comments for a DB2 Version 2 package by using the DataHub **Comment** action.

To create, change, or replace a comment

1 Select the package occurrence and click on the **Comment** push button. The Create Comment on Object window opens.

  If there is an existing comment, it is displayed in both the Old Comment and New Comment fields. If there is no existing comment, both fields are blank.

2 To add a new comment, enter it in the New Comment field.

3 To change an existing comment, edit the comment in the New Comment field.

4 To replace an existing comment, delete the comment in the New Comment field and enter a new one.

5 When finished, click on the **OK** push button.

Binding is the process that creates the DB2 Version 2 package the DB2 database manager needs in order to access the database when the application is executed. Use the DataHub **Rebind** action to rebind a DB2 Version 2 package.

Monitoring

Chapter 27. Monitoring Your Databases and Operating Systems

Overview of Monitoring

DataHub′s rule-based monitoring system helps you move towards continuous availability and optimal usage of your database and system resources. Intelligent agents on each managed system monitor these resources around the clock. The ability to set thresholds which represent exception conditions allows management by exception. Monitoring results from multiple resources on a given system can be correlated and evaluated by rules, which initiate corrective actions if the conditions specified in the rules are met. Trend analysis allows DataHub to take actions not only to correct problems but to prevent problems from occurring in the first place.

Local Rule Evaluation

Correlation and analysis of monitoring results from the operating system and any databases on a specific host is done locally at the managed host. Corrective actions, too, can be taken locally without the need to communicate with the DataHub control point GUI. This allows the managed host to monitor itself even if the control point is not available. Actions taken and their results are logged locally at each host; you can view this log from the control point. Optionally, monitoring results can be sent to the control point for storage in the DataHub control database for graphical display and historical analysis.

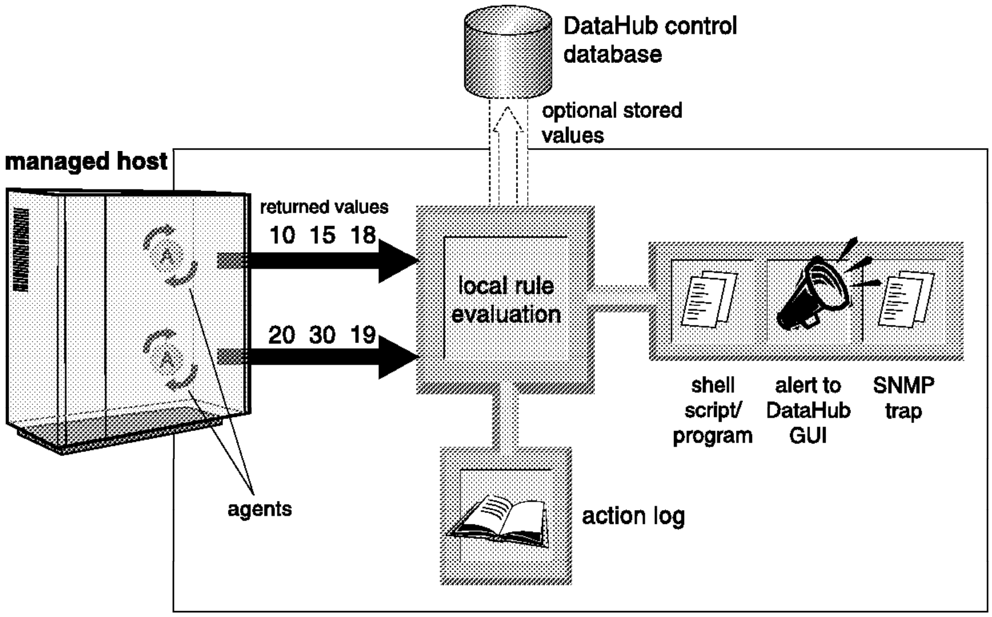


Figure 14. Schematic of Local Rule Evaluation

Global Rule Evaluation

DataHub can analyze monitoring results from resources on multiple hosts for cross-system problems. In this case, the intelligent agents send the appropriate monitoring results to the control point for correlation and analysis by global rules. Global rules then initiate preventative and corrective actions at the host or hosts causing the problem.

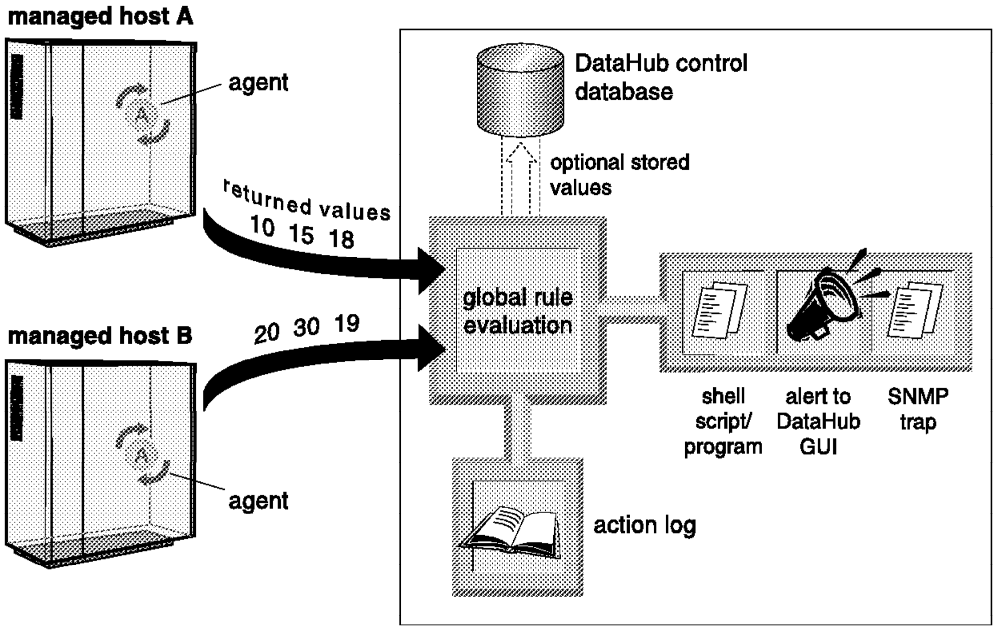


Figure 15. Schematic of Global Rule Evaluation

At the core of DataHub′s monitoring system are intelligent agents, called *watchdogs*. *Watchdog checks* contain the definition of what to monitor, when to monitor, and thresholds which specify exception conditions to report. *Watchdog rules* contain statements to evaluate the monitoring results from one or more watchdog checks the actions to be taken. Possible actions are: sending an alert to a DataHub administrator, sending an SNMP trap to any SNMP management application, paging or sending a message, invoking a UNIX or SQL script, a command or a program.

DataHub′s rule-based system can also be used to schedule recurring administrative tasks.

DataHub comes with hundreds of predefined watchdog checks for the operating systems and databases that DataHub manages. You can select any of these predefined watchdog checks and add your own filters and rules. Of course, you can modify any watchdog check and rule at any time to fit your changing environment. You can even extend the set of watchdog checks with your own in-house scripts, commands or programs.

Some of these checks are standard across most organizations and configurations, and can be very useful to you right out of the box. To get you started quickly and easily, DataHub has defined filter values and rules for this standard subset, and you can activate this subset with a single mouse click. The filter values represent typical exception conditions and the rules encapsulate DBA expertise to offer advice to the DataHub administrator for dealing with the reported situations.

Using DataHub′s rule-based monitoring system, its predefined checks, and out-of-the-box monitoring set, you can automate your operations based on your own business policies.

Watchdogs

There are two kinds of watchdogs: an *OS watchdog* executes OS checks; a *database watchdog* executes database checks. For each control point in your DataHub system, there is one OS watchdog and one or more database watchdogs per managed host.

For example, suppose you have a simple network: one control point from which you manage one host and on that host you manage Oracle and DB2 Version 2. In this configuration, there would be one OS watchdog, a database watchdog for Oracle, and a database watchdog for DB2 Version 2.

Watchdog Checks

You specify the conditions you want DataHub′s watchdogs to monitor by defining a watchdog check. What is a watchdog check? Briefly, a watchdog check (called *check* for short) consists of the following:

- An executable instruction.
- A time and/or time interval at which to execute the instruction.
- A filter value to filter out results that are not of interest to you. Filtering of results is optional.
- A request to store the check results in the DataHub control database. Storing results in the control database is optional.

In addition, each execution of a watchdog check produces a single value. This value can be an integer, floating point number, or string.

These characteristics of watchdog checks and examples of checks are described in more detail in the following sections.

You define the watchdog check via the graphical user interface. The watchdog check definition is stored in the control database, and routed to the watchdog on the appropriate managed host immediately after you have created the check.

DataHub comes with a large number of *predefined checks* that are valuable for monitoring database and operating system conditions. You can use predefined checks “as is” or modify them to suit your own requirements. You can also create new watchdog checks to monitor other database and operating system conditions of interest to you.

Examples of watchdog checks.

- Check #1 instructs a watchdog to (1) produce, at two-hour intervals, beginning immediately, running forever, a value denoting the number of users logged on to a given host, and (2) check whether or not the value is greater than 10.
- Check #2 instructs a watchdog to (1) produce, every day, for a month, beginning tomorrow at 7 am, a value denoting the number of tables in a given database, (2) determine whether or not the value is greater than 50, and (3) if the value exceeds 50, store the value in the control database.

Using these examples, you can see that:

- Each check has a unique identifying number (in our examples, 1 and 2). DataHub assigns this check number automatically.
- A check contains an instruction to be executed. This instruction produces a single value that indicates the status of a condition at a database or operating system.

  In an OS check, this instruction can be an OS command, a program, or a shell script invoking OS commands or programs. For readability, from here on, we refer to the instruction for an OS check as an OS command.

  In a database check, this instruction can be an SQL statement, an OS command, a program or a shell script containing any of these. For readability, from here on, we refer to the instruction for a database check as an SQL statement.

  For an illustration of using shell scripts in your custom checks, see “Working With Watchdog Checks” on page 297.

  When you define a check, you can specify the instruction from scratch, or you can use one that DataHub has predefined. Check #1 includes a predefined OS command:

  who | column 10 | sort -u | wc -l

  Check #2 includes a predefined SQL statement:

  select count(*) from sysibm.systables where type = ′T′

  If you specify an instruction from scratch, be sure to code it so that it returns only one value each time it is executed. Also verify that the data type of the returned value matches the data type you select when you specify the watchdog check.

- A check is run at a specific time or time interval

  You can specify:

  – Regular intervals, starting at a particular time and either stopping at a particular time or continuing indefinitely.

    For example, check #1 is to be executed at regular intervals (every two hours). Execution begins when a watchdog receives the check and continues indefinitely. Check #2 is also to be executed at regular intervals (every day). Execution starts the day after the check was created, at 7 am, and continues for a month.

  – One or more particular times.

    For example, you could have a check executed only once. Also, if you want a check executed at irregular intervals, you would need to indicate the time of each execution.

  – Regular intervals *and* one or more particular times.

    For example, you may want to execute a check on every Thursday in 1996, and on the days before a certain application is run (let's say a quarterly application which runs on April 1, July 2, October 3 and December 16).

- A check can contain a *filter* to filter out check values that are not of interest and only report exception conditions.

  Specifying a filter is optional. The filter in check #1 determines whether or not the value is greater than 10. The filter in check #2 determines whether or not the value is greater than 50.

- A check produces a single value when it has completed execution. This value can be an integer, floating point number, or string.

- A check can contain the request to store the filtered values in the control database. Storing check values in the control database is optional.

So far we have only discussed watchdog checks on individual managed objects. DataHub provides a grouping function that allows you to manage collections of objects as a unit. You can create a watchdog check on a whole group of objects; the single watchdog check you created is executed for every member of the group. See Chapter 5, "Managing Multiple Objects" on page 50 for more information on groups.

Watchdog Rules

The evaluation of monitoring results and the firing of actions occur using watchdog rules. Watchdog rules (or rules for short) are ″if-then″ statements; they specify that if monitoring results meet certain criteria, then DataHub should perform specified actions. Possible actions are: sending an alert to a DataHub user; sending an SNMP trap to any SNMP management application; paging or sending a message; or invoking a UNIX or SQL script, a command, or a program.

For example, consider check #1 (page 290). Every two hours, a watchdog executes this check and produces a value denoting the number of users logged on to a given host. The watchdog check definition states that we are interested in this value only if it exceeds 10 (in other words, if more than 10 users are logged on). You can use a watchdog rule to specify what you want done if the value exceeds 10. Such a rule could say that if two or more consecutive values exceed 12, then DataHub should notify the DBA via e-mail.

DataHub lets you define more than one rule for a given watchdog check. For example, you could define two additional rules on check #1. Your second rule could cause an alert to be sent to the DataHub administrator if more than 20 users are logged on (the value of check #1 exceeds 20). Your third rule could cause the administrator to be paged if more than 25 users are logged on (the value of check #1 exceeds 25).

In addition to defining multiple rules on a single check, you can define one or more rules on multiple checks. Suppose you wanted to check file system free space and system load in terms of number of users logged on and CPU utilization. If these three parameters indicate that the system load on the managed system is too high, you want to alert the administrator so that she can take action to ease the load on the system. You would need three checks: one to check file system free space, one to check number of users logged on, and one to check CPU usage. You could then define a rule specifying that if all of the following conditions are true:

- User file system free space is less than 10%
- The number of users logged on to the host is more than 15
- CPU usage is over 70%

then DataHub should notify the administrator with an appropriate message and a recommendation for correction.

The actions that DataHub can perform fall into one of three general categories: sending messages to a user, launching programs, generating SNMP traps.

Sending messages to users
The message DataHub sends is called an *alert*. An alert might simply report a condition that the user, for one reason or another, needs to keep tabs on. Alternatively, an alert can both report a condition of concern and indicate how to adjust it. This is illustrated in the preceding example, where the DBA is notified with a message and a recommendation for correction.

Launching programs:
DataHub can launch a program you wrote or a program provided by a system vendor, such as a program to page somebody or send an e-mail message to somebody. Launching a shell script falls into this action category. Here is an example:***:*** A rule is defined over a check that denotes whether the abc.staff table in the db2db database would benefit from a reorganization. The definition of the rule specifies that if the value is greater than zero, then DataHub should execute a program or shell script which performs a reorganization for the table and then updates the statistics using Run Statistics, followed by a rebind of all packages which use the table.

Generating SNMP traps:
Generating SNMP traps is a way of forwarding alerts to SNMP-based network management consoles (such as NetView, HP OpenView, Solstice SunNet Manager), systems management consoles such as Tivoli, or any management application which can receive SNMP traps.***:*** Just as you can create watchdog checks for groups of managed objects, you can create watchdog rules for these checks. For more information, see Chapter 5, “Managing Multiple Objects” on page 50.

Out-of-the-Box Monitoring

DataHub provides *out-of-the-box monitoring* for hosts, Sybase servers, and databases. You can activate out-of-the-box monitoring with a simple mouse click.

When you choose out-of-the-box monitoring for a host, Sybase server, or database, DataHub adds watchdog checks (with preset filter values) and associated watchdog rules to that host, Sybase server, or database. The action associated with out-of-the-box monitoring is to alert a DataHub user when the conditions defined in the rules are met.

You can add out-of-the-box monitoring to a host, Sybase server, DB2, ORACLE, INGRES database by selecting the Add Standard Monitoring check box when you are:

- Adding a new managed host, Sybase server, DB2, ORACLE, or INGRES database with the **Add** action
- Changing the definition of a currently managed host, Sybase server, DB2, ORACLE, or INGRES database with the **Definition** action

Syb You can add out-of-the-box monitoring to Sybase databases by using the **Add Std Montr** action button.

For information on the **Add** and **Definition** actions, see Chapter 6, "Managing Hosts with DataHub" on page 66, Chapter 7, "Managing Sybase Servers" on page 74, or the chapters on managing the different database types.

