Oracle® Database

2 Day DBA

10*g* Release 1 (10.1)

Part No. B10742-01

December 2003

ORACLE®

Oracle Database 2 Day DBA, 10*g* Release 1 (10.1)

Part No. B10742-01

Contributing Authors: Colin McGregor, Ruth Baylis, Sushil Kumar, Antonio Romero, David Austin, and Michele Cyran

Contributors: Daniela Hansell, Donna Keesling, Vasudha Krishnaswamy, Peter Laquerre, Tony Morales, Venkat Maddali, Mughees Minhas, Mark Townsend, and Wanli Yang

Contents

3 Getting Started with Oracle Enterprise Manager

4 Configuring the Network Environment

5 Managing the Oracle Instance

6 Managing Database Storage Structures

7   Administering Users and Security

8   Managing Schema Objects

10   Monitoring and Tuning the Database

Index

x

Send Us Your Comments

Oracle Database 2 Day DBA, 10*g* Release 1 (10.1)

Part No. B10742-01

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: infodev_us@oracle.com
- FAX: (650) 506-7227. Attn: Server Technologies Documentation Manager
- Postal service:

  Oracle Corporation
  Server Technologies Documentation Manager
  500 Oracle Parkway, Mailstop 4op11
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This preface contains these topics:

- Audience
- Organization
- Related Documentation
- Conventions
- Documentation Accessibility

Audience

Oracle Database 2 Day DBA is for anyone who wants to learn common day-to-day administrative tasks with the Oracle Database. Prior knowledge or experience with managing databases is not required. The only requirement is a basic knowledge of computers.

In particular this book is targeted toward the following groups of Oracle users:

- Developers wanting to acquire part-time DBA skills
- Anyone managing departmental servers
- Database administrators managing an Oracle database for small or medium business (SMB)

This book is equally useful for enterprise DBAs. It provides recommended best practices and describes efficient ways of performing administrative tasks with Oracle Enterprise Manager as the primary interface.

Organization

This document contains:

Chapter 1, "Introduction"

This chapter contains an brief overview of Oracle database administration.

Chapter 2, "Installing Oracle and Building the Database"

This chapter discusses how to install the Oracle software and database. It also describes how to configure additional databases, and how to upgrade and older version of a database to the current version.

Chapter 3, "Getting Started with Oracle Enterprise Manager"

This chapter introduces you to the Oracle Enterprise Manager and how to use it.

Chapter 4, "Configuring the Network Environment"

This chapter discusses how to configure a network so that clients can access your databases.

Chapter 5, "Managing the Oracle Instance"

This chapter discusses starting and stopping the database instance, and managing initialization parameters.

Chapter 6, "Managing Database Storage Structures"

This chapter discusses management of the database's storage structure.

Chapter 7, "Administering Users and Security"

This chapter discusses how to add and maintain user accounts. It includes information about using system privileges and roles to control user access to the database.

Chapter 8, "Managing Schema Objects"

This chapter discusses managing tables, indexes, and other schema objects.

Chapter 9, "Performing Backup and Recovery"

This chapter discusses how to backup and recover your database.

Chapter 10, "Monitoring and Tuning the Database"

This chapter discusses monitoring database activities and diagnosing performance problems.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Database Administrator's Guide*

- *Oracle Enterprise Manager Advanced Configuration*

- *Oracle Database Concepts*

- *Oracle Enterprise Manager Concepts*

- *Oracle Database SQL Reference*

- *Oracle Database Reference*

- *Oracle Database Backup and Recovery Basics*

- *Oracle Database Backup and Recovery Advanced User's Guide*

- *Oracle Database Installation Guide for Windows*

- *Oracle Database 10g Installation Guide, Release 1 (10.1) Beta for UNIX Systems: AIX-Based Systems, HP-UX, HP Tru64 UNIX, Linux, and the Solaris Operating System (SPARC)*

Many books in the documentation set use the sample schemas of the seed database, which is installed by default when you install Oracle. See *Oracle Database Sample Schemas* for information on how these schemas were created and how you can use them yourself.

Printed documentation is available for sale in the Oracle Store at

```
http://oraclestore.oracle.com/
```

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

```
http://otn.oracle.com/membership/
```

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

```
http://otn.oracle.com/documentation/
```

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.
`UPPERCASE monospace (fixed-width) font`	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a `NUMBER` column. You can back up the database by using the `BACKUP` command. Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view. Use the `DBMS_STATS.GENERATE_STATS` procedure.

Convention	Meaning	Example
`lowercase monospace (fixed-width) font`	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true`. Connect as `oe` user. The `JRepUtil` class implements these methods.
`lowercase italic monospace (fixed-width) font`	Lowercase italic monospace font represents placeholders or variables.	You can specify the `parallel_clause`. Run U`old_release`.SQL where `old_release` refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	`DECIMAL (digits [ , precision ])`
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	`{ENABLE	DISABLE}`
|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	`{ENABLE	DISABLE}` `[COMPRESS	NOCOMPRESS]`
...	Horizontal ellipsis points indicate either:	
	■ That we have omitted parts of the code that are not directly related to the example	`CREATE TABLE ... AS subquery;`
	■ That you can repeat a portion of the code	`SELECT col1, col2, ... , coln FROM employees;`

Convention	Meaning	Example
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	```SQL> SELECT NAME FROM V$DATAFILE; NAME ------------------------------------ /fsl/dbs/tbs_01.dbf /fs1/dbs/tbs_02.dbf . . . /fsl/dbs/tbs_09.dbf 9 rows selected.```
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	```acctbal NUMBER(11,2); acct CONSTANT NUMBER(4) := 3;```
Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	```CONNECT SYSTEM/system_password DB_NAME = database_name```
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	```SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;```
lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	```SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;```

Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Database Configuration Assistant, choose Start > Programs > Oracle - *HOME_NAME* > Configuration and Migration Tools > Database Configuration Assistant.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention.	```c:\winnt"\"system32 is the same as C:\WINNT\SYSTEM32```

Convention	Meaning	Example
`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp scott/tiger TABLES=emp` `QUERY=\"WHERE job='SALESMAN' and sal<1600\"` `C:\>imp SYSTEM/`*`password`*` FROMUSER=scott` `TABLES=(emp, dept)`
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`C:\> net start Oracle`*`HOME_NAME`*`TNSListener`
ORACLE_HOME and *ORACLE_ BASE*	In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level ORACLE_HOME directory. For Windows NT, the default location was `C:\orant`. This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level *ORACLE_HOME* directory. There is a top level directory called *ORACLE_BASE* that by default is `C:\oracle`. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora`*`nn`*, where *nn* is the latest release number. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle Database 10g Platform Guide for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Go to the *ORACLE_BASE\ORACLE_ HOME*`\rdbms\admin` directory.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over

time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

1

Introduction

As an Oracle Database Administrator (DBA), you are responsible for the overall operation of the Oracle database and maintaining its health.

This chapter contains the following topics:

- About this Book
- The Oracle Database
- Common Oracle DBA Tasks
- Tools for Administering the Database

About this Book

Oracle Database 2 Day DBA is a database administration quick start guide that teaches you how to perform common day-to-day database administrative tasks. The goal of this book is to give you adequate information to help you understand the concepts behind the Oracle Database. It teaches you how to perform all common administration tasks needed to keep the database operational, including how to perform basic troubleshooting and performance monitoring activities.

The primary administrative interface used in this book is Oracle Enterprise Manager in Database Console mode, featuring all the self-management capabilities introduced in the Oracle Database.

What this Book is Not

Oracle Database 2 Day DBA is task oriented. The objective is to describe why and when administrative tasks need to be performed. Where appropriate, it describes the concepts necessary for understanding and completing the task at hand, assuming the reader has no prior knowledge of the database. This book is not an exhaustive discussion of all Oracle database concepts. For this type of information, refer to *Oracle Database Concepts*

Additionally, for a complete discussion of administrative tasks, refer to *Oracle Database Administrator's Guide*

How to Use this Book with Related Material

This book is part of comprehensive set of learning material for administering an Oracle Database, which includes a *2 Day DBA Oracle by Example* (OBE) series, available on the web, and an Oracle University instructor led class.