DB2 The standard monitoring for DB2 databases requires DB2 Monitor Switches to be active for valid information to be collected. If the standard monitoring is activated and DB2 Monitor Switches are not active, some of the standard checks will not collect valid data. The DB2 Monitor Switches which are required by the standard monitoring are the Buffer Pool and Sort Monitor Switches. These switches are activated or deactivated at the DB2 instance (see "Setting Monitor Switches" on page 98).

To add out-of-the-box monitoring for a host, Sybase server, or non-Sybase database

1 Select the Add Standard Monitoring check box.

A question box opens asking for confirmation.

2 Click on **Yes**. A window opens asking you for the name of the DataHub user who will receive the alerts.

3 Click on **List** to select from a list of users and then click on **OK**.

Syb Because Sybase databases are retrieved dynamically from the Sybase server itself and are not defined directly to DataHub DataHub provides a special action button to add out-of-the-box monitoring to Sybase databases.

To add out-of-the-box monitoring for a Sybase database

1 Select the Sybase database and select the **Add Std Montr** action button.

A question box opens asking for confirmation.

2 Click on **Yes**. A window opens asking you for the name of the DataHub user who will receive the alerts.

3 Click on **List** to select from a list of users and then click on **OK**.

Adding out-of-the-box monitoring might take a few minutes. When the process is complete, a message is displayed that shows the number of watchdog checks and rules that have been added.

To see the watchdog checks that DataHub has added, display the watchdog checks on the object occurrence (host, Sybase server, or database) (see "Displaying Watchdog Checks and Rules" on page 296). To see the filter values used for these watchdog checks and how frequently the check is run, select the watchdog check object occurrence and click on **Alter**.

To see the watchdog rules that DataHub has added, display the watchdog rules for each check (see "Displaying Watchdog Checks and Rules" on page 296). To change the user that receives the alerts from these standard monitors, select the watchdog rule object occurrence and click on **Update**.

Please note:

1. If you add out-of-the-box monitoring more than once, you will get multiple copies of the same watchdog checks and rules.
2. DB2 Ing Ora If you want to add out-of-the-box monitoring for a database that you have created with DataHub, use the **Definition** action.
3. Syb If you want to add out-of-the-box monitoring for a database that you have created with DataHub, use the **Add Std Montr** action.
4. To remove out-of-the-box monitoring, you must drop each individual watchdog rule and watchdog check.

Starting and Stopping Watchdogs

You can start or stop the watchdogs for a particular host by using the **Monitoring System** choice of the **Administer** menu. This lets you temporarily disable all watchdog checks and rules on that host through a single action, without having to drop each watchdog check or update the start or stop times of each watchdog check.

Attention:

If you stop the monitoring system of your control point:

- Global rules will not be evaluated
- Check values will not be stored in the control database
- New checks and rules or changes to existing checks and rules will not be sent to hosts whose monitoring system is still running

We strongly advise that you **not** stop the monitoring system for the control point.

To stop the monitoring system for a particular host

1 Select the **Administer** menu, select **Monitoring System**, and select **Stop**. The Operating System Login window opens.

2 Enter the name of the managed host where you want to stop the watchdogs.

3 Enter a user name and password and click on **OK**. This user name must have root authority on the host.

A window opens that shows the status as the monitoring system stops. Click on **OK** to close this window.

To start the monitoring system for a particular host

1 Select the host object occurrence or an object occurrence displayed under that host.

2 Select the **Administer** menu, select **Monitoring System**, and select **Start**. The Operating System Login window opens.

3 Enter a user name and password and click on **OK**. This user name must have root authority on the host.

A window opens that shows the status as the monitoring system starts. Click on **OK** to close this window.

Displaying Watchdog Checks and Rules

You can display the watchdog checks that exist for a particular object occurrence. Watchdog checks that are created against an occurrence of a database object (such as a table) are displayed at the database on which the object resides.

For example, if you have watchdog checks against the tables on the DEVTEST database, you display the existing watchdog checks from the

DEVTEST occurrence on the object tree, not from the table occurrences under the database.

Watchdog checks that are created against an occurrence of a Sybase server object (such as a dump device) are displayed at the Sybase server on which the object resides.

To display existing watchdog checks

1. Double-click on the host, Sybase server, or database occurrence.
2. Select **Watchdog Check**, and click on **OK**. The Watchdog Check object type and any watchdog checks already defined against the selected object displayed on the object tree.

   For a database occurrence, this includes checks defined against objects within the database.

To display the watchdog rules associated with a particular watchdog check

1. Double-click on the watchdog check. The Watchdog Rule object type and the watchdog rules associated with the watchdog check are displayed on the object tree.

To display watchdog rules

1. Double-click on the host, Sybase server, or database occurrence.
2. Select Watchdog Rule from the Display Types window. The Watchdog Rule object type and any existing watchdog rule occurrences for the selected occurrence or one of its objects are added to the object tree.

To display the watchdog checks associated with a particular watchdog rule

Double-click on the watchdog rule. The Watchdog Check object type and the watchdog checks referenced by that rule are added to the object tree.

Working With Watchdog Checks

Working with watchdog checks involves:

- Creating checks
- Showing values for checks
- Deleting values for checks
- Altering checks
- Dropping checks

Creating Watchdog Checks

You create watchdog checks against object occurrences (such as hosts, databases, and table spaces).

To create a watchdog check for an object occurrence

1 Display the Watchdog Check object type under the object occurrence for which you want to create a watchdog check. To do this, double-click on the object occurrence, select Watchdog Check from the Display Types window, and click on **OK**.

2 Select the Watchdog Check object type and click on the **Create** push button. A window opens in which you can define the watchdog check. For explanations of all of the parameters, see online help. Some parameters are described briefly here.

Check
Name of the check. If you want to select a predefined check, select the **List** push button and choose one from the selection list. If you are creating your own check, enter a name for the check.

Command
The command or script to be run. The command or script must return a single value.

For host objects, this is an operating system command or shell script.

For databases and database objects, this can be an SQL statement or script or an OS command. It can be an SQL query only if the query returns a single value.

Return Type
Data type of the value that is returned. The value can be an integer, floating point number, or string.

Time
The scheduled time at which the watchdog check is to be run. See "Specifying Watchdog Check Scheduling" on page 301 for how to specify the scheduled time.

Filter Type and Filter Value
You can set a threshold to filter out values that are not of interest. Only values that satisfy the filter specification are used in watchdog rule evaluations. For example, if you specify a check to report the number of users on this host, you can set a filter to report back only if:

Number of Users > 20

Store check values

Specify whether or not you want to store the returned values in the control database.

If you select **No**, the values are used by the monitoring system as needed for watchdog rule evaluation and then discarded. By selecting **No**, you can minimize network traffic. If you select **Yes**, the values are stored in the control database for graphical display. You can also use your favorite query tool against this data.

3 The following sections describe how to specify your watchdog check. When you have specified the parameters, click on the **OK** push button of this window to create your check.

FYI: If you specify a filter, then only these filtered values are used in watchdog rules that reference this watchdog check. Avoid contradictions that might occur when you define a watchdog rule on a filtered value.

For example, if you request a filter that says only return the watchdog check value if the number of users is greater than 10, and your watchdog rule says to alert if the value is greater than 5, then you will only get alerts if the watchdog returned value is greater than 10, because the smaller values are already filtered out.

To use one of the predefined watchdog checks

1 Click on the **List** push button next to the Check parameter. The Select One window opens.

2 Select the predefined check you want. Click on the **OK** push button.

When you select a predefined check, the name of the check appears in the **Check** field and the operating system command, script name, or SQL command appears in the **Command** field.

Please note: If you use one of the predefined checks directly ("as is") without editing the command, you should use the **Test** push button to test your watchdog check before creating it. In some instances, the predefined checks might need to be tailored for your particular configuration. See "To test your watchdog check" on page 303 for more information on testing your watchdog check.

You can edit the command and the command name that is in this field to create your own watchdog check based on a predefined check. In this way, you can customize a predefined check to fit your business needs.

To edit the command, change focus to the Command field by clicking the mouse in the field.

To define your own watchdog check for a host

1 Type a name in the **Check** field.

2 Type the operating system command or script name in the **Command** field. This must be a command or script that can be executed by the operating system watchdog. In addition, the user ID that was defined as the Watchdog User name during the installation and configuration of DataHub must be able to execute this command or script. For most system performance information, general privileges are be sufficient.

Be sure to code the check so that it will return a single value. The value can be an integer, floating point number, or string.

3 In the **Return Type** parameter, select the appropriate data type for the returned value.

To define your own watchdog check for a database object

1 Type a name in the **Check** field.

2 Type the SQL statement in the **Command** field. Remember that the SQL statement must return a single value.

You can edit the command that is in this field to create your own watchdog check based on a predefined check. In this way, you can customize a predefined check to fit your business needs.

To edit the command, change focus to the Command field by clicking the mouse in the field. Be sure to code the check so that it will return a single value. The value can be an integer, floating point number, or string.

3 In the **Return Type** parameter, select the appropriate data type for the returned value.

Lets look at an example of a custom watchdog check, called Check #3. Check #3 instructs a DB2 watchdog to run the DB2 REORGCHK on the abc.staff table in the db2db database; the db2db database is in the db2inst instance. The check returns a value indicating whether this table would benefit from reorganization. Because this check requires multiple statements to be issued (OS statements as well as database commands), we want to create a shell script and run the shell script from the watchdog check.

Given our shell script:

- Named chkreorg
- Located in directory /mydir/mychecks/db2v2
- Expecting three parameters:
  1. db2 instance name
  2. db2 database name
  3. schema.table name

we enter in the **Command** field for the check definition this command:

```
/mydir/mychecks/db2v2/chkreorg db2inst db2db abc.staff
```

Our chkreorg shell script would contain the DB2 command to run the REORGCHK utility, redirect the utility's report to a file, and analyze the file so that it can produce a value that represents whether the table would benefit from a reorganization. The shell script returns the value for the watchdog check via stdout.

In your shell scripts, you should ensure that only the value you want to return for your check is sent to stdout.

Specifying Watchdog Check Scheduling

You specify the times at which you want the watchdog check to be run in the Time field of the Define Watchdog Check window. You can add, change, and drop time specifications in this field.

To add time specifications for the watchdog check

1 Click on the **Add** push button in the **Time** field. The Add Time for Check window appears.

In this window, you can specify:

- Start time for the watchdog check.
- How often the watchdog is to be run.
- End time for this watchdog check. Default is forever.

2 To specify a start time:

To start now, make sure the Start Now check box is selected.

To start at a later time,

- **a** Deselect the Start Now check box. The Start Time field is active.
- **b** Click on the **List** push button to open the Enter Time window.
- **c** In the Enter Time window, you can specify the year, month, day, hour, and minute at which the check will be run.
- **d** After making your specifications, click on the **OK** push button.

3 To specify how often to return values:

To repeat only once, make sure the Repeat Once check box is selected.

To repeat at intervals,

- **a** Deselect the Repeat Once check box. The Repeat Interval field is active.
- **b** Click on the **List** push button to open the Enter Interval window.
- **c** In the Enter Interval window, you can specify how many years, days, hours, and minutes are in the interval.
- **d** After making your specifications, click on the **OK** push button.

4 To specify an end time for this watchdog check:

To repeat forever, make sure the Repeat Forever check box is selected.

To specify an end time,

- **a** Deselect the Repeat Forever check box. The End Time field is active.
- **b** Click on the **List** push button to open the Enter Time window.
- **c** In the Enter Time window, you can specify the year, day, hour, and minute for the end time.
- **d** After making your specifications, click on the **OK** push button.

5 When finished in the Add Time for Check window, click on the **OK** push button. The time or time interval you specified appears in the Time field in the Define Watchdog Check window.

You can add more time specifications by selecting **Add** in the Time field and following the previous procedure for each time or time interval you want to add. You can specify one repeating time interval and multiple specific times associated with a single watchdog check.

To change a time in the Time field, select the row and click on the **Change** push button. To drop a time in the Time field, select the row and click on the **Drop** push button.

Specifying Filter Type and Filter Values

If you specify a filter for the watchdog check values, only values that satisfy the filter are used in watchdog rule evaluations.

To specify a filter

1. Click on the **Filter Type** list button. Click on the filter you want.
2. Change focus to the Filter Value field and enter a value.

Numeric filter values of less than 1 must have a leading zero (0). For example, if you want a filter so that returned values are all greater than 0.1, then enter 0.1 in the **Filter Value** field, not .1.

String filter values must be enclosed in double quotes (″) and the values cannot be more than 20 characters long.

To store check values

If you want the returned check values to be stored in the control database, select the **Yes** toggle button. If you specified a filter, only the filtered values are stored.

The default is **No**; if **No** is selected, the values are used by the monitoring system as needed for watchdog rule evaluation and then discarded.

If you select **No**, network traffic is minimized.

To test your watchdog check

1. After you have specified your parameters, click on the **Test** push button. The Operating System Login window opens.
2. Enter a user name and password and click on **OK**. The Values window opens and displays the results from a test run of the check.
3. Click on the **Close** push button to return to Create Watchdog Check window.

When you are done, select the **OK** in the Define Watchdog Check window. The watchdog check you defined appears in the object tree under the Watchdog Check object type.

Working With Stored Watchdog Check Values

If you selected to store check values when you created a watchdog check, the values are kept in the control database unless you explicitly delete them. You can use the DataHub **Show Values** action to display a line chart of the 40 most recent values for a particular watchdog check.

Please note: If a filter has been defined for the check, only values that meet the filter's criteria are stored. If the times or time intervals in the check have been updated, you might see values returned at the original times or time intervals, as well as values returned at the updated times or time intervals.

To show values for a check

1 Select the watchdog check.

2 Click on the **Show Values** push button. A window opens and displays the 40 most recent values on a line graph.

For an explanation of this window, see the online help.

You can delete the values for an individual watchdog check or delete the values for all of the watchdog checks in your DataHub installation.

You can choose to delete all values stored prior to a to a specified time.

To delete values for a watchdog check

1 Select the watchdog check.

2 Click on the **Delete Values** push button. The Delete Values window appears.

Enter the date and time from which data will be kept. All check results before the entered date are deleted.

3 Click on the **List** push button to specify the date in the Enter Time window.

4 When finished, click on the **OK** push button.

To delete values for all watchdog checks in your DataHub installation

1 Click on the **Administer** menu, click on **Check Values**, and click on **Delete Entries**. The Delete Values window appears.

Enter the date and time from which data will be kept. All watchdog check results before the entered date are deleted.

2 Click on the **List** push button to specify the date in the Enter Time window.

3 When finished, click on the **OK** push button.

The status message line shows whether deletion is successful.

Altering and Dropping Watchdog Checks

You can alter and drop watchdog checks using the **Alter** and **Drop** actions.

To alter a watchdog check

1. Select the watchdog check you want to alter.
2. Click on the **Alter** push button. The Define Check window opens.
3. Change those parameters you want to change. For a description of the parameters, see "Creating Watchdog Checks" on page 298 or online help.
4. Select the **OK** push button.

To drop a watchdog check

> **Please note:** If you try to delete a watchdog check that is used by one or more watchdog rules, a message appears saying that the check is being used. You should first delete the associated rules before attempting to delete the watchdog check.

1. Select the watchdog check you want to drop.
2. Click on the **Drop** push button.
3. Click on **Yes** in the Question Box.

The check is removed from the system.

When you drop a watchdog check that was storing the check values in the control database, all of the stored values are deleted.

Working With Watchdog Rules

Working with watchdog rules involves:

- Creating watchdog rules
- Updating watchdog rules
- Dropping watchdog rules

Creating Watchdog Rules

Before creating a watchdog rule, you must define all the watchdog checks that you want to use in the rule.

To bring up the Create Watchdog Rule window

1 Display the watchdog check or checks for which you want to add a rule. See "Displaying Watchdog Checks and Rules" on page 296.

2 Select all the watchdog checks this rule uses. You can select more than one check by holding the **Ctrl** key while clicking on the check name. You can also select checks that are on different hosts or databases.

3 Click on the **Create Rule** push button. The Create Watchdog Rule window appears.

The Create Watchdog Rule window has the following fields:

Watchdog checks
The checks you selected to be used in this rule are listed here. Each check is preceded by its identifier, the number which identifies the check.

Rule Text
Specifies the if part of the if-then rule for this watchdog rule.

Action Type
This parameter specifies the system action to take if the if-then rule is satisfied.

Action	***Result***
Alert	DataHub users notified
Launch	Command or script initiated
SNMP-Trap	SNMP trap generated

FYI: Have alerts sent to DataHub users who you think are likely to be logged on to the DataHub control point when the alerts are issued.

Users cannot receive alerts when logged off from the DataHub control point; however, the alerts are queued and will be displayed when the users start their DataHub sessions.

Action
Describes the action to take in order to solve the problem.

See the online help for more information on these parameters.

To enter the rule text

In the **Rule Text** field, enter the condition for this watchdog rule. A condition is a Boolean expression that follows the syntax described in "Syntax for Watchdog Rule Text" on page 307.

You use the identifiers for the watchdog checks to specify the values returned by them in the condition. You can specify values that were returned by previous generations of a watchdog check.

For example, if Check #4 is specified as Number of Free Space (Blocks) in a database (monitor the number of free space blocks) and you want to take action if there is a reduction of free space (free space is 10% less than last time you checked), then enter in the **Rule Text** field:

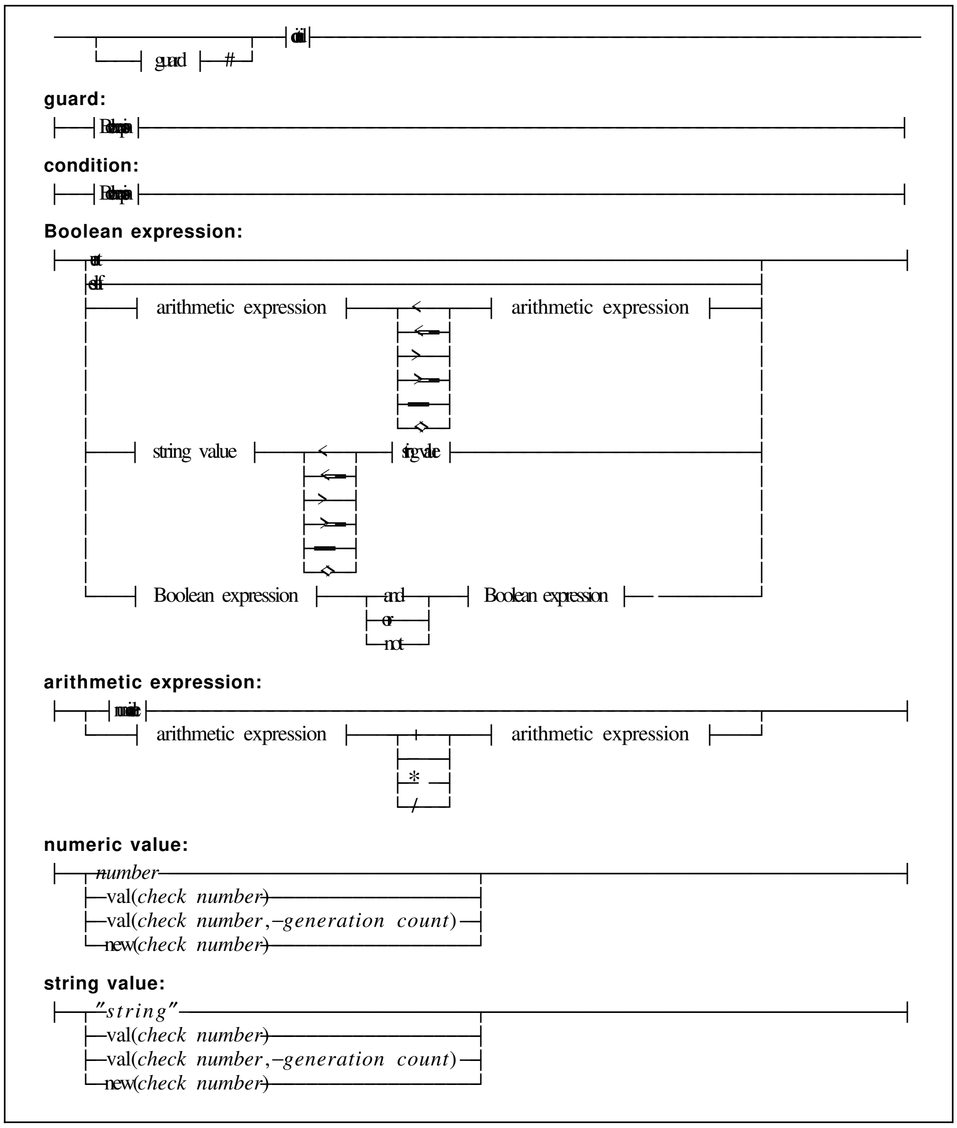
```
val(30) <= 0.9 * val(30,-1)
```

The value, val(30,−1) is the value that was returned just before the most recent value. The watchdog rule takes action when the free space in blocks is less than or equal to 90% of the prior value.

Syntax for Watchdog Rule Text

The Rule Text consists of two components: an optional *guard*, which determines whether the condition will be evaluated, and the condition, which represents the evaluation of check results against specific criteria.

The syntax for a watchdog rule text is described in the following syntax diagram.



When the guard evaluates to true, the condition will be evaluated. The guard can be used to ensure that new values for all checks involved in the rules evaluation have been seen prior to the condition in the rule being evaluated by using **new(***check number***)** in the guard.

val(*check number***)** The keyword **val** indicates the most recent value for the check associated with *check number*.

new(*check number***)** This is for use in the guard only. The keyword **new** indicates that a new value for the check associated with *check number* has been seen since the last time the guard evaluated to true. This does not guarantee that the value has changed, only that the value is being included in the rules evaluation.

check number A valid check number.

generation count A negative integer which represents a prior value for this check. For example, If you want to use the third prior value for a check, then *generation count* = −3, and the numeric value would be **val(***check number***, −3)**.

Standard operator precedence applies. Comparisons of strings is based on alphabetical order.

Watchdog rule examples

Here are some examples of watchdog rule text to illustrate the watchdog rule syntax:

(val(5)-val(10)) > (val(5,-1)-val(10,-1))
: The resulting value is bigger than the last time checks 5 and 10 were returned

val(4)<>″YES″
: The value is not equal to ′YES′

val(6)<=10 and val(6)>=5
: The value is between 5 and 10

val(2)=5
: The value is 5

val(2)/val(3) >0.20
: The ratio of check 2 to check 3 is greater than 20%

val(2)>val(3)*1.25
: The value of check 2 is more than 25% greater than the value of check 3

val(2) > ((val(2)+val(2,-1)+val(2,-2)+val(2,-3)+val(2,-4)) /5)
: The value of check 2 is greater than the average value returned over the previous five generations

val(2)=val(3)
: The value of check 2 is equal to the value of check 3

val(2)<val(2,-1)
: The value of check 2 is less than value of check 2 the last time check 2 was returned

val(4)=″Backup Complete″
: The value is ′Backup Complete′

val(8)< 1 and val(8,-1)>= 1
: The value of check 8 is less than one AND the value of check 8 the last time it was returned and >=1 (could indicate the status of the object being checked has changed status)

val(4)<>val(4,-1)
: The value of check 4 is not the same value that it was the last time it was returned

(val(8)-val(9)<2) and (val(12)-val(13)<2)
: Both calculations yield a result which is less than 2

new(12) and new(14) # val(12) / val(14) > 0.20
: When new values for checks 12 and 14 are both present calculate the ratio of check 12 to check 14 if the ratio is greater than 20%. trigger the action defined for the rule. The ratio will not be calculated again, until new values have been returned for both check 12 and check 14.

Specifying a Watchdog Rule Action

You can

- Alert DataHub users
- Launch commands or shell scripts
- Generate SNMP traps

Considerations for specifying alerts and launches

Every 5 minutes, the control point GUI looks to see if there are any alerts to be shown. If you schedule the watchdog check to return a value every 1 minute and the rule specifies an alert is to be sent, the control point GUI won′t show the alert to the user until the 5 minute interval is reached.

Each time an alert is sent, DataHub enters a row in the table, "alert_table", in the control database. If a large number of alerts are being sent, you will need to periodically clear the table. To accomplish this, use the SQL DELETE command on the table.

Launches are run immediately; therefore, if you need to see alert conditions closer to the time the condition occurs, consider using e-mail, beeper, or SNMP traps instead of alerts.

You can include the watchdog check values that triggered the rule within the definition of launch and SNMP trap actions, but not in DataHub alerts. This allows you to use the check value in taking corrective action, whether it is automated or manual. For more information, see the sections "To specify a launch" on page 311 and "To specify generation of an SNMP trap" on page 312.

To specify an alert

1. Click on the **Alert** radio button.
2. Click on the **List** push button next to the **Action** field. The Define alert window appears.
3. Select an alert class by clicking on the **Alert Class** list and clicking on a color.

   Color is only used to group classes of alerts and is not related to the color of the actual alert. The alert panel always shows a flashing red button.
4. Type in a description of the situation in the **Description** field. For example, in our scenario of the number of users greater than 12, enter:

   The number of users on this host is greater than 12.
5. In the **Action** field, enter the action to be taken by the alerted user. For our example,

   Tell 3 users to log off the system.
6. In the **Send alert to** field, either enter the names of users or select from a list. To display the list, click on the **List** push button.
7. Click on **OK** when finished. In the Create Rule window, the **Action** field shows the alert description.

To specify a launch

1. Click on the **Launch** radio button.
2. Click on the **List** push button next to the **Action** field. The Specify Operating System Command window appears.
3. Specify an operating system command or UNIX shell script that is performed when the watchdog rule is satisfied.

   Within your command, you can use expressions which substitute one or more watchdog check values that triggered the rule within your command. The syntax for including a watchdog check value within the command which is to be launched is:

   :> expression <:

The syntax for the included expression is the same as that for the watchdog rule conditions; see "Syntax for Watchdog Rule Text" on page 307.

For example, if the launch action is to send email to a particular user when the value of Check #30 Number of Free Space (Blocks) in a database is 10% less than the last time the Free Space was checked, then the command could be specified as:

```
echo Free space in database decreased at least 10%
from one interval to the next—from
a value of :> val(30,–1):<
to a value of :> val(30) :< | mail db2adm@diamond
```

4 Enter the host on which the command or script is to be launched in the Host field, or click on the **List** push button to select from a list of hosts.

5 Enter a user name in the User Name field, or click on the **List** push button to select from a list of users.

6 Enter the password for the user in the User Name field.

7 Click on **OK** when finished. In the Create Rule window, the **Action** field shows the operating system command or UNIX shell script.

To specify generation of an SNMP trap

1 Click on the **SNMP-Trap** radio button.

2 Click on the **List** push button next to the **Action** field. The Specify SNMP-Trap window opens. See the online help for information on the parameters.

Within the trap message, you can use expressions which substitute one or more watchdog check values that triggered the rule. The syntax for including a watchdog check value within the trap message is:

```
:> expression <:
```

The syntax for the included expression is the same as that for the watchdog rule specifications; see "Syntax for Watchdog Rule Text" on page 307.

For example, if the action is to send an SNMP trap when the value of Check #30 Number of Free Space (Blocks) in a database is 10% less than the last time the Free Space was checked, then the trap message could be specified as

```
Free space in database decreased at least 10% from a value of
:> val(30,-1) :< to a value of :> val(30) :<
```

3 Click on **OK** when finished.

To complete defining the watchdog rule

1 Click on **OK** in the Add Rule window.

2 Click on the Watchdog Rule object type and then click on the **Refresh** push button. The new watchdog rule is added to the object tree.

Updating and Dropping Watchdog Rules

You can update and drop watchdog rules by using the **Update** and **Drop** actions.

To update a watchdog rule

1 Select the watchdog rule you want to update.

2 Click on the **Update** push button. The Add Rule window appears. For descriptions of the parameters, see "Creating Watchdog Rules" on page 305 or online help.

3 Change the parameters you want to update. and click on the **OK** push button when finished.

To drop a watchdog rule

1 Select the watchdog rule.

2 Click on the **Drop** push button.

3 Click on **Yes** in the question box. The rule is removed from the object tree.

Viewing and Deleting the Action Log

When a launch or SNMP trap is initiated on a host, DataHub records the information in that host's action log. When the action completes, DataHub records the return code in the action log. Only launches and SNMP trap actions are logged; alerts are not.

You must have root privilege on the host to delete the host action log.

To view the action log for a host

1 Select the host object occurrence.

2 Click on the **Administer** menu, click on **Action Log**, and select **View Host Action Log**. The Operating System Login window opens.

3 Enter a user name and password combination.

A window opens and displays the contents of the action log file. Lines are added to the file in the order that the actions (or return codes) occur. See the online help for more information.

To delete the action log for a host

1. Select the host object occurrence.
2. Click on the **Administer** menu, click on **Action Log**, and select **Delete Host Action Log**. The Operating System Login window opens.
3. Enter a user name and password combination that has root privilege.

   The contents of the action log are deleted.

Advanced Topics

Chapter 28. User-Defined Objects and Actions

This chapter is intended to help the customer use, customize, and modify functions in DataHub.

In DataHub, you can add your own objects that can be displayed on the object tree. You can also add your own action buttons in the Action window to invoke tools by simply clicking on the action button.

To add user-defined objects or actions, you must add information into certain DataHub control tables in the control database.

Caution: Throughout this book, notes have advised not to directly manipulate the control database control tables. To add user-defined objects or actions, most of the tables you insert into are defined expressly for adding user-defined objects and actions.

However, there are some tables that are used by DataHub for all objects and actions and which you will have to manipulate. These tables are identified in this section.

Before manipulating these tables, you might want to back-up your control database, so that you can recover in case of corruption of the tables. Note that if you need to recover the control database, you must do so outside of DataHub, as DataHub will not be operational.

Defining Objects

To define your own objects to be managed by DataHub and displayed on the object tree, you need to insert data into the following DataHub control tables in the control database.

dba_class Contains definitions of all objects defined to DataHub

dba_udisp_func Contains relationships between object types.

dba_udisp_comm Contains the operating system or SQL commands that are executed when a user double-clicks on an object

dba_udisp_filter Contains information used in the filter window (if any)

dba_udisp_priv Contains information on DBA group privileges for the object type

dba_class_status Contains the colors that can be used for an occurrence of an object type, depending on the status of the occurrence.

The following sections describe:

- Details of the DataHub control tables listed above
- Example of a user-defined object

DataHub Control Tables for User-Defined Objects

To define DataHub objects, you must have the authorization to manipulate the DataHub control tables. When DataHub is installed, its control tables are created in the control database. These tables are initially owned only by the userid that was provided to the install process. However, after installation, user names that can start DataHub sessions should have at least select authorizations for the control tables (see “Creating Control Database Users” on page 36); however, to define objects, you need update and insert authorizations for these tables.

dba_class Table

Caution: The dba_class table is used to define all of the object types used by DataHub. Incorrect manipulation of this table might corrupt the table and DataHub itself.

Before manipulating this table, it is recommended that you back-up the table.

The dba_class table holds the definitions of all of the object types defined to DataHub. The following list describes the columns of this table:

class_nr Identifies the user-defined object. This is the primary key for this table. You should use a number in the range of 1000 to 5000 inclusive.

class_name A character string of maximum length 32 that is the object type name shown in the Display Types window when you double-click on an object occurrence from which you can display the user-defined object.

dba_udisp_func Table

This table holds the definitions of the relationships between object types. Every row in this table must have a unique number; this number is used in other tables. The fields of this table have the following descriptions:

ud_func_num Identifier of the function that implements the **Display** action from a “from_class” object occurrence to the related type to be displayed (the “to_class“ object type and object occurrences). This column is the primary key for this table.

from_class Identifies the source object type for the **Display** action. This column is a foreign key to the table “dba_class”.

from_type The qualifier that defines the host type or RDBMS type for which the relationship between “from_class” and “to_class” holds.

to_class Identifies the target object type for the **Display** action. This column is a foreign key to the table “dba_class”.

cmd_type Integer value indicating the type of command that will be issued to retrieve all the occurrences of the object type identified in the “to_class” for a given “from_class” object occurrence.