Every chapter in *Oracle Database 2 Day DBA* has an associated Oracle By Example lesson. The OBE steps through all tasks in the chapter and includes annotated screen shots. In some cases, the OBE provides additional information to help you complete the task at hand.

At the end of each chapter, you can find the link to its associated OBE lesson. The home page for the *2 Day DBA Oracle by Example* series is as follows:

```
http://otn.oracle.com/obe/2day_dba/index.html
```

In addition, Oracle University provides a related instructor-led course called *Oracle Database 10g:Quick Start to Administration*. Contact Oracle University for more information.

The Oracle Database

Oracle is a relational database. In a relational database, all data is stored in two-dimensional tables that are composed of rows and columns. The Oracle Database enables you to store data, update it, and efficiently retrieve it.

Oracle provides software to create and manage the Oracle database. The database consists of physical and logical structures in which system, user, and control information is stored. The software that manages the database is called the Oracle database server. Collectively, the software that runs oracle and the physical database are called the Oracle database system.

You will learn more about the operation of the database server and the structure of the Oracle database where they are relevant to the performance of specific database management tasks.

Common Oracle DBA Tasks

As an Oracle DBA, you can expect to be involved in the following tasks:

- Installing Oracle software
- Creating Oracle databases
- Performing upgrades of the database and software to new release levels
- Starting up and shutting down the database
- Managing the database's storage structures
- Managing users and security
- Managing schema objects, such as tables, indexes, and views
- Making database backups and performing recovery when necessary
- Proactively monitoring the database's health and taking preventive or corrective action as required
- Monitoring and tuning performance

In a small to midsize database environment, you might be the sole person performing these tasks. In large, enterprise environments, the job is often divided among several DBAs, each with their own area of specialty, such as the database security administrator or database tuning expert.

Tools for Administering the Database

The intent of this book is to allow you to quickly and efficiently create an Oracle database, and to provide guidance in basic database administration.

The following are some of the products, tools, and utilities you can use in achieving your goals as a database administrator.

- **Oracle Universal Installer (OUI)**

  The Oracle Universal Installer installs your Oracle software and options. It can automatically launch the Database Configuration Assistant to install a database.

- **Database Configuration Assistant (DBCA)**

  The Database Configuration Assistant creates a database from templates that are Oracle supplied, or you can create your own. It enables you to copy a preconfigured seed database, thus saving the time and effort of customizing and generating a database from scratch.

- **Database Upgrade Assistant**

  This tools guides you through the upgrade of your existing database to a new Oracle release.

- **Oracle Net Manager**

  This tool guides you through your Oracle Net network configuration.

- **Oracle Enterprise Manager**

  The primary tool for managing your database is Oracle Enterprise Manager, a web-based interface. After you have installed the Oracle software, created or upgraded a database, and configured the network, you can use Oracle Enterprise Manager as the single interface for managing your database. In addition, Oracle Enterprise Manager also provides an interface for performance advisors and an interface for Oracle utilities such as SQL*Loader and Recovery Manager.

2

Installing Oracle and Building the Database

This chapter describes how to install Oracle software and a starter database with the Oracle Universal Installer (OUI). It also describes upgrading an older database to the current release.

This chapter contains the following topics:

- Overview of Installation
- Installing Your Oracle Software and Database
- Using DBCA to Create and Configure a Database
- Upgrading a Database
- Installation: Oracle by Example Series

Overview of Installation

To install your Oracle software you use the Oracle Universal Installer (OUI). The Oracle Universal Installer is a GUI tool that enables you to view the Oracle software that is already installed on your machine, install new Oracle software, and deinstall Oracle software that you no longer intend to use. Online help is available to guide you through the installation process.

Checking Prerequisites

Before the installation process, Oracle performs a number of automated prerequisite checks to ensure that your machine fulfills the basic hardware and software requirements for an Oracle install. If your machine does not meet any of the requirements, an error message is displayed. The requirements may vary somewhat depending upon the type of machine and operation system you are running, but some examples of prerequisite checks include:

- A minimum of 512 MB of memory is available.
- Sufficient paging space is available.
- Appropriate operating system service packs or patches are installed.
- Appropriate files system format is being used.

The installer automatically sets any operating system environment variables that the Oracle database server requires for its operation.

> **See Also:** Your Oracle operating-system specific documentation if you experience problems and need more information about exact requirements

Installation Choices

The Oracle Universal Installer guides you through an interview phase that asks you to specify your choices for installation and database creation. The exact sequence of pages depends on your platform. All platforms include the following choices:

- What database product do you want to install?

   You can install one of the following:

   - Oracle Enterprise Edition—Oracle's most full featured database product providing data management for high-end applications.

   - Oracle Standard Edition—A less-expensive and somewhat scaled down version of the Oracle database suitable for workgroup or department level applications

   - Custom Install—Allows you to customize the Enterprise Edition by selecting products to install. Do this if you want to add a component to your database that would not normally be installed or prevent certain components from being installed.

- Do you want to create a starter database in addition to installing the Oracle software? To create the database, the installer automatically launches the Database Configuration Assistant at the end of the installation.

   If you want a preconfigured database installed, you can select one of the following database types, optimized according to usage:

   - General Purpose

   - Transaction Processing

   - Data Warehouse

   If you want to create an custom database where you configure your own database structure, you can select an advanced install.

   > **Best Practice:**   Oracle recommends that you install a preconfigured database, which is faster and easier to do. You can customize the database to your liking later.

- What are your Database Configuration Options? These include your Global Database Name, Oracle System Identifier (SID), and whether to install the example schemas. The Global Database Name is the full name of the database which uniquely identifies it from any other database. The global database name is of the form *database_name.database_domain* as in *sales.us.acme.com*. The database name portion sales is a simple name you call your database. The database domain portion *us.acme.com* specifies the database domain in which the database is located. Together database name and domain make up the Global Database Name.

   Oracle recommends that you install the example schemas so that you can run many of the examples in the documentation.

   You will also be prompted to enter your database schema password for the pre-loaded schemas such as `SYS` and `SYSTEM`, which enable you to manage and administer the database.

- What kind of storage mechanism do you want your database to use?

   A database comprises several files which stores the user data, database meta data, as well as information required to recover from failures. As an administrator, you

decide what kind of storage sub-system to locate these files on.You can select from the following options:

- File System—Creates database files that are managed by your operating system's file system. You can specify the directory path where database files are to be stored. Oracle can create and manage the actual files.

- Automatic Storage Management—This is new functionality that enables you to define a pool of storage (called a disk group) in which Oracle automatically manages database file placement and naming. For environments with a large number of disks, this option simplifies database administration and maximizes performance. Disk mirroring and striping are done automatically. Automatic Storage Management requires a separate instance to configure and manage disks groups. You are guided through the creation and launching of an Automatic Storage Management instance and its components by the database installation process. For more information, see *Oracle Database Administrator's Guide*.

- Raw Devices—Enables you to manage your storage devices outside of the operating system file system by providing Oracle with unformatted physical disk space called raw devices. This method is primarily used in Oracle Real Application Clusters environments. If you selected this option, you must ensure that the raw devices have been created using an operating system command, and that they are not being used for any other purposes or by other Oracle databases.

If you are not certain of which option to use, go with the default selection of File System.

- What are your Database Management Options? Here you specify whether to manage your database centrally or locally. The central deployment mode enables you to manage multiple targets, such as databases, and application servers from a single console. The local deployment mode on the other hand can manage only a single database instance. The central deployment mode requires a special agent to be installed on the database machine. If no such agent is found, this option is disabled.

    If you are setting up a single database for the first time, it is recommended that you deploy Enterprise Manager locally. You can later install additional databases and migrate your deployment mode to central management.

At the end of the interview a summary page displays the options that you have chosen, space requirements, and the components that will be installed. If you have chosen to install a database, that will be done immediately after the software installation is complete.

Installing Your Oracle Software and Database

You are now ready to start the installation of your Oracle software. These steps provide a summary of the installation process. Online help is available to provide further direction in the choices that you make.