The command itself is defined in the table “dba_udisp_comm”. The allowed values are:

- 1 - indicates an operating system command is to be issued
- 2 - indicates an SQL command is to be issued

output_sep Used only for operating system commands. It defines the field separator for the operating system commands because they do not return columns as the SQL commands do.

status A single character that describes whether or not the command specified in table dba_udisp_comm returns a status as the first field. The values can be either “Y” or “N”.

- if “status” is “Y”, the first column returned by the command is the value for the status of the object occurrence, the second column is the label that will be displayed in the object tree next to the object occurrence, and the remaining columns are stored as internal object values.

- if “status” is “N”, the first column returned by the command is the label that will be displayed in the object tree next to the object occurrence, and the remaining columns are stored as internal object values.

Internal object values can be referenced by actions acting on that object occurrence or a related object type branching out from that object occurrence. See the description for **&1 through &9** (page 319) for more information.

dba_udisp_comm Table

This table holds the command to be executed when the user displays the user-defined object type and occurrences from the parent object occurrence.

ud_func_num An integer that identifies the command for the **Display** action. This column is a foreign key to the table "dba_udisp_func".

comm_part Command sequence number, starting from 1.

Columns "ud_func_num" and "comm_part" together make up the primary key for this table.

command A character string (maximum length 255 characters) for the command that will be issued. If the command is longer than 255 characters, you should break the command into multiple parts, one row in this table for each part, and use the "comm_part" field to indicate the order of the command parts.

The command is executed on the target host. It is either executed by a rexec command (for operating system commands) or by the database driver (for SQL commands).

For writing the command, the following rules are observed:

- If the command is an operating system command, the maximum length of the command is 2048 bytes.
- If the command is an SQL command, the maximum length is 20K bytes.
- The output of the command should have the following order of columns:
  - Optional column indicating status of the object
  - Column used as the label that is displayed next to the object in the object tree
  - Columns used for DataHub actions on that object occurrence or for displaying other object types from this object occurrence. These internal columns are stored as data within the object browser.
- The command itself can contain variables that will be substituted prior to execution of the command. These variables are:

  #1 through #9
  Values entered by the user in a filter window

&1 through &9
: Variables representing the internal object values associated with the object occurrence being selected for the **Display** action. These variables, if referenced in the command, will be substituted by the corresponding values before DataHub executes the command used in implementing the **Display** action.

&dbpasswd
: The password used to connect to the current database under which the object is selected

&dbname
: The current database name under which the object is selected

&instance
: The current DB2 instance name under which the object is selected

&server
: The current Sybase server name under which the object is selected

&hostname
: The current host name

&dbuser
: The user name for the connection to the current database or Sybase server

&scriptdir
: Script directory for the selected object DataHub uses &scriptdir to locate scripts to be executed for certain tasks (such as ORACLE shutdown).

The script directory is the host directory ("host_dir" as defined in the table dba_host) concatenated with /drivers/scripts/ and the corresponding database type. For example, in the case of an ORACLE shutdown, if the host directory for a particular managed host is /u/dhu/AIX, then &scriptdir has the value /u/dhu/AIX/drivers/scripts/ora7.

&display
: Current value of the DISPLAY environment variable

&dbdir
Database directory of the database that has the selected object (for example, ORACLE_HOME for ORACLE, SYBASE for Sybase, or INGRES for INGRES).

dba_udisp_filter Table

This table holds specifications for a filter window where the user can enter values to filter the user-defined object occurrences that will be displayed on the object tree.

ud_func_num
Identifies the **Display** action for a user-defined object. This column is a foreign key to the table "dba_udisp_func".

filter_num
Integer ranging from 1 through 9 inclusive. It is the filter substitution, as specified in the command string in the dba_udisp_comm table (the #1 through #9 values).

Columns "ud_func_num" and "dba_udisp_func" together make up the primary key for this table. If there is more than one filter for the **Display** action, the values of "filter_num" have to be in sequence starting from 1.

filter_name
Character string of maximum length 255 that will be displayed in the filter window.

dba_udisp_priv Table

In DataHub, there are display privileges associated with all the DataHub objects (see "Granting and Revoking Privileges to Display Objects" on page 43). In this table, you specify the groups that should have the display privileges for this user-defined object type. You should at least grant the group DATAHUB the display privilege on the new user-defined object types.

group_num Integer that identifies the group that will be granted the display privilege for the user-defined object type. This column is a foreign key to the table "dba_group".

ud_func_num Identifies the **Display** action for a object type on which the display privilege is granted. This column is a foreign key to the table "dba_udisp_func".

Columns "group_num" and "ud_func_num" together make up the primary key for this table.

The "group_num" for the DATAHUB group is 0. For every unique "ud_func_num" in table "dba_udisp_func" you should have one entry in "dba_udisp_priv" granting the DATAHUB group the privilege.

dba_class_status Table

In this table, you can define the object occurrences reaction to various status values by changing the color of the object occurrences' icons. The columns in this table are:

class_nr
Integer that identifies the object type. This column is a foreign key to the table "dba_class".

dbtype
Character string of maximum length 20 identifying the database type for the status. Valid values are:

- "db6000" for DB2 Version 1
- "db2_2" for DB2 Version 2
- "ing64" for INGRES
- "ora7" for ORACLE
- "syb" for Sybase
- "NULL" indicates the status value is interpreted in the same way regardless of RDBMS type

value
Character string of maximum length 30 identifying the status value that affects the color change.

Columns "class_nr", "dbtype", and "value" together make up the primary key for this table.

color
Character string of maximum length 20 indicating the color the object occurrence icon should change to depending on the status value obtained.

Example of a User-Defined Object

The following example shows how to add user-defined objects to DataHub.

Define the Object Types

Given a simple project registration program, you define the following user tables, owned by “user1” :

```
create table project (
    projectcode     varchar(20),
    projectname     varchar(255)
    )

create table component (
    projectcode       varchar(20),
    componentcode     varchar(20),
    componentname     varchar(255),
    startdate         date,
    enddate           date,
    finished          integer
    )
```

Make the Object Types Known to DataHub

Insert the following rows into table “dba_class”

```
insert into dba_class values ( 1000, 'Project')
insert into dba_class values ( 1001, 'Component')
```

Set the Relationships Between Object Types

You want the object type ‘Project’ to be displayed from a database object occurrence for the database type ‘DB2 Version 1’ and the object type Component to be displayed from a project object occurrence. (Note the Database object type has “class_nr”= 6.)

Insert the following rows into table “dba_udisp_func”

```
insert into dba_udisp_func values
                          ( 1, 6, 'db6000', 1000, 2, null, 'N');

insert into dba_udisp_func values
                          ( 2, 1000, 'db6000', 1001, 2, null, 'Y');
```

Note that you display the Component object type (1001) from the Project object type (1000).

Specify the Command for Displaying on the Object Tree

Next, specify the command to be executed in the table "dba_udisp_comm". As indicated in the above table, we will be executing SQL commands.

Insert the following rows into table "dba_udisp_comm"

```
insert into dba_udisp_comm values (1, 1, '
    select projectname, projectcode              2
    from  user1.project
    where projectname like ''#1%'' ')            1

insert into dba_udisp_comm values (2, 1, '
    select finished, componentname, projectcode, componentcode    3
    from  user1.component
    where projectcode = ''&1'' ')                4
```

- 1 The #1 will be substituted by the value entered by the user in the filter window.
- 2 This SQL command implements the **Display** action for displaying the Project object type and any object occurrences from a DB2 Version 1 database object occurrence. The value of the "status" column for this **Display** action is "N". In this case, the first column (projectname) returned by the SQL command will be the label that is displayed as a project object occurrence in the object tree (see the description of **status** on page318 for more information on the "status" column).

  The second column (projectcode) returned is the first internal object value, &1, that will be referenced by the second SQL command (in 4).
- 3 The second SQL statement implements the **Display** action for displaying the Component object type and any object occurrences from a project object occurrence. The value of the "status" column for this **Display** action is "Y". In this case, the first column (finished) returned by the SQL command is the status value for a component object occurrence.

  The second column (componentname) returned will be the label that is displayed as a component object occurrence in the object tree.

  The rest of the columns (projectcode, componentcode) are the internal object values for this command.
- 4 The internal object value, &1, is the value of the first internal object (projectcode) returned by the first SQL command (2). The value returned will be substituted into the SQL command before the command is executed.

Specify a Filter Window

DataHub will bring up a filter window and the user can specify special substitutions that will be used in the commands. See the 'like' clause in the first SQL statement above.

You also specify the entry field label to be used in the filter window.

Insert the following row into "dba_udisp_filter"

```
insert into dba_udisp_filter values ( 1, 1, 'Project Name')
```

Give Display Privilege to DATAHUB Group

Update the table "dba_udisp_priv" to indicate the group DATAHUB can display the user-defined object types.

```
insert into dba_udisp_priv values ( 0, 1)
insert into dba_udisp_priv values ( 0, 2)
```

Add Colors for Status of Component Object Occurrence

In the table "dba_class_status" you can specify the color of the object occurrence icon based upon the status of the object occurrence.

Insert the following rows into "dba_class_status"

```
insert into dba_class_status values ( 1001, 'db6000', '0', 'red')
insert into dba_class_status values ( 1001, 'db6000', '1', 'green')
```

User-Defined Actions

In DataHub, you can define actions that can act on DataHub objects or user-defined objects. Push buttons for the user-defined actions applicable to an object selected on the object tree will appear in the Action window, along with the DataHub action push buttons (if any) for that object.

To define your own actions that you can use within DataHub, you need to insert data into the following DataHub control tables in the control database.

dba_function Contains the actions defined to DataHub

dba_privs Contains the relationships between actions, object types, and privileges

dba_func_attr_desc Contains additional information on the action itself

dba_func_comm Contains characteristics of the command to be executed when the action push button is selected

dba_ud_func_desc Contains the actual command to be executed when the action push button is selected

dba_func_parameters Contains descriptions of the parameters for the action for which the user will be prompted

The following sections describe:

- Details of the DataHub control tables listed above
- Example of a user-defined action

DataHub Control Tables for User-Defined Actions

To define actions, you must have the authorization to manipulate the DataHub control tables. When DataHub is installed, its control tables and views are created in the control database. These tables and views are initially owned only by the userid that was provided to the install process. However, after installation, user names that can start DataHub sessions should have at least select authorizations for the control tables (see "Creating Control Database Users" on page 36); however, to define actions, you need update and insert authorizations for these tables.

dba_function Table

Caution: The dba_function table is used to define all of the actions used by DataHub. Incorrect manipulation of this table might corrupt the table and DataHub itself.

Before manipulating this table, it is recommended that you back-up the table.

The dba_function table contains the definitions of all of the actions defined to DataHub. The following list describes the columns of this table:

function A unique integer that represents the action. You should use a unique number in the range of 20,000 to 50,000 inclusive.

name A string of a maximum length 40 that is the name of the action that will be displayed in the action push button (in the DataHub Action window).

color A string of a maximum length 20 that is the color of the action push button. Acceptable values for this field are the Motif colors that are supported on your system.

type A single character indicating whether the action is a DataHub provided action or a user-defined action. The only possible values are:

- **"i"** For a DataHub provided action ("internal")
- **"x"** For a user-defined action ("external")

Specify "x" for your user-defined actions.

dba_privs Table

This table contains the relationships between the action, object types, and privileges.

> **Caution:** The dba_privs table is used to define the access privileges each DBA group has for built-in DataHub actions and user-defined actions. Incorrect manipulation of this table might corrupt the table and DataHub itself.
>
> Before manipulating this table, it is recommended that you back-up the table.

The table columns have the following descriptions:

function_nr Identifies the action; this is a foreign key to the table "dba_function".

class_nr Identifies the object type upon which this action acts; this is a foreign key to the table "dba_class" (see "dba_class Table" on page 317).

host_type A string of a maximum length 10 that identifies the operating system type associated with the action (if applicable). Valid values are:

- **NULL** No applicable operating system type
- **aix** AIX operating system
- **hpux** HP-UX operating system
- **solaris** Solaris operating system

This column is a foreign key to the table "dba_host_type".

db_type A string of a maximum length 10 that identifies the database type associated with the action (if applicable). Valid values are:

- **NULL** No applicable database type
- **db6000** DB2 Version 1 database
- **db2_2** DB2 Version 2 database
- **ing64** INGRES database
- **ora7** ORACLE database
- **syb** Sybase database

This column is a foreign key to the table "dba_database_type".

group_nr An integer that identifies the group that will have the privilege to execute the action. This columns is a foreign key to the table "dba_group".

The "group_nr" for the DATAHUB group is 0. At a minimum, you should grant the execute privilege to the DATAHUB group.

dba_func_attr_desc Table

In this table, you specify whether the action can be performed on an object type, or on object occurrences of the object type, and whether the action can be performed on multiple object occurrences simultaneously.

> **Caution:** The dba_func_attr_desc table is used to define the properties of all of the actions used by DataHub. Incorrect manipulation of this table might corrupt the table and DataHub itself.
>
> Before manipulating this table, it is recommended that you back-up the table.

The table columns are:

function_nr Identifies the action; this column is a foreign key to the table "dba_function".

class_or_instance A character field that indicates whether the action can be performed on an object type or on object occurrences of the object type. Valid values are:

- **"class"** The action operates on the object type
- **"inst"** The action operates on object occurrences

one_or_multi A character field that indicates whether the action can be performed on a single object occurrence or on multiple object occurrences simultaneously. Valid values are:

- **"multi"** Can perform the action on multiple object occurrences
- **"one"** Can perform the action on a single object occurrence

dba_func_comm Table

This table provides information on the type of command that will be executed. The columns are:

func_nr Identifies the action; this is a foreign key to the table "dba_function".

command_nr An integer that represents the sequence number (starting at 1) identifying the command for the action.

It is possible that a given action has multiple commands associated with it.

Columns "func_nr" and "command_nr" together make up the primary key for this table.

command_type A single character indicating how the command should be issued. The valid values are:

- **"q"** Indicates an SQL statement
- **"o"** Indicates an operating system command.
- **"x"** Indicates an X-based application. This type of command will go off and start an X-based application while you continue to use DataHub.

dba_ud_func_desc Table

This table provides information on the commands that are executed for a given action. The columns are:

func_nr Identifies the action.

command_nr An integer that represents the command sequence number for the action.

The columns "func_nr" and "command_nr" together are a foreign key to the table "dba_func_comm".

segment_nr An integer identifying the sequence number of the command segment.

If the command is too long to be contained in a character string of length 255, “segment_nr” provides a way to break the command into separate segments.

Columns “func_nr”, “command_nr”, and “segment_nr” together make up the primary key for this table.

segment This is the command itself, or the command segment if more than one segment are need to represent the command. It has a maximum length of 255 characters. If the length of the command exceeds 255 characters, you can break it into segments.

The command itself can contain strings that will be substituted prior to execution of the command. These substitution strings are:

#1 through #9
Parameters entered by the user in a window. The parameters will be substituted into the command segment before the corresponding command is executed.

&1 through &9
Variables representing the internal object values associated with the object occurrences being selected for the action. These variables, if referenced in the command, will be substituted by the corresponding values before DataHub executes the command used in implementing the action.

&dbpasswd
The password used to connect to the current database under which the object is selected

&dbname
The current database name under which the object is selected

&instance
The current DB2 instance name under which the object is selected

&server
The current Sybase server name under which the object is selected

&hostname
The current host name

&dbuser

The user name for the connection to the current database or Sybase server

&scriptdir

Script directory for the selected object. DataHub uses &scriptdir to locate scripts to be executed for certain tasks (such as ORACLE shutdown).

The script directory is the host directory ("host_dir" as defined in the table dba_host) concatenated with /drivers/scripts/ and the corresponding database type. For example, in the case of an ORACLE shutdown, if the host directory for a particular managed host is /u/dhu/AIX, then &scriptdir has the value /u/dhu/AIX/drivers/scripts/ora7.