1. Log on to your computer as a member of the administrative group that is authorized to install Oracle software and create and run the database. Refer to your operating specific documentation or contact your system administrator find out if you have the necessary privileges to install new software.

2. Insert the distribution CD for the database into your CD drive. The Autorun window will appear automatically. Select **Install/Deinstall Products**.

If downloading from Oracle's download site, follow the instructions given on the Web site.

3. The Oracle Universal Installer Welcome page appears. Select **Next** to begin the installation of your software

4. **Linux and Unix Only**. If this is the first time that you are installing any Oracle software on this machine, the following occurs:

   ■ The Specify Inventory Directory page appears. You must specify a directory for installation files and the name of an operating system group that has write permission to the directory. OUI uses this location to keep track of all Oracle software. This information is used while applying patches to or upgrading and existing installation and while deinstalling Oracle software. Note that this area is different from the Oracle home, which you specify later. Select **Next** to continue.

   ■ A dialog page appears. You are asked to open a new terminal window, log in as root, and to run the `oraInst.sh` script. When the script finishes, return to the Oracle Universal Installer page and select **Continue**.

5. The Specify File Locations page appears. Enter the Oracle home name and directory path in which to install the Oracle software, or accept the default, then select **Next**.

   If there is already an Oracle home on your machine, Oracle suggests that home name and directory path for the install. If that home is for a different release of Oracle, it is recommended that you not use this home, but instead specify a new home name and path.

6. In the next two pages:

   ■ Specify to install the **Enterprise Edition** or **Standard Edition**. **Custom** is selected only for advanced environments.

   ■ Select one of the three preconfigured database types to be installed: **General Purpose**, **Transaction Processing**, or **Data Warehouse**. Do not select **Advanced** at this time.

   ■ Specify that you want to **Create a starter database**. If you select **Do not create a starter database**, then you will not create a database at this time and must use the Database Configuration Assistant to create a database at a later time.

7. You have given Oracle enough information for installing the software. Now more information is required for the database that is be installed. In the following series of pages you enter:

   ■ The database identification—You enter a global database name and SID (system identifier).

   ■ The database character set—Accept the default, unless you are aware of another character set that you should be using.

   ■ Your database management choice---You can manage your database and Enterprise Manager centrally using Oracle Enterprise Manager 10*g* Grid Control or locally using Oracle Enterprise Manager 10*g* Database Control. Select **Use Database Control for Database Management**. This is the recommended choice for a first time installation. You can use Grid Control if you plan to manage more than one database from Enterprise Manager. For more information, see "Installation Choices" on page 2-2

   ■ Your selection of database storage—This is discussed in "Installation Choices" on page 2-2.

- Your selection for backup and recovery options. You can enable automated backups into a recovery area, or optionally, later create your own backup strategy.

- Passwords for database administrative accounts, such as `SYS` or `SYSTEM`—These are the primary usernames for managing the database. The passwords that you assign should only be revealed to the highest level of administrators for your database.

You are almost done with the interview. A summary page displays the components that are about to be installed. Click **Install** to start the install. A progress bar will track the progress of the install.

8. A dialog page appears. You are asked to open a new terminal window, log in as root, and to run the `root.sh` script. When the script finishes, return to the Oracle Universal Installer page and select **Continue**.

9. At the end of the install of the software, the Configuration Tools page appears. Allow the tools to install and start. They configure your network, start an Oracle Net Services listener process for connecting to the database, and create the database and configure management tools. Click **Next**.

10. A page appears showing a progress bar for database creation. When database creation is done, an informational page appears that lists information about your database. Review this list. Click **Password Management** at the bottom of the page if you want to unlock or change passwords on database accounts.

11. The Password Management page appears. Locked accounts cannot be accessed. Remove the check from the **Lock Account** column for any accounts you might want to use, and assign new passwords for them. If you are not sure of these accounts, do nothing. These user accounts are explained to you in Chapter 7, "Administering Users and Security". You can unlock them later.

12. After returning from the Password Management page, a final End of Installation page appears with important information about Web application port numbers.

You have now completed the installation of Oracle software and have a running database. At this time, a web browser window will automatically be opened pointing to the URL of local Database Control. You will be prompted to enter your user name and password. You can enter username `SYS` and the associated password.

In Chapter 3, "Getting Started with Oracle Enterprise Manager", you become familiar with Enterprise Manager, a Web-based user interface for managing your database, and you are shown how to shut down your running database and start it up again.

Using DBCA to Create and Configure a Database

During installation, you can create and configure a new database. If you have already done so, you do not need to read this section and can skip to the next chapter.

However, if you choose to install software only and later create a database, or if you want to create additional databases using the release software that you just installed, you can do so by using the Database Configuration Assistant (DBCA). Other uses for DBCA are to delete a database, add options to a database, or to manage templates. A template is a definition of a database saved in an XML file format that can be used to create other databases.

Online help is available by clicking **Help**. It provides information that guides you in selecting configuration options.

Starting DBCA

To launch DBCA on the Windows operating system use the following procedure:

1. Log onto your computer as a member of the administrative group that is authorized to install Oracle software and create and run the database.

2. To launch DBCA on a Windows operating system select the following:

    ```
    Start > Programs > Oracle - home_name > Configuration and Migration Tools >
    Database Configuration Assistant
    ```

    To launch the DBCA on a UNIX, or as another method on a Windows operating system, enter the following at a command prompt:

    ```
    dbca
    ```

    which is typically found in `$ORACLE_HOME/bin`. The Welcome page appears.

3. Click **Next** to continue.

Using DBCA to Create a Database

Select **Create a Database** on the DBCA Operations page to begin a wizard that enables you to configure and create a database. During the interview, you are asked for your input on the following:

- Database Templates
- Database Identification
- Management Options
- Database Credentials
- Storage Options
- Database File Locations
- Recovery Configuration
- Database Content
- Initialization Parameters
- Database Storage
- Creation Options

Note that most pages of the wizard provide a default setting that you can accept. To accept all the default parameters, you can click **Finish** at any step.

Database Templates

This page enables you to select the type of database you want to create. By default, Oracle Corporation ships pre-defined templates. There are templates for Data Warehouse, General Purpose, and Transaction Processing databases. The templates contain settings optimized for workload. Click **Show Details** to see the configuration for each type of database. Choose the template suited to the type of workload your database will support. If you are not sure, select the default General Purpose template.

For more complex environments, you may want to select the **Custom Database** option. This will result in a more extensive interview and it will take longer to create your database, since a database creation script must be run.

For more information about using database templates, see "Managing Templates with DBCA" on page 2-10.

Database Identification

Enter the Global Database Name, in the form *database_name.domain_name* and SID (Oracle system identifier). The SID defaults to the database name and uniquely identifies the instance that runs the database.

Management Options

Use this page to set up your database so it can be managed with Oracle Enterprise Manager, which provides web-based management tools for individual databases, as well as central management tools for managing your entire Oracle environment.

To use Enterprise Manager, check **Configure the Database with Enterprise Manager.**

If the Oracle Management Agent has been installed on your host computer, then you are given the option of selecting central management by selecting **Use Grid Control for Database Management**. If you select this type of management, you must also indicate which management service to use in the drop-down menu.

Otherwise, select **Use Database Control for Database Management** to manage your database locally. If you choose this option, you can additionally **Enable Email Notifications**, for Oracle to email you alerts regarding potential problems, and **Enable Daily Backup.** Click **Help** for more information about these options.

Database Credentials

In this page, you specify the passwords for the administrative accounts such as SYS and SYSTEM. To use the same password for all accounts, specify **Use the Same Password for All Accounts** and enter the password.

Otherwise, specify **Use Different Passwords** and specify passwords individually.

Storage Options

Specify the type of storage mechanism you would like your database to use. For more information, refer to "Installation Choices" earlier in this chapter.

Database File Locations

In this page, you specify the Oracle home and directory path in which to install the Oracle software. Choose one of the following:

- **Use Database File Locations from Template**---Selecting this option instructs the DBCA to use the directory information as specified in the template. Even if you choose this option, you will have an opportunity later to make modifications to database filenames and locations.