&display

Current value of the DISPLAY environment variable

&dbdir

Database directory of the database that has the selected object (for example, ORACLE_HOME for ORACLE or SYBASE for Sybase)

dba_func_param Table

This table describes the parameters for which the user will be prompted for values when performing the user-defined action.

The table columns are:

func_nr Identifies the action; this column is a foreign key to the table "dba_function".

param_nr An integer that represents the sequence number (starting at 1) of the parameters.

Columns "func_nr" and "param_nr" together make up the primary key for this table.

param_name A character of length 40 that will be displayed on the window generated by DataHub when the user is prompted to supply parameters.

Example of a User-Defined Action

In this example, we add the following actions for the user-defined objects defined in "Example of a User-Defined Object" on page 322. We want to add actions and corresponding push buttons so we can:

- Create a new project
- Delete a project
- Create a new component
- Close a component
- Reopen a component
- Delete a component
- Show components

In addition to these actions, we want to link the DataHub **Display** and **Refresh** actions to the Project object type, and the **Refresh** action to the Component object type.

Define the Actions

We insert the following rows into the "dba_function" table to define the name and number for the actions, and the colors of the corresponding action push buttons.

Insert the following rows into the "dba_function" table

```
insert into dba_function values ( 20000, 'Create Project',
        'Yellow', 'x')
insert into dba_function values ( 20001, 'Delete Project',
        'Red', 'x')
insert into dba_function values ( 20002, 'Create Component',
        'Yellow', 'x')
insert into dba_function values ( 20003, 'Close Component',
        'Turquoise', 'x')
insert into dba_function values ( 20004, 'Reopen Component',
        'Turquoise', 'x')
insert into dba_function values ( 20005, 'Delete Component',
        'Red', 'x')
insert into dba_function values ( 20006, 'Show Component',
        'Purple', 'x')
```

Link Actions to Object Types

Next step is linking the actions to the correct object types and granting access to the DATAHUB group. The "group_num" for the DATAHUB group is 0.

For this example, we define these actions for the DB2 Version 1 database type.

Insert the following rows into the "dba_privs" table

```
insert into dba_privs values ( 20000, 1000, null, 'db6000', 0)
insert into dba_privs values ( 20001, 1000, null, 'db6000', 0)
insert into dba_privs values ( 20002, 1001, null, 'db6000', 0)
insert into dba_privs values ( 20003, 1001, null, 'db6000', 0)
insert into dba_privs values ( 20004, 1001, null, 'db6000', 0)
insert into dba_privs values ( 20005, 1001, null, 'db6000', 0)
insert into dba_privs values ( 20006, 1001, null, 'db6000', 0)
```

The Create Project (20000) and Delete Project (20001) actions act on the user-defined object Project (1000).

We also want to link the DataHub **Display** and **Refresh** actions to the Project object type, and **Refresh** to the Component object type. The **Display** action has the "function_nr" 0; the **Refresh** action has the "function_nr" −100.

```
insert into dba_privs values ( 0, 1000, null, 'db6000', 0)
insert into dba_privs values ( -100, 1000, null, 'db6000', 0)
insert into dba_privs values ( -100, 1001, null, 'db6000', 0)
```

Specify Additional Action Information

We want the **Create Project** and **Create Component** actions to be performed on the object type, while the other actions to be performed on object occurrences. We also want the user-defined actions to be performed on one object occurrence at a time.

Insert the following rows into the "dba_func_attr" table

```
insert into dba_func_attr_desc values ( 20000, 'class', 'one')
insert into dba_func_attr_desc values ( 20001, 'inst', 'multi')
insert into dba_func_attr_desc values ( 20002, 'class', 'one')
insert into dba_func_attr_desc values ( 20003, 'inst', 'multi')
insert into dba_func_attr_desc values ( 20004, 'inst', 'multi')
insert into dba_func_attr_desc values ( 20005, 'inst', 'multi')
insert into dba_func_attr_desc values ( 20006, 'class', 'one')
```

In table "dba_function", the "function" column is 20000 for the **Create Project** action, 20002 for the **Create Component** action, and 20006 for the **Show Component** action. In the example above, the corresponding rows specify that these actions are performed on the object types, while the other actions are performed on the object occurrences.

Indicate the Type of Command

We specify the type of command in the "dba_func_comm" table. For our example, all of the commands are SQL statements.

Insert the following rows into the "dba_func_comm" table

```
insert into dba_func_comm values ( 20000, 1, 'q')
insert into dba_func_comm values ( 20001, 1, 'q')
insert into dba_func_comm values ( 20001, 2, 'q')
insert into dba_func_comm values ( 20002, 1, 'q')
insert into dba_func_comm values ( 20003, 1, 'q')
insert into dba_func_comm values ( 20004, 1, 'q')
insert into dba_func_comm values ( 20005, 1, 'q')
insert into dba_func_comm values ( 20006, 1, 'q')
```

Notice that for the **Delete Project** action (func_nr 20001) there are two commands associated with the action (see explanation in Specify the Commands for the Actions).

Specify the Commands for the Actions

We specify the actual commands to be executed for the actions. See "dba_ud_func_desc Table" on page 329 for a description of parameters you can use in these commands.

Insert the following rows into the "dba_func_desc" table

```
insert into dba_ud_func_desc values ( 20000, 1, 1,
    'insert into project values ( ''#1'', ''#2'')')     1
insert into dba_ud_func_desc values ( 20001, 1, 1,              2
    'delete component
        where projectcode = ''&1''')                            3
insert into dba_ud_func_desc values ( 20001, 2, 1,              2
    'delete project
        where projectcode = ''&1''')                            3
insert into dba_ud_func_desc values ( 20002, 1, 1,
    'insert into component
        values ( ''&1'', ''#1'', ''#2'',           4 ,  5
                 ''#3'', '' #4'', #5)')     5
insert into dba_ud_func_desc values ( 20003, 1, 1,
    'update component set finished = 1
        where componentcode =  ''&2''                   6
        and   projectcode = ''&1''')
insert into dba_ud_func_desc values ( 20004, 1, 1,
    'update component set finished = 0
        where componentcode =  ''&2''                   6
        and   projectcode = ''&1''')
insert into dba_ud_func_desc values ( 20005, 1, 1,
    'delete component
        where componentcode =  ''&2''                   6
        and   projectcode = ''&1''')
insert into dba_ud_func_desc values ( 20006, 1, 1,
    'select * from user1.component
        where projectcode = ''&1''')
```

- 1 The #1 and #2 will be substituted by the parameter values entered by the user in the entry window (see “Specify Entry Field Labels” on page 335, numbers 1 and 2 for “func_nr” 20000).
- 2 The **Delete Project** action (“func_nr” 20001) has two commands associated with it. The first command deletes all the components within the project; the second command deletes the project itself.
- 3 The &1 in these two commands are the internal object values that were stored for the project object occurrence.
- 4 This command acts on the Component object type; the &1 in this command comes from displaying the project occurrence from which the Component object type is displayed. Here, it is the projectcode for the component being created in this Create Component action.
- 5 The #1 through #5 in this command will be substituted by the parameter values entered by the user in the entry window (see “Specify Entry Field Labels,” numbers 1 through 5 for “func_nr” 20002).
- 6 The &1 and &2 in these three commands come from the component occurrence; they are stored as internal object values (“projectcode” and “componentcode” respectively) when the component object occurrence is displayed.

Specify Entry Field Labels

We specify the labels for window parameters where the user enters data to be substituted in the commands.

Insert the following rows into “dba_func_param” table

```
insert into dba_func_param values ( 20000, 1, 'Project Code')
insert into dba_func_param values ( 20000, 2, 'Project Name')
insert into dba_func_param values ( 20002, 1, 'Component Code')
insert into dba_func_param values ( 20002, 2, 'Component Name')
insert into dba_func_param values ( 20002, 3, 'Start date')
insert into dba_func_param values ( 20002, 4, 'End date')
insert into dba_func_param values ( 20002, 5, 'Finished')
```

Chapter 29. Interacting with SNMP Network Management Applications

DataHub provides the option of forwarding a message to SNMP-based Network Management Systems (such as NetView) using traps. With this capability, the network administrator can be notified of conditions detected by DataHub watchdogs, or when a particular offline job fails. Examples of using this function are:

- A watchdog check detects an abnormal situation in a database. The watchdog triggers a trap which is forwarded to the network management system.
- A scheduled job to backup production tables returns a condition code indicating the lack of disk space. The offline job examines this return code and forwards a trap to the network management system.

Generating SNMP traps in DataHub is very simple. It involves invoking a DataHub module called dhtrap and passing to it the parameters necessary to formulate the contents of the trap. The dhtrap module can be invoked as a result of satisfying a watchdog rule that has as its action one of the following items:

- Generating an SNMP-trap (see “To specify generation of an SNMP trap” on page 312 for more details)
- Launching a dhtrap call with the appropriate parameters

The format of the dhtrap call is:

dhtrap [-c community] [-d dest] [-f fromHost] dbName trapCode trapMessage

where

dhtrap
: The command executed that generates and sends the trap message to the SNMP Network Management System.

community
: Optional parameter specifying the community name to be included in the trap message.

dest
: Parameter specifying the symbolic name of the host where the Network Management System resides. The trap is sent here.

  This parameter is optional if the system has an snmpd configuration file containing the IP address of the host where the network management system resides.

fromHost
Optional parameter that tells the SNMP management station where the trap originated.

dbName
The symbolic name of the DB where the condition was detected. If no database name is associated with this, a '−' could be used as a filler. This parameter is displayed at the SNMP management station.

trapCode
An alphanumeric code associated with this trap; this code is user-defined. This parameter is displayed at the SNMP management station.

trapMessage
The complete text of the message for this trap. This parameter is displayed at the SNMP management station.

The execution of the dhtrap command with the appropriate parameters will produce a trap that is displayed on the SNMP management station.

For additional information on dhtrap configuration requirements and usage, enter dhtrap -? at a UNIX command line.

Appendixes

Appendix A. Working with Queries in the DataHub Query Window

You can use the Query Window to write, load, plan, execute, and save database queries while in your DataHub session.

To use the Query Window, you should first select the database on the object tree for which the query is intended.

To open the DataHub Query Window

Select the **Options** pull-down menu on the menu bar and select the **Query** choice. A window opens in which you can work with queries.

In this window are:

- Bordered rectangular area
- Menu bar with **File**, **Options**, and **Help** pull-down menus
- **Execute** and **More** push buttons

The bordered rectangle at the top of the window is a text input area where you can enter a query from the keyboard. You can also load a query from a file by selecting the **Open** choice from the **File** pull-down menu.

This window is independent of the main DataHub window. You can move the Query window to another part of your screen and work with the object tree in the main DataHub window, and then come back to the Query window to work with queries.

Please note: To get online help in the Query window, select the **Help** pull-down menu in the Query window itself.

Entering Queries

You can enter queries from the keyboard directly or load them from a file.

To enter a query from the keyboard

Select the rectangular area and begin typing.

Tips:

1. You can enter any SQL statement that is valid for the database selected for this query; you can use standard comments in the text input area.
2. You do not have to delimit the SQL statement with a ';'.

To load a query from a file

1 Select the **File** pull-down menu and select **Open**. The Select File window opens.

For information on using filters in this window, see online help.

2 Enter the path to the file and the file name in the Selection field and click on **sql_script**. The SQL query is displayed in the rectangular text input area.

You can edit the loaded query by placing the cursor at the point where you want to edit and typing or deleting characters.

You can issue only one single SQL statement at a time in the text input area; however, you can comment out any unwanted SQL statements in the input area.

Please note: DataHub does not automatically commit any DML statement; for example, the ORACLE commit convention is observed.

Submitting Queries

After creating a query in the text input area, submit it by clicking on the **Execute** push button at the bottom of the Query window. Output from the query appears in the area below the text input area. If the output exceeds the allotted screen area, use the scroll bars to see more output.

If the output exceeds 30 lines of text, click on the **More** push button to see the next section of query output.

Explaining Queries

You can use the **Explain** choice under the **Options** pull-down menu in the Query window to determine the results of a query before submitting it. Before using **Explain**, select a database from the object tree window.

The results from **Explain** are database specific. ORACLE and DB2 for AIX query paths are shown graphically. Other database system query paths are shown in text format.

Clearing Queries

To clear the query currently in the text input area, select the **File** pull-down menu in the Query window and select the **New** choice.

Saving Queries

You can save the query shown in the text input area by using the **Save Query** or **Save As** choices in the **File** pull-down menu of the Query window.

Use **Save Query** if you want to save the query to the file from which it was loaded.

Use **Save As** if you want to save a new query to a file or save a loaded query to a different file.

Exiting from the Query Window

To exit from the Query window, select the **Exit** choice from the **File** pull-down menu in the Query window.

Appendix B. Extracting Lists of DataHub Predefined Checks

All of DataHub's predefined checks are stored in the dba_pred_checks table in the control database. You can extract a list of the predefined checks by running an SQL query against this table using the Query window. You can get a list of all predefined checks, or a list for a particular type of check; for example, all predefined checks for DB2 Version 2 databases.

You need to know the name of the owner of the dba_pred_checks table; this is usually the user name that was used to install DataHub.

To see a list of all DataHub predefined checks

1 In the object tree, select the occurrence of the DataHub control database.

2 Select the **Options** menu and select **Query**. The DataHub Query window opens.

3 Enter the following SQL SELECT statement; substitute the owner of the table in your control database where *owner* appears.

```
select check_name, check_comments, check_object,
       check_db_type, check_os_type
from owner.dba_pred_checks
```

4 To run the SQL statement, select the **Execute** push button.

The selected rows are displayed in the lower part of the window.

You can qualify the SELECT statement to see all checks for a particular host or database type.

To see a list of DataHub predefined checks for a particular host type

Use the following SQL statement in the preceding instructions:

```
select check_name, check_comments, check_object,
       check_db_type, check_os_type
from owner.dba_pred_checks
where check_os_type='host-type' and
      check_object='host'
```

Substitute the host type you are interested in for *host-type* in the SQL statement; for example, check_os_type='aix' if you want all predefined checks you can use for AIX hosts.

The valid host type values are

Host type	*host-type* value
AIX	aix
HP	hpux
Sun	solaris

To see a list of DataHub predefined checks for a particular database type

Use the following SQL statement in the preceding instructions:

```
select check_name, check_comments, check_object,
        check_db_type, check_os_type
from owner.dba_pred_checks
where check_db_type='database-type' and
       check_object='database'
```

Substitute the database type you are interested in for *database-type* in the SQL statement; for example, check_db_type='db2_2' if you want all predefined checks you can use for DB2 Version 2 databases.

The valid database type values are

Database type	*database-type* value
DB2v1	db6000
DB2v2	db2_2
Ing	ing64
Ora	ora7
Syb	syb

Appendix C. Using Delimited Identifiers and Quoted Names

Some relational database management systems support the use of delimited identifiers or quoted names. DataHub supports the use of delimited identifiers and quoted names in your managed databases if their particular type of RDBMS offers this support. Where you can use delimited identifiers or quoted names depends on how the particular RDBMS supports them.

Using Delimited Identifiers in DB2

DB2 For DB2 databases, there are two types of identifiers:

- An *ordinary identifier* is a letter followed by zero or more characters, each of which is an uppercase letter, a digit, or the underscore character. An ordinary identifier must not be identical to a reserved word.
- A *delimited identifier* is a sequence of one or more characters enclosed within quotation marks (″). Two consecutive quotation marks are used to represent one quotation mark within the delimited identifier. In this way, an identifier can include lowercase letters.

For example, these are distinct identifiers

WKLY_SAL ″wkly_sal″

DB2v1 For DB2 Version 1, you can use delimited identifiers as names for

- Columns
- Constraints
- Indexes
- Tables
- Views

Delimited identifiers should follow standard naming conventions (see *DB2 Version 1 Administration Guide* for more information).