- **Use Common Location for All Database Files**---This option requires you to specify a new directory for the Oracle home. All the database files will be created in this location. Even if you choose this option, you will have an opportunity later to make modifications to database filenames and locations.

- **Use Oracle-Managed Files**---Select this option to eliminate the need for you, the DBA, to directly manage operating system files comprising an Oracle database. You specify default location called a database area for all your files. Oracle thereafter automatically creates and deletes files in this location as required. You also have the option to create multiple copies of your redo and online log files by

selecting **Multiplex Redo Logs and Control Files**. To learn more about redo logs and control files, refer to Chapter 9, "Performing Backup and Recovery".

Selecting this option enables you to delegate the complete management of database files to the database. You no longer need to specify the filenames, location, or their sizes.

Recovery Configuration

When you create a new database, it is important to configure the database so you can recover your data in the event of a system failure. Use this page to specify a flash recovery area and to enable archiving.

To specify a backup and recovery area, choose Specify Flash Recovery Area and specify its directory location and size. You can use variables to identify some standard locations. To review or add additional locations, click **File Location Variables** at the bottom of the page.

By checking **Enable Archiving**, you can enable archiving. This includes archiving database redo logs, which can be used to recover a database. Selecting this option is the same as enabling Archive Log Mode in Oracle Enterprise Manager or running the database in ARCHIVELOG mode. You can accept the default archive mode settings or change them by selecting **Edit Archive Mode Parameters**

Oracle recommends you select **Enable Archiving**. Selecting this option provides better protection for your database in the case of software or hardware failure. If you do not select this option now, you can set up archive log mode later. See "Configuring Your Database for Basic Backup and Recovery" on page 9-4.

Database Content

In the Sample Schemas property page, check Sample Schemas if you want to include the Sample Schemas (EXAMPLE) tablespace in your database. The Sample Schemas provide a common platform for examples. Oracle books and educational materials contain examples based upon the Sample Schemas. Oracle recommends they be included in your database.

In the Custom Scripts property page, you can specify one or more SQL scripts to run after your database is created. This is useful for performing post-installation tasks, such as loading custom schemas. To specify scripts, check **Run the following scripts**. Note that if you choose to run scripts after installation, your scripts must include a connect string that identifies the database. Click **Help** for more information.

Otherwise to run no script, accept the default **No scripts to run.**

Initialization Parameters

The links on this page provide access to pages that enable you change default initialization parameter settings. These parameters fall into the following categories:

- Memory

- Sizing

- Character Sets

- Connection Mode

Memory Use this page to set the initialization parameters that control how the database manages its memory usage. You can choose from one of two approaches to memory management:

Typical---requires little configuration, and allocates memory as a percentage of total overall physical system memory. To use this method, select **Typical** and enter a percentage value. Click **Show Memory Distribution** to see how much memory the Assistant assigns to the System Global Area (SGA) and the Program Global Area (PGA). To learn more about PGA and SGA, refer to "Managing Memory Parameters" in Chapter 5, "Managing the Oracle Instance"

Custom---requires more configuration, but gives you more control over how the database uses system memory. This option is meant for more experienced database administrators. You can directly specify memory sizes for the SGA and PGA and their sub-structures, such as the shared pool and buffer cache.

To use this method, click **Custom**. To allocate specific amounts of memory to SGA and PGA, select **Automatic**. To customize how the SGA memory is distributed among the SGA memory substructures, select **Manual** and enter specific values for each SGA component. For a complete description of these memory areas, click **Help**.

Sizing In this property page, you specify the smallest block size and the maximum number of operating system user processes that can simultaneously connect to the database.

To specify block size, enter the size in bytes or accept the default. Oracle database data is stored in these blocks. One data block corresponds to a specific number of bytes of physical space on disk. While using pre-defined templates, this field is not enabled since the database will be created with the default block size of 8KB. But while using the custom option, you can change block size. Selecting a block size other than the default 8 KB value requires advanced knowledge and should only be done when absolutely required.

To specify maximum number of processes that can simultaneously connect to the database, enter the number or accept the default. The value should be 6 or greater. This value should allow for all background processes, such as locks and parallel execution processes. The default value for this parameter is 150 which is good enough for many environments.

Character Sets Use this page to define the character sets used by your database. Character sets are the encoding schemes used to display characters on your computer screen. Choosing a character set determines what languages can be represented in the database.

For *Database Character Set*, select from one of the following options:

- **Use the Default**--Select this option if you need to support only the language currently used by the operating system for all your database users and your database applications.

- **Use Unicode (AL32UTF8)**--Select this option if you need to support multiple languages for your database users and your database applications.

- **Choose from the list of character sets**--Select this option if you want the Oracle Database to use a character set other than the default character set used by the operating system.

The national character set is an alternate character set that enables you to store Unicode characters in a database that does not have a Unicode database character set. Another reason for choosing a national character set is that programming in the national character set might be easier. Select the **National Character Set** from the drop-down menu or accept the default.

The **Default Language** determines how the database supports locale-sensitive information such as day and month abbreviations, default sorting sequence for character data, and writing direction (left or right). Select the default language from the drop-down menu or accept the default.

The **Default Date Format** determines the convention for displaying the hour, day, month, and year. For example in the United Kingdom, the date format is DD-MM-YYYY. Select the format from the drop-down menu or accept the default.

Connection Mode Use this page to select the database mode. You can run the database in dedicated server mode or shared server mode.

In **Dedicated Server Mode**, there is a dedicated server process for each user process. Select this option when the number of total clients is expected to be small, or when database clients will make persistent, long-running requests to the database.

In **Shared Server Mode**, several client connections share a database-allocated pool of resources. Use this mode when a large number of users need to connect to the database. It is also useful when database memory is limited and for better performance, since you can have more client connections to the database than in dedicated server mode with the same memory. If you choose shared server mode, you must also indicate the number of server processes you want to create when an instance is started up. For more information about setting this parameter, click **Help**.

Database Storage

A tree listing shows you the storage structure of your database (control files, datafiles, redo log groups, and so forth). If you are not satisfied with the storage structure or parameters you can make changes. You can create a new object with **Create** and delete existing objects with **Delete**.

Note that if you selected one of the preconfigured templates for a database, you cannot add or remove control files, datafiles or rollback segments.

Creation Options

You have the option of creating your database at this time, or saving the database definition as a template to use at another time, or both.

Using DBCA to Delete a Database

To delete a database using DBCA, select **Delete a database** from the Operations page. When you select this option, DBCA deletes all the files associated with this database. On Windows, any associated services are also deleted.

Using DBCA to Configure Database Options

To change the database configuration, select **Configure database options in a database** from the Operations page. You can change your configuration from a dedicated server to a shared server. You can add database options that have not been previously configured for use with your database, for example: Oracle Label Security or Oracle OLAP.

Managing Templates with DBCA

DBCA templates are XML files that contain information required to create a database. Templates are used in DBCA to create new databases and make clones of existing databases. The information in templates includes database options, initialization

parameters, and storage attributes (for datafiles, tablespaces, control files and redo logs).

Templates can be used just like scripts, and they can be used in silent mode. But they are more powerful than scripts, because you have the option of cloning a database. This saves time in database creation, because copying an already created seed database's files to the correct locations takes less time than creating them as new.

Templates are stored in the following directory:

```
$ORACLE_HOME/assistants/dbca/templates
```

Advantages of Using Templates

The following are some of the advantages of using templates:

- They save you time. If you use a template you do not have to define the database.

- By creating a template containing your database settings, you can easily create a duplicate database without specifying parameters twice.

- They are easily edited. You can quickly change database options from the template settings.

- Templates are easy to share. They can be copied from one machine to another.

Types of Templates

There are two types of templates:

- Seed templates

- Non-seed templates

The characteristics of each are shown in the following table:

Type	File Extension	Include Datafiles?	Database Structure
Seed	.dbc	Yes	This type of template contains both the structure and the physical datafiles of an existing (seed) database. When you select a seed template, database creation is faster because the physical files and schema of the database have already been created. Your database starts as a copy of the seed database, rather than having to be built.
			You can change only the following:
			- Name of the database - Destination of the datafiles - Number of control files - Number of redo log groups - Initialization parameters
			Other changes can be made after database creation using custom scripts that can be invoked by DBCA, command line SQL statements, or the Oracle Enterprise Manager.
			The datafiles and redo logs for the seed database are stored in a compressed format in another file with a `.dfj` extension. The corresponding `.dfj` file's location is stored in the `.dbc` file.