DB2v2 For DB2 Version 2, you can use delimited identifiers as names for

- Aliases
- Columns
- Constraints
- Data types (distinct)
- Events
- Functions
- Indexes

- Packages
- Tables
- Table spaces
- Triggers
- Views

For some objects, you can specify a delimited identifier for the object owner field in the Create window for that object. This field corresponds to the schema associated with the object (for information on schemas, see the *DB2 Version 2 Administration Guide*). These object are: tables, views, aliases, user-defined data types, functions, indexes, and triggers.

Delimited identifiers should follow standard naming conventions (see *DB2 Version 2 Administration Guide* for more information).

Using Quoted Names in ORACLE

Ora In ORACLE databases, *quoted names* are names enclosed in double quotes ("). Quoted names can contain any combination of characters. They can be case-sensitive, can use reserved words (including the word DUAL), and are not limited to certain types of characters. They can also include spaces. Quoted names can begin with characters other than an alphabetic character.

Once you have given an object a quoted name, you must use the double quotes whenever you refer to the object. For more information on quoted names, see the *ORACLE SQL Reference Manual*.

Ora For ORACLE, you can use delimited identifiers as names for

- Clusters
- Columns
- Constraints
- Database links
- Functions
- Indexes
- Packages
- Procedures
- Profiles
- Roles
- Rollback segments
- Sequences
- Snapshots
- Snapshot logs
- Synonyms
- Tables

- Table spaces
- Triggers
- Views
- Users

By using quoted names, you can give the following names to different objects in the same namespace

```
emp  "emp"  "Emp"  "EMP "
```

Conversely, ORACLE interprets the following names as identical and they cannot be used for different objects in the same namespace

```
emp  EMP  "EMP"
```

If you cannot give a user a quoted name, the name cannot contain lowercase letters.

Using Delimited Identifiers in Sybase Databases

Syb In Sybase databases, delimited identifiers are object names enclosed in double quotes ("). By using delimited identifiers, you can avoid certain restrictions on object names. They can be reserved words, can begin with nonalphabetic characters, and are not limited to certain types of characters as other identifiers are. Delimited identifiers cannot exceed 28 bytes.

Syb For Sybase, you can use delimited identifiers as names for

- Columns
- Tables
- Views

Before creating or referencing a delimited identifier, you must turn on the Sybase quoted_identifier option.

To turn on this option using DataHub, click on the **Options** menu and select **Query**. In the Query window, enter

```
set quoted_identifier on
```

and click on the **Execute** push button.

If you turn on this option using the Query window, you can use delimited identifiers within your current connection to the Sybase server. If you disconnect and then reconnect to the Sybase server, you need to turn on the option again to use delimited identifiers.

While this option is on, do not use double quotes around character or date strings; use single quotes instead. For example, to insert a character string into column col1 of table 1foryou, use

```
insert "1foryou"(col1) values ('abc')
```

not

```
insert "1foryou"(col1) values ("abc")
```

For more information, see *Sybase SQL Server Reference Manual*.

Appendix D. Customizing Fonts or Colors for DataHub

You can customize the fonts or colors of DataHub for all users of that DataHub control point or just for your own control point sessions. To customize fonts, you need to know what fonts and colors are available on your system and how to use a UNIX editor that you can access on your system.

AIX 4.1 users: When running under the Common Desktop Environment (CDE), colors are set by CDE. See your system administrator for information on customizing fonts and colors under CDE.

To customize the fonts or colors for all users of a particular DataHub control point, you edit the *<installdir>* /startup/XDataHub resource file. For more information, see "Customizing Fonts or Colors for All Control Point Sessions."

To customize the fonts or colors for you own DataHub control point sessions, you can do either one of the following actions:

- Edit the .Xdefaults file in your home directory (see "Customizing Your .Xdefaults File" on page 352).
- Copy the XDataHub file from the startup directory to your home directory and edit that copy of the file (see "Customizing Your XDataHub File" on page 354).

Customizing Fonts or Colors for All Control Point Sessions

You customize fonts or colors for all Control Point sessions by editing the *<installdir>*/startup/ XDataHub resource file on the system where the DataHub session is invoked. Figure 16 on page 350 shows an example of an XDataHub file.

```
!
!
! Font Specifications
!
!*fontList                        : Rom14, RomanKn12, Kanji12    1
!*font                            : Rom14                        2
!
! Text Specifications
!
*title                            : Datahub for UNIX
!
! User Interface Behaviour Specifications
!
!*geometry                        : 750x580+546+134
*saveUnder                        : true
*defaultPosition                  : false
*listSizePolicy                   : xmCONSTANT
*resizePolicy                     : XmRESIZE_NONE
*XmLabel*resizePolicy             : XmRESIZE_ANY
*XmMessageBox*resizePolicy        : XmRESIZE_ANY

!
! Color Specifications
!
*background                       : white    3
*foreground                       : black    4
*selectColor                      : green
!
*iconImage:            /u/sy950123/admin/icons/dhubnew.xbm
*iconImageBackground:  #ff0000
*iconImageForeground:  #00ff00
*iconName:                  Datahub
!
! End of XDataHub Resource File
!
```

Figure 16. Example XDataHub file

To customize the fonts or colors

1 Open the XDataHub file with an editor.

Please note that the lines for *font and *fontList are initially commented out with the exclamation point (!) (see 1 and 2).

2 *To change the fonts*:

a Remove the exclamation point (!) from the left of *font and *fontList.

b You specify the fonts allowed for DataHub in the *fontList variable (see 1 in Figure 16). If the font you want to use is not shown in this list of fonts, add the name of the font to the list. If the *fontList variable is not listed in the XDataHub file, add a line like the one in Figure 16

For example, if you want to use the Itl14 font, add Itl14 to the list:

```
*fontList                    : Rom14, RomanKn12, Kanji12, Itl14
```

You can also delete fonts from the list.

c The font used by DataHub is specified in the *font variable (see 2 in Figure 16 on page 350). In the example, Rom14 is used. Change this font to the one you want to use. If the *font variable is not listed in the XDataHub file, add a line like the one in Figure 16 on page 350

For example,

```
*font                        : Itl14
```

3 *To change the colors*:

a The background color for DataHub windows is specified by the *background variable (see 3 in Figure 16 on page 350). Change this color to the one you want to use. If the *background variable is not listed in the XDataHub file, add a line like the one in Figure 16 on page 350

For example,

```
*background                  : thistle
```

b The color of the window text for DataHub, except for the text in the Object Tree window, is specified by the *foreground variable (see 4 in Figure 16 on page 350). Change this color to the one you want to use. If the *foreground variable is not listed in the XDataHub file, add a line like the one in Figure 16 on page 350

For example,

```
*foreground                  : navy
```

4 Save the XDataHub file.

Important:

1. Do not leave blanks at the end of a line. If you leave blanks at the end of a line, the system will not recognize font or color names.
2. Do not change any resources other than the fonts and colors mentioned in the preceding section.

FYI:

For Japanese to work in DataHub, you need to add the following lines to the XDataHub file

- *fontList : *gothic*
- /usr/lib/X11/fonts/JP to the fontpath with xset

Customizing Fonts or Colors for Your Control Point Sessions

You customize fonts or colors for your Control Point sessions by either editing the .Xdefaults file in your home directory or copying the XDataHub file to your home directory and editing it.

Customizing Your .Xdefaults File

Edit your .Xdefaults file on the system from which you are invoking DataHub. Changing your fonts and colors will affect all DataHub sessions or control points that you invoke from this account. Figure 17 shows an example of a .Xdefaults file.

```
XDesktop*pictureDirectory:   /usr/local/etc/make.dt/bitmaps
!#
!#
!# Component Appearance Resources
!#
!
!
!
Mwm*client*activeBackground:   cadetBlue
Mwm*icon*activeBackground:   green
Mwm*icon*background:   dimgray
Mwm*client*background:   gray
Mwm*feedback*background:   cadetBlue
Mwm*iconAutoPlace: true
Mwm*keyboardFocusPolicy:   pointer
Mwm*useIconBox:   false
Mwm*iconPlacement: left bottom
Mwm*buttonBindings:   DefaultButtonBindings
!
XDataHub*fontList                : Rom14
XDataHub*font                    : Rom14
!
```

Figure 17. Example .Xdefaults file

FYI:

For Japanese to work in DataHub, you need to add the following lines to the XDataHub file

- *fontList : *gothic*
- /usr/lib/X11/fonts/JP to the fontpath with xset

To customize your fonts or colors

1 Open your .Xdefaults file with an editor.

2 *To change the fonts*:

Put the following lines into the file:

- XDataHub*fontList : Rom14
- XDataHub*font : Rom14

You can substitute Rom14 with your preferred font.

3 *To change the colors*:

Put the following lines into the file:

- XDataHub*background : thistle
- XDataHub*foreground : navy

The background color for DataHub windows is specified by the *background variable. The color of the window text for DataHub, except for the text in the Object Tree window, is specified by the *foreground variable. You can substitute the colors in the example above with your preferred colors.

4 Save the .Xdefaults file.

Important: Do not leave blanks at the end of a line. If you leave blanks at the end of a line, the system will not recognize font or color names.

Please note that the change will not take effect until you restart the Motif Window Manager (mwm). When you start mwm, the resource lines are read from .Xdefaults into the resource database for that mwm session. The fonts and colors will affect all DataHub sessions that you invoke from the mwm session.

Edit your .Xdefaults file on the system from which you are invoking DataHub. Changing your fonts and colors will affect all DataHub sessions or control points that you invoke from this account.

Customizing Your XDataHub File

Instead of editing your .Xdefaults file, you can edit a copy of the XDataHub file to change fonts and colors for your own DataHub sessions. Figure 16 on page 350 shows an example of an XDataHub file.

> **FYI:**
>
> For Japanese to work in DataHub, you need to add the following lines to the XDataHub file
>
> - *fontList : *gothic*
> - /usr/lib/X11/fonts/JP to the fontpath with xset

To customize the fonts or colors

1 Copy the XDataHub file from the *installdir*/startup directory to your home directory.

2 Open the XDataHub file with an editor.

Please note that the lines for *font and *fontList are initially commented out with the exclamation point (!) (see 1 and 2 in Figure 16 on page 350).

3 Edit the file to add the DataHub application name (XDataHub) in front of the font or color resource types.

For example,

```
XDataHub*fontList        : Rom14
XDataHub*font            : Rom14

XDataHub*background      : thistle
XDataHub*foreground      : navy
```

By qualifying the resources with the XDataHub name, only DataHub sessions will be affected.

4 Save the XDataHub file.

5 Issue the following command to initiate the changes:

```
xrdb -screen -merge /XDataHub
```

If you want the change to persist across mwm sessions, use the following command instead:

```
xrdb -merge /XDataHub
```

Without the -screen option, the resource database will be updated.

Other types of Motif resources can be changed in a similar fashion. Consult the xrdb man page for more information on its usage.

Appendix E. Defining Printers to DataHub

Some of the windows in DataHub have a **Print** push button that allows you to print the contents of the window (usually a table or graph) to a printer. To be able to use this push button to print, you need to define the printers to DataHub.

You define printers to DataHub by editing the apps.pl file. You need to know the printer names and queue names that are available on your system.

To get the information about the printer queues

1 Issue the command **lpstat**. The result will show what printer queues are defined on your system.

You might want to direct the output of the **lpstat** command to a file to record the values; for example, issue the command **lpstat** > queue.info to record the values in the file queue.info.

To open the apps.pl file

1 Open a window on your workstation.

2 Change your current directory to the directory where the apps.pl file resides (usually the directory named “gui” in the DataHub install directory).

3 Open the apps.pl file with an editor.

Figure 18 shows an example of an apps.pl file.

```
db_login('db6000','bobcat', 'inst1','ctldb','!?','!?').
system_name('windmill').
initial_font( font( 'helvetica', xFalse, xFalse, 14)).
```

Figure 18. Example “apps.pl” file

To define printers to DataHub

1 Open the apps.pl file with an editor.

2 Add a line to the bottom of the apps.pl file for each printer you wish to define. Follow this format:

printer(hostname, printername, queuename, ps, 'description').

where

hostname — The name of the machine where the printer queue is defined. This is not necessarily where the printer actually resides.

printername — The name of the printer. Use the value from the Dev column of the **lpstat** output.

queuename — The name of the print queue. Use the value from the Queue column of the **lpstat** output.

ps — The exact character string "ps".

description — A description that is meaningful to you.

Figure 19 shows an example of an apps.pl file with a PostScript printer added. The name of the machine is "bobcat", the name of the printer is "lpt16", the name of the print queue is "st49d2ps", and the description chosen is "My PostScript printer in room D2".

```
db_login('db6000','bobcat', 'inst1','ctldb','!?','!?').
system_name('windmill').
initial_font( font( 'helvetica', xFalse, xFalse, 14)).
printer(bobcat, lpt16, st49d2ps, ps, 'My PostScript printer in room D2').
```

Figure 19. Example "apps.pl" file with printer defined

Appendix F. Data Type Conversions For Copying Tables

Different RDBMSs support different data types for columns. When you copy tables from one database type to another with DataHub, column types are converted. The following tables show how column types are converted for the different database types DataHub supports.

For descriptions of the column types, see the documentation for your RDBMS.

Table 6 (Page 1 of 2). Data type conversions from Sybase to other databases.

From Sybase	To ORACLE	To DB2 V.1	To DB2 V.2	To INGRES
BIT	NUMBER(1,0)	SMALLINT	SMALLINT	INTEGER1
TINYINT	NUMBER(4,0)	SMALLINT	SMALLINT	INTEGER1
SMALLINT	NUMBER(6,0)	SMALLINT	SMALLINT	INTEGER1
INT	NUMBER(12,0)	INTEGER	INTEGER	INTEGER
DECIMAL(p<=31,s)	NUMBER(p,s)	DECIMAL(p,s)	DECIMAL(p,s)	FLOAT4
DECIMAL(p>31,s)	NUMBER(p,s)	DECIMAL(31,s) (note 1)	DECIMAL(31,s) (note 1)	FLOAT4
NUMERIC(p<=31,s)	NUMBER(p,s)	DECIMAL(p,s)	DECIMAL(p,s)	FLOAT
NUMERIC(p>31,s)	NUMBER(p,s)	DECIMAL(31,s) (note 1)	DECIMAL(31,s) (note 1)	FLOAT
FLOAT(p)	NUMBER(p)	FLOAT	FLOAT	FLOAT
REAL	NUMBER	FLOAT	FLOAT	FLOAT
BINARY(n<255)	RAW(n)	CHAR(n) for BIT data	CHAR(n) for BIT data	No conversion
BINARY(n=255)	RAW(255)	VARCHAR(255) for BIT data	VARCHAR(255) for BIT data	No conversion
VARBINARY(n)	RAW(n)	VARCHAR(n) for BIT data	VARCHAR(n) for BIT data	No conversion
CHAR(n<=254)	CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)
CHAR(255)	CHAR(255)	VARCHAR(255)	VARCHAR(255)	CHAR(255)
VARCHAR(n)	VARCHAR2(n)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)
DATETIME	DATE	TIMESTAMP	TIMESTAMP	DATE
SMALLDATETIME	DATE	TIMESTAMP	TIMESTAMP	DATE
TIMESTAMP	RAW(8)	VARCHAR(8) for BIT data	VARCHAR(8) for BIT data	DATE
MONEY	NUMBER(38,2)	DECIMAL(31,2)	DECIMAL(31,2)	MONEY
SMALLMONEY	NUMBER(12,2)	DECIMAL(12,2)	DECIMAL(12,2)	MONEY
PRICE	NUMBER(38,2)	DECIMAL(31,2) (note 1)	DECIMAL(31,2) (note 1)	No conversion
NCHAR(n)	CHAR(n) (note 2)	GRAPHIC(n)	GRAPHIC(n)	VARCHAR(2*n)

Table 6 (Page 2 of 2). Data type conversions from Sybase to other databases.				
From Sybase	**To ORACLE**	**To DB2 V.1**	**To DB2 V.2**	**To INGRES**
NVARCHAR(n)	VARCHAR(n) (note 2)	VARGRAPHIC(n)	VARGRAPHIC(n)	VARCHAR(2*n)

Notes:

1. Possible loss of significant data.
2. Using a multibyte code set.

Table 7. Data type conversions from ORACLE to other databases.