Type	File Extension	Include Datafiles?	Database Structure
Non-seed	.dbt	No	This type of template is used to create a new database from scratch. It contains the characteristics of the database to be created. Non-seed templates are more flexible than their seed counterparts because all datafiles and redo logs are created to your specification (not copied), and names, sizes, and other attributes can be changed as required.

DBCA Templates Provided by Oracle

Oracle provides templates for the following environments:

Environment	Description of Environment
Data Warehouse	Users perform numerous, complex queries that process large volumes of data. Response time, accuracy, and availability are key issues.
	These queries (typically read-only) range from a simple fetch of a few records to numerous complex queries that sort thousands of records from many different tables.
Transaction Processing	Many concurrent users performing numerous transactions requiring rapid access to data. Availability, speed, concurrence, and recoverability are key issues.
	Transactions consist of reading (SELECT statements), writing (INSERT and UPDATE statements), and deleting (DELETE statements) data in database tables.
General Purpose	This template creates a database designed for general purpose usage. It combines features of both the DSS and OLTP database templates.
Custom Database	This template allows you maximum flexibility in defining a database.

Creating Templates Using DBCA

The Template Management page provides you with the option of creating or deleting a template.

For creating a template, you have the following three options:

- **From an existing template**

  Using an existing template, you can create a new template based on the pre-defined template settings. You can add or change any template settings such as initialization parameters, storage parameters, or use custom scripts.

- **From an existing database (structure only)**

  You can create a new template that contains structural information about an existing database, including database options, tablespaces, datafiles, and initialization parameters specified in the source database. User defined schema and their data *will not* be part of the created template. The source database can be either local or remote. You should choose this option when you want the new database to be structurally alike the source database and not contain the same data as contained by the source database.

- **From an existing database (structure as well as data--a seed database)**

You can create a new template that has both the structural information and physical datafiles of an existing database. Databases created using such a template are identical to the source database. User defined schema and their data *will* be part of the created template. The source database must be local. You should choose the option when you want to create an exact replica of the source database which is both structurally alike as well as contains the same data as that of the source database.

Oracle saves templates as XML files.

While creating templates from existing databases, you can optionally choose to translate file paths into OFA (Optimal Flexible Architecture) or maintain existing file paths. OFA is recommended if the machine on which you plan to create the database using the template has a different directory structure. Non-OFA can be used if the target machine has a similar directory structure.

Deleting DBCA Templates

The Template Management page also lets you delete existing templates. The DBCA displays a list of your current templates. You must select the template to delete. When you delete a template, it is no longer available for creating a new database or a new template.

Upgrading a Database

Perhaps your task is not to create a new database, but to upgrade an existing database. A database upgrade is necessary when you have an older database such as Oracle9*i* and need to upgrade it to the current release. Oracle provides a tool for upgrading an existing database, called the Database Upgrade Assistant (DBUA).

Overview of the Database Upgrade Assistant

The Database Upgrade Assistant (DBUA) interactively steps you through the upgrade process and configures the database for the new Oracle Database 10*g* release. The Database Upgrade Assistant automates the upgrade process by performing all of the tasks normally performed manually. The Database Upgrade Assistant makes appropriate recommendations for configuration options such as tablespaces and redo logs. You can then act on these recommendations.

For example, the Database Upgrade Assistant recommends sizing information for the new SYSAUX tablespace, which is required in Oracle Database 10*g*.

Before the Upgrade

The Database Upgrade Assistant performs the following pre-upgrade steps:

- It checks for any invalid user accounts or roles
- It checks for any invalid datatypes
- It checks for any desupported character sets
- It checks for adequate resources, including rollback segments, tablespaces, and free disk space
- It optionally backs up all necessary files

The Database Upgrade Assistant does not begin the upgrade until it completes all of the pre-upgrade steps.

During the Upgrade

The Database Upgrade Assistant automatically modifies or creates new required tablespaces, invokes the appropriate upgrade scripts, archives the redo logs, and disables archiving during the upgrade phase.

While the upgrade is running, the Database Upgrade Assistant shows the upgrade progress for each component. The Database Upgrade Assistant writes detailed trace and log files and produces a complete HTML report for later reference. To enhance security, the Database Upgrade Assistant automatically locks new user accounts in the upgraded database. The Database Upgrade Assistant then proceeds to create new configuration files (parameter and listener files) in the new Oracle home.

Real Application Clusters Support

The Database Upgrade Assistant is fully compliant with the Real Application Clusters (RAC) environment. In a RAC environment, the Database Upgrade Assistant upgrades all the database and configuration files on all nodes in the cluster.

Support for Silent Mode

The Database Upgrade Assistant supports a silent mode of operation where no user interface is presented to the user. Silent mode lets you use a single statement for the upgrade.

Oracle Database Version Restrictions

The Database Upgrade Assistant supports the following versions of the Oracle Database for upgrading to Oracle Database 10*g*:

- Oracle database version 8.0.6 (single instance databases only)

- Oracle8*i* Release 8.1.7 (single instance and Real Application Clusters databases)

- Oracle9*i* Release 1 (9.0.1) (single instance and Real Application Clusters databases)

- Oracle9*i* Release 9.2.0 (single instance and Real Application Cluster databases)

If your database version is not in this list, you need to upgrade first to the closest release as listed. You can then upgrade the database to the current release.

Starting the Database Upgrade Assistant

During installation, if you install software only and specify that you are upgrading your database to the new Oracle release, then the Database Upgrade Assistant is launched automatically at the end of the software install and you can continue as described in "Upgrading the Database Using the Database Upgrade Assistant" on page 2-15

If you perform a software-only install and do not upgrade your database at that time, then you can do so later by launching the Database Upgrade Assistant.

To launch the Database Upgrade Assistant on the Windows operating system choose:

```
Start > Programs > Oracle - home_name > Configuration and Migration Tools >
Database Migration Assistant
```

To launch the Database Upgrade Assistant on a UNIX or Linux operating system enter the following at a command prompt:

```
dbua
```

typically found in your `$ORACLE_HOME/bin` directory.

> **See Also:** *Oracle Database Upgrade Guide* for information about
> Database Upgrade Assistant command line options

When the Database Upgrade Assistant starts, the Welcome page appears.

Upgrading the Database Using the Database Upgrade Assistant

Complete the following steps to upgrade a database using the Database Upgrade
Assistant:

1. At the Welcome page of the Database Upgrade Assistant, make sure the database
   being upgraded meets the specified conditions. Then, click **Next**.

   If you need help at any page or want to consult more documentation about the
   Database Upgrade Assistant, then click **Help** to open the online help.

2. At the Select a Database page, select the database you want to upgrade. Then, click
   Next.

   You may need to provide a user name and password with SYSDBA privileges if
   you do not have operating system authentication.

3. At the Sysaux page, give the related attributes for creation of a SYSAUX tablespace.
   Then, click **Next**.

4. At the Recompiling Invalid Objects page, decide whether you want the Database
   Upgrade Assistant to recompile all invalid PL/SQL modules after the upgrade is
   complete. When you upgrade a database to the new Oracle Database 10*g* release,
   many of the PL/SQL modules in your database will become invalid. As a result,
   all existing PL/SQL modules in an INVALID state must be recompiled, such as
   packages, procedures, types, and so on.

   By default, the Oracle Database recompiles invalid PL/SQL modules as they are
   used. For example, if an invalid PL/SQL module is called, it will first be
   recompiled before it is actually executed. The time it takes to recompile the
   module can result in poor performance as you begin to use your newly upgraded
   database.

   To eliminate these performance issues, select **Recompile invalid objects at the end
   of upgrade**. When you select this option, the Database Upgrade Assistant
   recompiles all the invalid PL/SQL modules immediately after the upgrade is
   performed. This will ensure that you will not experience any performance issues
   later, as you begin using your newly upgraded database.