From ORACLE	To Sybase	To DB2 V.1	To DB2 V.2	To INGRES
NUMBER(p<=31,s)	NUMERIC(p,s)	DECIMAL(p,s)	DECIMAL(p,s)	FLOAT
NUMBER(p>31,s)	NUMERIC(p,s)	DECIMAL(31,s) (note 1)	DECIMAL(31,s) (note 1)	FLOAT
CHAR(n<=254)	CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)
CHAR(n=255)	CHAR(n)	VARCHAR(n)	VARCHAR(n)	CHAR(n)
VARCHAR2(n<=255)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)
VARCHAR2(n>255)	TEXT	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)
RAW(n)	BINARY(n)	VARCHAR(n) for BIT data	VARCHAR(n) for BIT data	No conversion
DATE	DATETIME	TIMESTAMP	TIMESTAMP	DATE
CHAR(n) (note 2)	NCHAR(n)	GRAPHIC(n)	GRAPHIC(n)	CHAR(n)
VARCHAR(n) (note 2)	NVARCHAR(n)	VARGRAPHIC(n)	VARGRAPHIC(n)	VARCHAR(n)

Notes:

1. Possible loss of significant data.
2. Using a multibyte code set.

Table 8. Data type conversions from DB2 Version 1 to other databases.

From DB2 V.1	To DB2 V.2	To ORACLE	To Sybase	To INGRES
SMALLINT	SMALLINT	NUMBER(6,0)	SMALLINT	INTEGER1
INTEGER	INTEGER	NUMBER(12,0)	INT	INTEGER
FLOAT	FLOAT	NUMBER	FLOAT	FLOAT
DECIMAL(p,s)	DECIMAL(p,s)	NUMBER(p,s)	NUMERIC(p,s)	FLOAT
DATE	DATE	DATE	DATETIME	DATE
TIME	TIME	DATE	DATETIME	DATE
TIMESTAMP	TIMESTAMP	DATE	DATETIME	DATE
CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)
VARCHAR(n<=255)	VARCHAR(n)	VARCHAR2(n)	VARCHAR(n)	VARCHAR(n)
VARCHAR(255<n<=2000)	VARCHAR(n)	VARCHAR2(n)	TEXT	VARCHAR(n)
VARCHAR(n>2000)	VARCHAR(n)	LONG (note 1)	TEXT	VARCHAR(2000) (note 3)
LONG VARCHAR	LONG VARCHAR	LONG (note 1)	TEXT	VARCHAR(2000) (note 3)
CHAR(n) for BIT data	CHAR(n) for BIT data	RAW(n)	BINARY(n)	No conversion
VARCHAR(255<=n) for BIT data	VARCHAR(n) for BIT data	RAW(n)	BINARY(n)	No conversion
VARCHAR(255>n) for BIT data	VARCHAR(n) for BIT data	LONG RAW (note 1)	IMAGE	No conversion
LONG VARCHAR for BIT data	LONG VARCHAR for BIT data	LONG RAW (note 1)	IMAGE	No conversion
GRAPHIC(n)	GRAPHIC(n)	CHAR(n) (note 2)	NCHAR(n)	CHAR(2*n)
VARGRAPHIC(n)	VARGRAPHIC(n)	VARCHAR(n) (note 2)	NVARCHAR(n)	VARCHAR(2*n)
LONG VARGRAPHIC(n)	LONG VARGRAPHIC(n)	LONG(n) (note 2)	TEXT(n)	VARCHAR(2000) (note 3)

Notes:

1. Only one LONG field allowed in ORACLE
2. Using a multibyte code set.
3. Loss of information. Table row width limited to 2000 bytes in INGRES

Table 9. Data type conversions from DB2 Version 2 to other databases.				
From DB2 V.2	**To DB2 V.1**	**To ORACLE**	**To Sybase**	**To INGRES**
SMALLINT	SMALLINT	NUMBER(6,0)	SMALLINT	INTEGER1
INTEGER	INTEGER	NUMBER(12,0)	INT	INTEGER
FLOAT	FLOAT	NUMBER	FLOAT	FLOAT
DECIMAL(p,s)	DECIMAL(p,s)	NUMBER(p,s)	NUMERIC(p,s)	FLOAT
DATE	DATE	DATE	DATETIME	DATE
TIME	TIME	DATE	DATETIME	DATE
TIMESTAMP	TIMESTAMP	DATE	DATETIME	DATE
CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)
VARCHAR(n<=255)	VARCHAR(n)	VARCHAR2(n)	CHAR(n)	VARCHAR(n)
VARCHAR(255<n<=2000)	VARCHAR(n)	VARCHAR2(n)	TEXT	VARCHAR(n)
VARCHAR(n>2000)	VARCHAR(n)	LONG (note 1)	TEXT	VARCHAR(2000) (note 3)
LONG VARCHAR	LONG VARCHAR			VARCHAR(2000) (note 3)
CHAR(n) for BIT data	CHAR(n) for BIT data	RAW(n)	BINARY(n)	No conversion
VARCHAR(n<=255) for BIT data	VARCHAR(n) for BIT data	RAW(n)	BINARY(n)	No conversion
VARCHAR(255>n) for BIT data	VARCHAR(n) for BIT data	LONG RAW (note 1)	IMAGE	No conversion
LONG VARCHAR for BIT data	LONG VARCHAR for BIT data	LONG RAW (note 1)	IMAGE	No conversion
GRAPHIC(n)	GRAPHIC(n)	CHAR(n) (note 2)	NCHAR(n)	CHAR(2*n)
VARGRAPHIC(n)	VARGRAPHIC(n)	VARCHAR(n) (note 2)	NVARCHAR(n)	VARCHAR(2*n)
LONG VARGRAPHIC(n)	LONG VARGRAPHIC(n)	LONG(n) (note 2)	TEXT(n)	VARCHAR(2000) (note 3)

Notes:

1. Only one LONG field allowed in ORACLE
2. Using a multibyte code set.
3. Loss of information. Table row width limited to 2000 bytes in INGRES

Table 10. Data type conversions from INGRES to other databases.

From INGRES	To DB2 V.1	To DB2 V.2	To ORACLE	To Sybase
CHAR(n) or Cn (1<=n<=254)	CHAR(n)	CHAR(n)	CHAR(n)	CHAR(n)
CHAR(n) or Cn (255<=n<=2000)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	TEXT
TEXT(n) (1<=n<=254)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)
TEXT(n) (255<=n<=2000)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	TEXT
VARCHAR(n) (1<=n<=254)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)
VARCHAR(n) (255<=n<=2000)	VARCHAR(n)	VARCHAR(n)	VARCHAR(n)	TEXT
INTEGER1	SMALLINT	SMALLINT	NUMBER(3,0)	TINYINT
SMALLINT	SMALLINT	SMALLINT	NUMBER(6,0)	SMALLINT
INTEGER	INTEGER	INTEGER	NUMBER(12,0)	INTEGER
FLOAT4	FLOAT	FLOAT	NUMBER	FLOAT(4)
FLOAT	FLOAT	FLOAT	NUMBER	FLOAT(8)
MONEY	DECIMAL(31,2)	DECIMAL(31,2)	NUMBER(38,2)	MONEY
DATE	TIMESTAMP	TIMESTAMP	DATE	DATETIME (note 1)
TABLE_KEY	No conversion	No conversion	No conversion	No conversion
OBJECT_KEY	No conversion	No conversion	No conversion	No conversion

Notes:

1. Possible loss of significant data

Appendix G. Logging the DataHub Display Window

The DataHub display window shows the SQL statements used by DataHub. You can write the contents of the display window to a log file by using the **Start Logging** choice in the **Options** menu. The log file will be stored on the host at which the control point is running.

To log the DataHub display window to a file

1 Click on the **Options** menu and click on **Start Logging**.

The Select File window opens. For information about the fields in this window, see online help.

The Selection field initially shows the HOME directory for the operating system user who started this DataHub session. Specify the file name for your log file in this field.

2 Click on the **Log File** push button.

From this point on, the statements that are displayed in the DataHub display window are logged to the specified file.

Bibliography

This bibliography includes all publications cited in this book; it also lists publications that provide additional information for DataHub users.

The DataHub for UNIX Operating Systems Library

The following books comprise the DataHub for UNIX Operating Systems library:

- *DataHub for UNIX Operating Systems User's Guide*, SC26-8668

  This book describes how to use the DataHub product. It uses a task-oriented approach, describing actions you can take rather than how the product works.

- *DataHub for UNIX Operating Systems Installation and Configuration Guide*, SC26-8669

  This book describes installation, configuration, customization, and troubleshooting procedures for DataHub.

Related IBM Publications

This section lists books associated with IBM products.

AIX/6000 Publications

- *AIX Version 3.2 for RISC System/6000 Installation Guide*, SC23-2341
- *AIX Version 3.3 for RISC System/6000 Commands Reference*, GBOF-1802
- *AIXwindows Environment/6000 AIXwindows and AIXwindows Desktop User's Guide*, GC23-2432

DB2 for AIX Publications

- *DB2/6000 Administration Guide*, SC09-1571
- *DB2 for AIX Information and Concepts Guide*, S20H-4664
- *DB2 for AIX Administration Guide*, S20H-4580
- *DB2 for AIX Command Reference*, S20H-4645
- *DB2 for AIX API Reference*, S20H-4984
- *DB2 for AIX SQL Reference*, S20H-4665
- *DB2 for AIX Database System Monitor Guide and Reference*, S20H-4871
- *DB2 for AIX Messages Reference*, S20H-4808
- *DB2 for AIX Problem Determination Guide*, S20H-4779
- *DB2 for AIX Planning Guide*, S20H-4758
- *DB2 for AIX Installation and Operation Guide*, S20H-4757
- *DB2 for AIX Master Index and Glossary*, S20H-4781

DB2 for HP-UX Publications

- *DB2 for HP-UX Installation and Operation Guide*, S34H-0898
- *DB2 for HP-UX Planning Guide*, S34H-0897

DB2 for AIX publications listed under "DB2 for AIX Publications" on page 364 (except for S20H-4757 and S20H-4758) apply to DB2 for HP-UX.

DB2 for Solaris Publications

- *DB2 for the Solaris Operating Environment Installation and Operation Guide*, S34H-0893
- *DB2 for the Solaris Operating Environment Planning Guide*, S34H-0892

DB2 for AIX publications listed under "DB2 for AIX Publications" on page 364 (except for S20H-4757 and S20H-4758) apply to DB2 for Solaris.

Publications of Other Companies

This section lists RDBMS books produced by other companies.

INGRES Publications

- *INGRES Database Administrator's Guide (UNIX)*, 64-99-17901
- *INGRES Installation and Operations Guide (UNIX)*, 6-99-#io9-64
- *INGRES Master Index (UNIX)*, 64-99-18902
- *INGRES Object Management User's Guide*, 63-9(9)-47301
- *INGRES/QUEL Command Summary*, 64-9(9)-48106

Oracle Publications

- *ORACLE7 Server Concepts Manual*
- *ORACLE7 PL/SQL User's Guide and Reference*
- *ORACLE7 Messages and Codes Manual*
- *ORACLE7 SQL*Plus User's Guide and Reference*
- *ORACLE7 Server Administrator's Guide*
- *ORACLE7 Server Application Developer's Guide*
- *ORACLE7 Server Utilities User's Guide*

SYBASE SQL Publications

- *SYBASE SQL Server Administration Guide*
- *SYBASE SQL Server Master Index*
- *SYBASE SQL Server Reference Manual*
- *SYBASE SQL Server Troubleshooting Guide*
- *SYBASE SQL Server Utility Programs for UNIX*

- *Transact-SQL User's Guide*
- *Open Client Embedded SQL Reference Manual*

Terms and Abbreviations

This glossary contains information about the terms and abbreviations used in the DataHub for UNIX Operating Systems user interface and publications. Many definitions include additional information about how the term applies when you use DataHub for UNIX Operating Systems.

This list is not exhaustive; you should refer to the manuals for your particular RDBMS for the definitions of database objects particular to that RDBMS.

A

action. A defined task that DataHub performs. An action might be *to display* database activity, *to add* authorizations, or *to run* SQL.

action log. A file on a managed host to which is written watchdog actions and results.

action type. A category of event that occurs when a watchdog rule condition is met. An alert is one action type.

Action window. The left window of the three windows that together make up the DataHub for UNIX Operating Systems *main window*. It contains zero, one, or more action push buttons which represent actions that can be performed on the objects selected in the Object Tree window.

Advanced Interactive Executive (AIX or AIX/6000). IBM′s operating system for the RISC System/6000 workstation.

AIX. See *Advanced Interactive Executive.*

AIX/6000. See *Advanced Interactive Executive.*

alert. (1) The screen object, usually containing a text message and a flashing color, sent to a named user in response to a condition reported by a watchdog and evaluated by a watchdog rule. (2) An action type associated with a watchdog rule. When a watchdog rule is defined, part of the definition is the watchdog rule action type. Possible action types include sending an alert, launching a program, and generating an SNMP trap.

See also *watchdog rule* and *watchdog check*.

alias. (1) In DB2, an alternative name used to identify an object or a database. (2) In Sybase, an alternative name used to identify a database user.

authorization. A right granted to a user or group of users to communicate with or make use of a computer system or database.

automated operations. DataHub functions that can take automatic corrective actions for databases and operating systems. The automatic operations occur when a predefined condition or set of conditions is met (defined using watchdog checks and rules). See *offline monitoring*.

B

backup. (1) To copy information onto a diskette, fixed disk, tape, or other storage media for record keeping or recovery purposes. (2) A copy of one or more table spaces or a copy of an entire database at a given point in time. This copy is saved for future reference or for recovery purposes in case the original data is damaged or lost.

C

catalog. See *database catalog*.

cascading pull-down menu. A pull-down menu invoked by selecting an option on another pull-down menu.

check. See *database check* and *watchdog check*.

collapse. (1) To temporarily prune or remove branches of the tree currently displayed in the DataHub for UNIX Operating Systems Object Tree window. A collapse operation is reversed by selecting **Expand**.

(2) A term relating to the Table of Contents of an online publication. To collapse a topic, you can use the mouse to click on the minus sign beside the topic. The plus sign that appears beside the topic indicates that you can *expand* the topic.

commit. In SQL, the process that allows data changed by one application or user to be used by other applications or users. When a commit operation is complete, the locks are released to allow other applications to use the changed data.

configuration. The process used to set up the DataHub files and environment according to user specifications. This process sets up the control point application, loads the control database, and sets up and starts the watchdog server, driver, and watchdog processes. This process is used on the control point system and the managed hosts. The installation process must be completed prior to starting the configuration process.

container. A generic term describing an allocation of space to a table space. Depending on the table space type, a container can be a directory name, a device name, or a file name.

contextual help. Help that gives information about the specific item the cursor is on.

Contrast with *general help.*

control database. A database containing DataHub for UNIX Operating Systems tables, watchdog rules, watchdog checks, alerts, users, managed systems, and other such information.

control point. The DataHub for UNIX Operating Systems control point is the programmable workstation from which you invoke DataHub functions.

Contrast with *managed host.*

See also *managed database.*

D

daemon. An unattended UNIX process that provides services upon request or periodically. DataHub watchdogs are examples of daemons.

database. A collection of data with a given structure for accepting, storing, and providing data for multiple users on demand.

database administrator (DBA). An individual or group responsible for the design, development, operation, safeguarding, maintenance, and use of a database.

database catalog. A set of system tables maintained by the database manager. Catalog tables are created when the database is created and contain information about database objects such as tables, views, and indexes and their relationships.

database device. A Sybase device for storing the objects that make up databases in a Sybase server.

database driver. A process supporting DataHub database operations.

DataHub provides database drivers for all RDBMSs that it manages. See *DataHub for UNIX Operating Systems* for a list of RDBMSs. See also *driver*.

database watchdog. An unattended process that performs watchdog checks on a managed database at predefined times or time intervals.

DATABASE 2 for AIX (IBM DB2 for AIX). A relational database management system designed for the AIX platform. The major components include database services and facilities to access data, the DBA utility, and application interfaces. DB2 for AIX allows users to create, update, and access databases.

data dictionary. For ORACLE databases, the data dictionary is a set of tables that provides information about its associated database.

DataHub. See *DataHub for UNIX Operating Systems*.

DataHub for UNIX Operating Systems. An IBM product that provides database and system administration and monitoring functions. DataHub for UNIX Operating Systems manages DB2, INGRES, ORACLE, and SYBASE SQL Server databases. DataHub for UNIX Operating Systems operates on and manages AIX, HP-UX, and Sun Solaris.

DataHub group. An object type that appears in the object tree, the occurrences of which represent more than one object. There are group objects for hosts, servers, and databases. See also *group member* and *group object*.

DataHub for UNIX Operating Systems window. The main product window. Composed of three distinct areas: Action window, Object Tree window, and Navigation window. All three windows change dynamically to reflect your selections, available actions, and your navigation path through and current position in the Object Tree window.

DBA. See *database administrator.*

DB2. See *DATABASE 2 for AIX*.

default device. The device chosen by the Sybase system when no device is specified.

delimited identifier. In SQL, a sequence of one or more characters enclosed within quotation marks.

device. In Sybase systems, a disk space or a file in the file system that is used to store databases and their objects.

distributed relational database system. A system involving multiple locations connected together in a communications network, in which each location may have one or more like or unlike relational database systems. In a distributed relational database management system, a user at one location can access any data in the network as if the data were stored at the user's location.

DNS. See *Domain Name Service*.

Domain Name Service. A mechanism, in TCP/IP, that tracks the host names in a network. A domain name consists of a sequence of sub-names separated by a delimiter character.

double-byte character set (DBCS). A set of characters in which each character occupies two bytes. Languages, such as Japanese, Chinese, and Korean, that contain more symbols than can be represented by 256 code points require double-byte character sets. Entering, displaying, and printing DBCS characters requires special hardware and software support.

driver. A process supporting DataHub operations on a specific operating system or specific RDBMS. A minimum DataHub configuration requires one operating system driver and one database driver (for the control database). See also *operating system driver* and *database driver.*

dump device. A Sybase device for storing backup copies of databases or transaction logs.

E

entry field. An area of the window where you can type information. Its boundaries are usually indicated. See also *selection field.*

event. An occurrence of significance to a task. In an INGRES RDBMS, an event is any type of a program detectable occurrence.

expand. (1) To display collapsed branches of the object tree currently displayed in the DataHub for UNIX Operating Systems window. A plus (+) symbol inside a circle to the right of a branch indicates that the branch can be expanded.

(2) A term relating to the Table of Contents of an online publication. To expand a topic, click on the plus sign beside the topic.

A minus sign beside a topic indicates that you can *collapse* the topic.

extended help. Help that provides information about the contents of the window from which a user requested help. Contrast with *contextual help*.

F

filter. (1) A value you can specify to limit the objects displayed in the DataHub for UNIX Operating Systems Object Tree window. For example, filters allow you to limit the number of tables that will be displayed on the object tree to those owned by a specific user or those tables with names starting with a specific character string. (2) A value you can specify to establish a threshold for a watchdog check. When the threshold is reached or exceeded, the check results are evaluated by rules. Watchdog filters can help DBAs implement a management-by-exeception system.

fire. To initiate the action specified by a rule when the condition stated by the rule is satisfied.

foreign key. A column or set of columns in a table whose values are required to match at least one primary key value of a row of its parent table.

function. A program or piece of a program that can be called. For DB2, a function represents a specific purpose of an entity or its characteristic action, such as a column function or a scalar function. For ORACLE, a function is a PL/SQL block that contains procedural constructs and embedded SQL statements that can have input arguments only.

G

general help. In DataHub, a help panel providing general information about the DataHub for UNIX Operating Systems window. This panel is invoked from the Help menu.

Contrast with *contextual help* and *extended help*.

global rules evaluation. The analysis of monitoring results from resources on multiple hosts for cross-system problems. See *watchdog server*.

graying. An indication that a choice is unavailable.

group. (1) A logical organization of users that have IDs according to activity or resource access authority. (2) DataHub group.

group member. A member of a DataHub group, such as a host, server, or database. Group members can be displayed in the object tree below the group to which they belong. See also *DataHub group*.

group object. DataHub groups can have group objects that represent an object that is common to all of the members of the group. Group objects are displayed in the object tree below the group to which they belong. For example, a database group could have group users, group table spaces, and group tables. See also *DataHub group*.

guard. An optional component of the rule text for a watchdog rule that determines whether the condition is evaluated.

GUI. graphical user interface.

H

heterogeneous environment. A computing environment composed of differing hardware and software components typically supplied by multiple vendors. DataHub works across different UNIX systems and controls RDBMSs from various database vendors.

hide. To remove objects and branches of objects from the Object Tree window. Objects are not actually removed from the system or RDBMS; Hide merely removes objects from the Object Tree window.

host. A local or remote workstation running a DataHub-supported operating system. DataHub manages host operating systems and host RDBMSs.

HP-UX. Hewlett-Packard operating system for UNIX workstations

I

identifier. One or more characters used to identify or name a data element and possibly to indicate certain properties of that data element.

index. A set of row pointers that are logically ordered by the values of a key. Indexes provide quick access to data and can enforce uniqueness on the rows in the table.

index key. A column or an ordered collection of columns on which an index is defined. See also *key*.

installation. The process used to copy DataHub files from the install media to the DataHub directories on the control point and managed hosts.

instance. A logical database manager environment similar to an image of the actual database manager environment.

intelligent agent. See *watchdog*.

J

join. A relational operation that allows for retrieval of data from two or more tables based on matching column values.

join condition. A condition where two tables are brought together and compared; rows from one table are selected when columns from that table match columns (over a condition) from the other table.

journal. A file used to record changes made in a system. Synonymous with *log*.

K

key. A column or an ordered collection of columns that uniquely identifies a row.

L

LAN. See *local area network.*

launch. The initiation of an executable program, shell script, or command. A launch is one of the three options you can specify in a watchdog rule as the action to be taken if the conditions specified in the watchdog rule are met.

local area network (LAN). A network of devices that are connected to one another for communication. A local area network can be connected to another LAN or to a larger network (wide area network or WAN). See also *wide area network*.

local rules evaluation. Analysis of monitoring results from a specific managed host done at that host.

log. (1) To record; for example, to log all messages on the system printer. (2) See *journal*, *action log*.

login. In a Sybase RDBMS, the name used to log in to a SQL Server.

M

managed database. A database defined to the DataHub control point. The DataHub control point always has at least one managed database (the control database).

managed host. An operating system that is defined to the DataHub control point.

managed system. See *managed host*.

management by exception. See *filter*.

menu bar. The rectangular area at the top of the DataHub for UNIX Operating Systems window containing headings for pull-down menus. The headings, called menu choices, are File, Tree, Options, Administer, and Help. Click on menu choices to access pull-down menus. For example, when you select **Help** on the menu bar, the **Help** pull-down menu is displayed.

N

Navigation window. The window on the right side of DataHub for UNIX Operating Systems window, showing the path you travelled in the Object Tree window.

O

object class. See *object type*.

object instance. See *object occurrence*.

object occurrence. An element of the environment managed by DataHub. Each named object under an object type (host, database, package, and so on) is an occurrence of that object. In DataHub, an example of an object occurrence might be a relational database named CELDIAL that appears under the Database object type in the DataHub for UNIX Operating Systems Object Tree window.

Object Tree window. The window in the middle of the DataHub for UNIX Operating Systems window. The Object Tree window contains a list of managed object types and object occurrences within the DataHub for UNIX Operating Systems window.

The objects in the DataHub for UNIX Operating Systems window are displayed in a tree formation. The tree structure shows relationships between hosts, databases, and related database objects. Objects are shown in a hierarchical list with connecting lines between associated objects.

object type. DataHub grouping term for a stored collection of elements with the same definition. An object type (host, database, user, and so on) groups named occurrences of that type.

occurrence. See *object occurrence.*

offline monitoring. The DataHub monitoring capabilities that use watchdogs. DataHub watchdog checks monitor operating systems and RDBMSs without user intervention. Monitoring results can be filtered so only exception conditions are reported.

Contrast with *online monitoring*.

See also *watchdog check* and *watchdog rule*.

online monitoring. Direct, user-initiated monitoring operations started from the DataHub for UNIX Operating Systems window. Examples of online monitoring are displaying file system and CPU usage, disk I/O, and processes.

operating system driver. A required process supporting DataHub operations on UNIX operating systems. There is one operating system driver for each operating system managed from a DataHub for UNIX Operating Systems control point.

DataHub for UNIX Operating Systems provides operating system drivers for AIX, HP-UX, and Sun Solaris systems.

operating system watchdog. An unattended process that performs watchdog checks on a managed host at predefined times or time intervals.

ordinary identifier. In SQL, a letter, which might be followed by zero or more characters, each of which is a letter (a-z and A-Z), a symbol, a number, or the underscore character, used to form a name.

out-of-the-box monitoring. A subset of the DataHub predefined checks, with associated filter values and rules, that can be activated as a unit whenever a host, Sybase server or database is added or its definition modified. The filters represent typical exception conditions, and the rules offer advice to the DataHub administrator for dealing with the detected situation.

P

package. The control structure produced when the SQL statements in an application program are bound to a RDBMS. The RDBMS uses the control structure to process SQL statements encountered during statement execution.

predefined check. A watchdog check that is available for selection when you create a new check. Check commands and return types are provided in predefined checks. See also *watchdog check*.

primary key. A column or an ordered set of columns that contain non-null values and whose values uniquely identify a row. A table cannot have more than one primary key.

privilege. The right or authority to access a specific database object in a specific way.

prompt. A symbol or action that requests you to enter information.

pull-down menu. A list of choices extending from a selected menu choice that gives you access to actions, routings, and settings.

push button. A small, selectable rectangle with text inside used in program windows to start actions. Examples of push buttons are the **Display** push button and the **OK** push button.

Q

quiesce. To bring a device or system to a halt by rejecting new requests for work.

quoted name. An identifier enclosed in double quotes.

R

radio button. A diamond with text beside it. Radio buttons come in multiples to show a fixed set of choices from which only one can be selected. The diamond is partially filled when a choice is selected.

RDB. See *relational database.*

RDBMS. See *relational database management system.*

referential constraint. The relationship between two tables that controls the association between them, ensuring data consistency.

recovery. (1) The act of resetting a system, or data stored in a system, to an operable state following damage. (2) The process of rebuilding databases by restoring a backup copy and rolling forward the logs associated with it.

relational database (RDB). A database in which the data is organized and accessed according to relations.

relational database management system (RDBMS). A system that manages relational databases. DataHub for UNIX Operating Systems supports the following RDBMSs:

- DB2 Version 1
- DB2 Version 2
- INGRES
- ORACLE
- SYBASE SQL Server

restore. To return a backup copy to active storage for use.

roll-foward. To update the data in a restored database by applying changes recorded in the logs of transactions in the database.

root object. (1) In the UNIX operating system, the user name for the system user with the most authority. (2)An icon at the top of the object tree. The root icon is selected to display object types such as hosts, servers, and DataHub groups.

rule. See *watchdog rule*.

S

scroll bar. A part of a window, associated with a scrollable area, that a user interacts with to see information that is not currently visible. Scroll bars allow scrolling up and down, and left and right.

select. To identify one or more object types or occurrences to which a subsequent action should apply. An object can be selected by moving the cursor to the item and clicking mouse button 1.

selection field. An area of a window that contains a fixed number of choices from which the user can select. See also *entry field.*

selection list. A list of choices that a user can scroll through to select a choice.

self-join. To join a table or view with itself. See also *join*.

shell procedure. See *shell script*.

shell script. A series of commands combined in a file that carries out a particular function when the file is run or when the file is specified as a value to the **sh** command. Synonymous with shell procedure.

SNMP. Simple Network Management Protocol

Solaris. Sun Microsystems Solaris operating system for UNIX workstations.

SQL. See *Structured Query Language*.

Structured Query Language (SQL). A language that can be used within programs or interactively to define database objects, manipulate data, and control access to resources in a relational database environment.

stored procedure. A block of procedural constructs and embedded SQL statements that is stored in a database and can be called by name.

T

table. A named data object consisting of a specific number of columns and some unordered rows.

table column. A vertical component of a table that consists of a name, a particular data type, and, if necessary, a length. All values in a particular column share this same data type.

table space. In DB2 Version 2, a pageset that is used to store the records of one or more tables and, optionally, indexes.

tablespace. In ORACLE, a logical storage unit into which a database is divided.

TCP/IP. See *Transmission Control Protocol/Internet Protocol.*

threshold. A value that can be specified for a watchdog check. When the result of a watchdog check meets or exceeds this threshold value, the rule associated with this check is evaluated. See also *filter*.

token-ring address. A unique 12-digit number that identifies a system's location within a token-ring network.

toggle button. A small window object providing binary selection (such as on or off) for a process option in a window.

transaction log. A system table in which all changes to the database are recorded.

Transmission Control Protocol/Internet Protocol (TCP/IP). Part of the internet suite of protocols that allows users to communicate with each other in a consistent way regardless of the technology implemented in each node across the network.

trigger. Defines an action to be executed when a given update operation (DELETE, INSERT, or UPDATE) occurs for a specified table.

U

user directory. A directory in the UNIX operating system where user files and other directories are typically stored. The conventional names for these directories are: /u for AIX, /usr for HP-UX, and /opt for Sun Solaris.

user ID. A string of characters that identifies a user to a system (RDBMS or managed host system.)

user password set. An object in the object tree that represents a user ID and password for DataHub group objects. A user password set links the user ID and password for each group member to the user ID and password for a whole group.

W

WAN. See *wide area network.*

watchdog. A process providing *offline monitoring* capabilities. A watchdog is also referred to as an intelligent agent. Watchdogs are specific to run-time environments (operating systems and RDBMSs). For example: an AIX operating system is monitored by an operating system watchdog specific to AIX; an ORACLE database is monitored by an ORACLE watchdog. Each system and RDBMS to be monitored requires a watchdog. For example, if there are two occurrences of DB2 for AIX and one occurrence of an ORACLE RDBMS on a managed host, and all RDBMSs are to be monitored, you must have two DB2 for AIX watchdogs and one ORACLE watchdog running on that managed host.

Initially from the global rules evaluator, and then from the monitoring system on the managed host, watchdogs receive conditions to monitor (watchdog checks), filter values, watchdog check times, and instructions on whether or not to store check values. Results (watchdog values) are returned to the monitoring system for evaluation by watchdog rules.

watchdog check. An operating system command, SQL statement, or shell script run at predefined times. Watchdog checks are executed by watchdog processes. Any number of watchdog checks can be defined for each active watchdog. A watchdog check must return a single value that is an integer, floating point number, or string. Filtered values can be stored in the control database.

The values produced by executing watchdog checks are evaluated by watchdog rules.

See also *alert*, *filter*, *watchdog*, *watchdog rule*, and *watchdog server*.

watchdog rule. A condition or set of conditions for values returned by one or more watchdog checks. Watchdog rules define acceptable limits for the condition or conditions and specify an action to be taken if the conditions are met.

watchdog server. The watchdog server performs global rule evaluation. It sends watchdog checks, filter values, and watchdog check times to watchdogs. It receives watchdog check values from the watchdogs when rule evaluation requires input from two or more managed hosts. If the watchdog server evaluates a rule, it determines whether an action is required and what action to take. The action can include sending an alert or launching an executable.

wide area network. A network of devices that are connected to one another for communication over a large distance.

window. An area of the GUI with visible boundaries within which information is displayed. A window can be smaller than, or the same size as, the screen on the display device.

X

X client. Hardware-independent process that requests input/output from an X server.

X server. An application that controls the basic windowing mechanism used for input and output on X Window System displays.

X Window System. The window management and drawing management standard for UNIX operating systems. The X Window System consists of an X server, which manages a visual display, and client application programs.

Index

Special Characters

A

B

C

D

E

F

G

H

L

M

N

O

P

Q

R

S