   > **Note:** Selecting Recompile invalid objects at the end of upgrade is
   > equivalent to running the *ORACLE_*
   > *HOME*/rdbms/admin/utlrp.sql script, which is used to
   > recompile stored PL/SQL and Java code and is described in the
   > *Oracle Database Upgrade Guide*.

   The task of recompiling all the invalid PL/SQL modules in your database can take
   a significant amount of time and increase the time it takes to complete your
   database upgrade. If you have multiple CPUs, then you can reduce the time it
   takes to perform this task by taking advantage of parallel processing on your
   available CPUs. If you have multiple CPUs available, then the Database Upgrade
   Assistant automatically adds an additional section to the Recompile Invalid
   Objects page and automatically determines the number of CPUs you have
   available.

The Database Upgrade Assistant also provides a recommended degree of parallelism, which determines how many parallel processes are used to recompile your invalid PL/SQL modules. Specifically, the Database Upgrade Assistant sets the degree of parallelism to one less than the number of CPUs you have available. For example, if you have three CPUs available for processing, then the Database Upgrade Assistant selects 2 from the Degree of Parallelism menu. You can adjust this default value by selecting a new value from the Degree of Parallelism menu.

5. At the Management Options page, you have the option of setting up your database so it can be managed with Oracle Enterprise Manager. Oracle Enterprise Manager provides Web-based management tools for managing individual database instances, as well as central management tools for managing your entire Oracle environment, including multiple databases, hosts, application servers, and other components of your network.

   a. When you run the Database Upgrade Assistant, the assistant checks to see if the Oracle Management Agent has been installed on the host computer. If the assistant locates an Oracle Management Agent, select the **Grid Control** option and select an Oracle Management Service from the drop-down list. When you finish installing the Oracle Database, the database will automatically be available as a managed target within the Oracle Enterprise Manager Grid Control.

   b. If you are not centrally managing your Oracle environment, you can still use Oracle Enterprise Manager to manage your database. When you install an Oracle Database, you automatically install the Oracle Enterprise Manager Database Control, which provides Web-based features for monitoring and administering the single-instance or clustered database you are installing.

   To configure the database so it can be managed with the Oracle Enterprise Manager Database Control, select the **Database Control** option.

   c. When you select the Database Control management option, you can configure Enterprise Manager so that email notifications will be enabled immediately upon installation.

   Select **Enable Email Notifications** if you want the SYSMAN user (the default Super Administrator and owner of the Management Repository schema) to receive email notification when a metric for a specified condition reaches a critical or warning threshold. For example, Enterprise Manager can send an email when a target goes down or when there are database space usage problems.

   d. If you select the **Database Control** management option, you can also enable automatic daily backups of your entire database.

   Select **Enable Daily Backups** to use the Oracle-suggested backup strategy to back up your entire database with a minimum amount of configuration. Later, you can use Oracle Enterprise Manager to customize your backup strategy further.

   When you select this option, Enterprise Manager will be configured to back up your database, based on the scheduled start time you enter on this page, immediately after you finish installing the Oracle Database. Enterprise Manager will back up the database to the Flash Recovery Area that you specify later on the Recovery Configuration page of the Database Upgrade Assistant.

   e. After you have made your choices, click **Next**.

6. At the Database Credentials page, secure your database with passwords for the Enterprise Manager accounts. You can set a single password, which will be applied to each of the listed Enterprise Manager user accounts, or enhance the security of the accounts by providing unique passwords for each user.

7. At the Recovery Configuration page, specify a flash recovery area and enable archiving. When you are managing your database, it is important to configure the database so you can recover your data in the event of a system failure.

   The Flash Recovery Area can be used to recover data that would otherwise be lost during a system failure; this location is also used by Enterprise Manager if you have enabled local management and daily backups on the Management Options page shown previously in the Database Upgrade Assistant.

8. At the Backup page, you have two options:

   - Choose **I have already backed up my database** if you completed a backup before running the Database Upgrade Assistant.

   - Choose **I would like this tool to back up the database** if you did not complete a backup. If you choose this option, then you can select the backup directory by clicking **Browse**.

   After you have made your choice, click **Next**.

9. At the Network Configuration for the database page, there are two property page options:

   - The **Listeners** property page is displayed if you have more than one listener in the new release's Oracle home. Select the listeners for which you would like to register the upgraded database.

   - The **Directory Service** property page shows up if you have directory service is configured in the new release's Oracle home. You can select to either register or not register the upgraded database with the directory service.

10. At the Summary page, make sure all of the specifications are correct. If anything is incorrect, then click **Back** until you can correct the specification. If everything is correct, then click **Finish**.

11. The Database Upgrade Assistant lists the initialization parameters that will be set for the database during the upgrade. Click **Upgrade**.

    A Progress dialog appears and the Database Upgrade Assistant begins to perform the upgrade.

12. During the upgrade, you may encounter error messages with **Ignore** and **Abort** choices. If other errors appear, than you must address them accordingly.

    - If **Ignore** is presented as a choice in the message, then clicking the button will ignore the current upgrade step.

      This causes the Database Upgrade Assistant to move on to the next step in the upgrade, ignoring this and any dependent steps. After the upgrade is complete, you can fix the problem, restart the Database Upgrade Assistant, and complete the ignored steps.

    - If **Ignore** is not presented as a choice in the message, then you need to terminate the process by clicking **Abort**.

      This will terminate the upgrade process. The Database Upgrade Assistant prompts you to restore the database if the database backup was taken by the Database Upgrade Assistant.

>After the database has been restored, you need to correct the cause of the error and restart the Database Upgrade Assistant to perform the upgrade again.
>
>If you do not want to restore the database, then the Database Upgrade Assistant leaves the database in its present state so that you can proceed with a manual upgrade.

13. After the upgrade has completed, the following message is displayed in the **Progress** dialog:

    ```
    Upgrade has been completed. Click the "OK" button to see the
    results of the upgrade.
    ```

    Click **OK**.

14. At the **Results** dialog, you can view the details of the upgrade. You can also unlock and set passwords to the user accounts.

    If you are not satisfied with the upgrade, then you can restore the database back to the previous release.

    If you are satisfied with the upgrade, then click **Done**.

15. At the Changes in Default Behavior page, the Database Upgrade Assistant displays some changes in behavior of Oracle Database 10*g* from that of previous releases. In some cases the default values of some initialization parameters have changed. In other cases some new behavior/requirement has been introduced that may affect current scripts or applications.

16. Perform the following actions, as necessary:

    a. Back up the database

    b. Change passwords for any new Oracle supplied administrative accounts.

    c. Migrate your traditional text initialization parameter file to a server parameter file, if you had not already done so in the previous release. This is not a required step, but using a server parameter file enables initialization parameter settings to be persistent across startup and shutdown and eliminates manual maintenance of initialization parameters. For more information, see*Oracle Database Upgrade Guide*.

    d. Adjust your parameter file for the new release.

Installation: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter, and includes annotated screen shots.

To view the Installation OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/install/install.htm
```

3

Getting Started with Oracle Enterprise Manager

This chapter introduces you to the Oracle Enterprise Manager Database Control, the web-based interface for managing an Oracle database. Understanding this tool is essential to managing the Oracle database as described in the rest of this book.

This chapter contains the following topics:

- Introduction to the Oracle Enterprise Manager Database Control
- Starting the Oracle Enterprise Manager Console
- Accessing the Oracle Enterprise Manager Database Control
- Granting Database Control Administrative Privileges
- Setting Up Enterprise Manager Preferences
- SQL Statements and Other Management Tools: SQL*Plus and iSQL*Plus
- Managing Your Database: A Roadmap
- Getting Started: Oracle by Example Series

Introduction to the Oracle Enterprise Manager Database Control

When you install an Oracle database, the Oracle Universal Installer also installs the new Oracle Enterprise Manager Database Control. The web-based Database Control serves as the primary tool for managing your Oracle database and sets a new standard in ease-of-use.

The following topics introduce you to the Oracle Enterprise Manager Database Control:

- Database Management Features
- Online Help
- Navigational Features

Database Management Features

From the Oracle Enterprise Manager Database Control, you can perform administrative tasks such as creating schema objects (tablespaces, tables, and indexes), managing user security, backing up and recovering your database, and importing and exporting data. You can also view performance and status information about your database instance.

Online Help

You can access context sensitive online help by clicking **Help** displayed on every page. In addition to viewing the help specific to the task at hand, there is a **View Contents** link that takes you to top level help where you can view a table of contents and link to topics. A search facility enables you to search the contents of help.

Navigational Features

Navigational features of Oracle Enterprise Manager include the following:
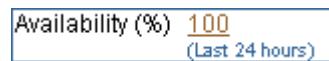
- Property pages on each home page help you organize management tasks into distinct categories; for example, click Maintenance to perform tasks associated with backing up and maintaining your data.

Figure 3–1   Database Home Page Navigational Property Pages

- Drill down links that provide increasing levels of detail.

Figure 3–2   Database Home Page Drill Down Link

Clicking the `100` link provides additional levels of detail about the database's availability.

- Suggested related links for more information.

Figure 3–3   Database Home Page Related Links

Starting the Oracle Enterprise Manager Console

To access the Oracle Enterprise Manager Console from a client browser, the dbconsole process needs to be running on the server. This process is automatically started after installation.

However, if the process is not running because of a system restart or otherwise, you can manually start it at the command line.

To start the dbconsole process:

1. Navigate into your $ORACLE_HOME/bin directory

2. Run the following statement:

   ```
   ./emctl start dbconsole
   ```

Additionally, you can stop the process and view its status.

To stop the dbconsole process:

```
./emctl stop dbconsole
```

To view the status of the dbconsole process:

```
./emctl status dbconsole
```

Starting dbconsole in Windows

In windows in addition to using the command line, you can start the dbconsole process as a service.

To start dbconsole as a service:

1. Click Start->Control Panel->Administrative Tools->Services. The Services page appears.

2. Oracle services begin with *Oracle*. The dbconsole service is listed as *Oracle<oracle_home><SID>DBConsole*. The status of this process is listed in the Status column, either Started or Stopped. Double click the service. The property page appears.

3. In the properties page, ensure that the Startup Type is either Manual or Automatic and not Disabled. Click **Start**, if the process is not already started. Click **OK**.

You can also use the Services page to Stop the process.

Accessing the Oracle Enterprise Manager Database Control

At the end of a successful database creation, Oracle Enterprise Manager is automatically launched and the database is started. At other times, provided the dbconsole process is running on the server, you can navigate to your Database Control from your client browser as follows:

1. Point your web browser to the following URL:

   http://*hostname:portnumber*/em

   For example, if you installed the database on a host computer named *comp42*, and the installer indicated that your Enterprise Manager Console HTTP port number is 5500 (also recorded in the $ORACLE_HOME/install/portlist.ini file), enter the following URL:

   ```
   http://comp42:5500/em
   ```

   If the database is up, Enterprise Manager displays the Database Control Login page.

   If the database is down and needs to be started, Enterprise Manager displays the Startup/Shutdown and Perform Recovery page. If this is the case, click the Startup/Shutdown button. You are then prompted for the host and target database login usernames and passwords, which you must enter. For the database user and password, use SYS and the password you specified during installation.

   Click **OK** to start the database. In the Confirmation screen, click **YES** to start the database in open mode.

2. Log in to the database using a username that is authorized to access the Database Control. This initially could be SYS or SYSTEM. Use the password you specified for the account during the database installation.

   Enterprise Manager displays the Database Home page (Figure 3–4).

The property pages across the top of the page enable you to access performance, administration, and maintenance pages for managing your database. The functionality provided by these pages is discussed in other chapters of this book.

The various sections of the Database Home page, and related links, provide a wealth of information about the database's environment and health. For example, the **Alerts**, **Related Alerts**, and **ADDM Analysis** sections warn you of errors and performance problems that are impacting the operation of your database. You can click the provided links to see more detail about the problem area, and even to obtain recommendations and SQL for resolving the problem. This is all discussed in Chapter 10, "Monitoring and Tuning the Database".

Figure 3–4   Database Home Page

Granting Database Control Administrative Privileges

When you log in to the Oracle Enterprise Manager Database Control using the `SYS` or `SYSTEM` user account, you are logging in as the Oracle Enterprise Manager super user. These are the only accounts that are automatically granted the roles and privileges required to access all the management functionality provided the Database Control.

To grant management access to other database users, use the following procedure:

1. Start your Web browser and log in to the Database Control as the `SYS` or `SYSTEM` database user.

2. Click **Setup** at the top of the Database Home page.

3. Click **Administrators** in the left navigation bar.

   If you need to create a new administrator, you must first create the user and then assign administrative privileges as described in this section. To create a new user, see "Administering Database Users" on page 7-7

4. Click **Create** to create a new Enterprise Manager user by assigning the management privileges to an existing database user.

5. Click the flashlight icon next to the Name field and select an existing database user from the pop-up window.

6. Enter the password for the selected user and click **Finish**.

Enterprise Manager assigns the management privileges to the selected user. The selected database user is now included in the list of management users on the Setup Administrators page. All the users shown on this page can log in to the Database Control and perform database management tasks.

Setting Up Enterprise Manager Preferences

Enterprise manager enables you to set up preferences that help you manage the database. These include the following:

- Notification---These are settings that enable Oracle to email you alert notifications. Alerts are notifications of when a database is in an undesirable state and needs your attention. By default, the Enterprise Manager home page lists all alerts. However, set up is required for email notification. For more information about alerts and setting up notifications, see "Setting up Notification" in Chapter 10, "Monitoring and Tuning the Database".

- Blackout Administration---When you have planned downtime for database maintenance, you can indicate to Oracle that you do not want false alerts to be sent to you. To do so, you define a black out period. See Defining Blackout Time Periods in this section.

- Preferred Credentials---Enterprise Manager can automatically execute many routine administrative tasks, such as backups on your behalf. This is done using sophisticated job scheduling system built into the Enterprise. To keep your environment secure, setting up tasks for automatic execution in Enterprise Manger requires you to provide login information for the machine and database. In order to avoid having to enter this information every time you create a job or task, Enterprise Manager enables you to save this information as preferred credentials. Preferred credentials are stored in the database in encrypted mode to protect it from unauthorized use. See Setting Preferred Credentials in this section.

Defining Blackout Time Periods

When you plan to bring your database down for maintenance, you can indicate that you do not want alert notifications to be sent to you. Alerts are notifications of when the database is in an undesirable state and needs your attention. For more information about alerts, see "Alerts" in Chapter 10, "Monitoring and Tuning the Database".

To define a blackout time period, follow these steps

1. From the home page, click **Setup** at the top of the page. The Setup page appears.

2. Click **Blackouts** in the left hand pane. The Blackouts page appears.

3. Click **Create**. This starts the Create Blackout wizard. On the Properties page, name your blackout, and select the target that has planned downtime. Click **Next**.

4. On the Schedule page, enter the start time of your planned blackout, or choose immediately if you are bringing the database down now. Also select the duration of the blackout, either as indefinite, as a length or time, or until a time in the future. Under Repeating, accept the default of Do Not Repeat. If you want to repeat the blackout periodically, you can select a repeat frequency in the pull down menu. Click **Next**.

5. The review page appears. Review what you have entered, and if satisfied, click **Finish**. If not, you can click Back to change a setting.

Setting Preferred Credentials

You can have Enterprise Manager automatically fill in host and database login credentials for you, such as when you schedule jobs and tasks to perform administrative operations like backup and recovery. For security, Oracle stores preferred credentials in encrypted mode.

To set preferred credentials for the database, do the following:

1. From the home page, click **Preferences** at the top of the page. The Preferences page appears.

2. Click **Preferred Credentials** in the left hand pane. The Preferred Credentials page appears. For the Database target, click the icon under Set Credentials. The Database Preferred Credentials page appears.

3. For your database, enter the credentials for Normal Username/Password, SYSDBA Username/Password, and Host Username/Password.

4. Click **Test** to test your credentials. You should get a confirmation message if your credentials can be verified.

5. Click **Apply** to apply the changes.

SQL Statements and Other Management Tools: SQL*Plus and *i*SQL*Plus

Enterprise Manager is a web-based graphical interface that enables you to perform management operations on the database like startup and shutdown, object creation, and schema management.

The underlying operations are performed using Structured Query Language (SQL) statements. SQL is an English-like computer programming language for accessing and manipulating data contained in a database. It is an industry-standard language for accessing databases, including those provided by vendors other than Oracle.

The following is an example of a SQL query:

```
SELECT COUNTRY_ID, COUNTRY_NAME FROM HR.COUNTRIES;
```

SQL is a powerful language and can be used to perform a variety of database administrative tasks. While performing these tasks, you can click Show SQL to see the underlying SQL statements being issued.

For more details on this language, see *Oracle Database SQL Reference*.

In addition to Enterprise Manager, you can use other Oracle tools to issue SQL statements such as SQL*Plus and its web version *i*SQL*Plus. These tools enable you to perform the same database management operations, as well as to query, insert, update or delete data directly in the database.

Starting SQL*Plus

SQL*Plus is a command line program you use to issue SQL statements to an Oracle database. You can issue these statements interactively or in batch. SQL*Plus comes with your Oracle installation and is located in your $ORACLE_HOME/bin directory.

To start SQL*Plus, enter the following at the command line on all platforms:

```
sqlplus username/password
```

You can use the SYS administrative user and password you set up during installation. SQL*Plus starts and connects you to the default database. At the SQL prompt, you can enter statements that perform administrative tasks such as database startup and shutdown. You can also query, insert, update and delete data.

On Windows, you can use the SQL*Plus Windows GUI. To start this, select Start->Programs->Oracle-OraHomeName->Application Development->SQL*Plus.

For more information about SQL*Plus, see *SQL*Plus User's Guide and Reference*.

For more information about SQL statements, see *Oracle Database SQL Reference*.

Starting *i*SQL*Plus

*i*SQL*Plus is the web version of SQL*Plus. Before you can connect to it with a URL, you must start the *i*SQL*Plus application server. To do so, enter the following at the command line:

```
isqlplusctl start
```

After the server is started, you can connect to *i*SQL*Plus with a URL such as

```
http://machine_name:5560/isqlplus
```

Alternatively, you can connect to *i*SQL*Plus using Enterprise Manger. To do so from the home page, click **iSQL*Plus** under Related Links.

For more information about *i*SQL*Plus, see *SQL*Plus User's Guide and Reference*.

Managing Your Database: A Roadmap

The following are some specific tasks to perform to become familiar with your database. They are presented as links to the chapters that contain information about these tasks and other related database management tasks.

1. Start up your instance and open the database. Initially, at the end of install, your instance is already started and your database is open. In the future there will be times, perhaps for doing database maintenance or because of a power or media failure, that you shut down your database or instance and later restart it. See "Shutting Down and Restarting the Instance and Database" on page 5-6.

2. Optionally configure the network environment to enable clients to connect to your database. See Chapter 4, "Configuring the Network Environment".

3. Review your database storage structure: tablespaces and datafiles, redo log files, and control files. See Chapter 6, "Managing Database Storage Structures".

4. Review, unlock, and reset passwords as necessary for initial database users. Create new users and assign privileges and roles to them. See Chapter 7, "Administering Users and Security".

5. Create users and necessary schema objects, including tablespaces and tables. Populate tables with data. See Chapter 8, "Managing Schema Objects".

6. Review a backup strategy for the database and back up the database. See Chapter 9, "Performing Backup and Recovery".

7. Enable archiving of redo logs, if not already done. See "Configuring Your Database for Basic Backup and Recovery" on page 9-4.

8. Monitor database performance, diagnose performance problems and tune the database as necessary. See Chapter 10, "Monitoring and Tuning the Database".

Getting Started: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter, and includes annotated screen shots.

To view the Getting Started OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/gettingstarted/gettingstarted.htm
```

4

Configuring the Network Environment

At the end of the install of your Oracle software and database, you have a fully functional database with a client/server network environment that has been minimally configured.

This chapter contains the following topics to help you completely configure your client/server network:

- Understanding Network Configuration
- Tools for Network Configuration
- Viewing Listener Configuration
- Starting and Shutting Down the Listener
- Configuring Oracle Networking on Client Machines
- Networking: Oracle by Example Series

Understanding Network Configuration

A client is any application that needs to connect to the Oracle database to send or retrieve data. An Oracle client application can reside on any machine provided it has Oracle client software installed.

Oracle Net is a software component that resides on the client and the Oracle database server. It is responsible for establishing and maintaining the connection between the client application and the server, as well as exchanging messages between them, using industry standard protocols.

For the client application and a database to communicate, the client application must specify location details for the database it wants to connect to and the database must provide some sort of identification, or address.

> **See Also:** *Oracle Net Services Administrator's Guide* for more guidance in understanding Net Services and for more configuration details.

Oracle Net Listener Configuration

On the database server side, the Oracle Net listener, commonly known as the listener, is a process that listens for client connection requests. It is configured in a file named `listener.ora`, with a protocol address that identifies the database. This address defines the protocol the listener is listening on and any other protocol specific information. For example, the listener could be configured to listen at the following protocol address:

```
(DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=my-server) (PORT=1521)))
```

This example shows a TCP/IP protocol address that specifies the host machine of the listener and a port number.

Client Configuration

When a client configured with the same protocol address broadcasts a request for a connection, the listener on the server machine brokers the request and forwards it to the Oracle database.

The client uses a connect descriptor to specify the database it wants to connect to. This connect descriptor contains a protocol address and a database service name. A database can have multiple services defined, so a specific one must be specified for the connection. In the case of the preconfigured database that you installed, there is only one service, the name of which defaults to the global database name.

The following example shows a connect descriptor that enables clients to connect to a database service called `mydb.us.acme.com`:

```
(DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=my-server) PORT=1521))
   (CONNECT_DATA=
      (SERVICE_NAME=mydb.us.acme.com)))
```

Connection Requests

Users initiate a connection request by providing a connect string. A connect string includes a username and password, along with a connect identifier. This connect identifier can be the connect descriptor itself, or a name that resolves to the connect descriptor. One of the most common connect identifiers is a net service name, a simple name for a service. The following example shows a connect string that uses net service name mydb as the connect identifier.

```
CONNECT scott/tiger@mydb
```

While the connect identifier is relatively simple as shown, the y can be long and inconvenient to use if your environment configuration is more complex. With long connect identifiers, you can use a mapping method that maps the net service name to the connect descriptor. This mapping information is stored in one or more repositories of information that are accessed with naming methods.

Naming Methods

Oracle Net provides support for the following mapping methods:

- Local Naming
- Directory Naming
- Easy Connect Naming
- External Naming

The preconfigured database uses local naming. Only this form of naming is used in this book.

Local Naming

The local naming method stores connect descriptors, identified by their net service name, in a local (on the client) configuration file named tnsnames.ora.

Directory Naming

The directory naming method stores connect identifiers in a centralized LDAP-compliant directory server to access a database service.

Easy Connect Naming

The easy connect naming method enables clients to connect to an Oracle database server by using a TCP/IP connect string consisting of a host name and optional port and service name:

```
CONNECT username/password@host[:port][/service_name]
```

For example:

```
CONNECT scott/tiger@my-server:1521/mydb
```

The easy connect naming method requires no configuration.

External Naming

The external naming method stores net service names in a supported non-Oracle naming service. These supported third-party services include:

- Network Information Service (NIS) External Naming

- Distributed Computing Environment (DCE) Cell Directory Services (CDS)

Tools for Network Configuration

Oracle enables you to manage your network configuration with the following tools: Oracle Net Configuration Assistant, Enterprise Manager, and Oracle Net Manager.

Oracle Net Configuration Assistant

The Oracle Universal Installer launches Oracle Net Configuration Assistant after the database is installed. Oracle Net Configuration Assistant enables you to configure the listening protocol address and service information for an Oracle database.

During a typical database install (as opposed to client install), Oracle Net Configuration Assistant automatically configures a listener with a name of LISTENER that has a TCP/IP listening protocol address for the Oracle database that you install. If you do a custom install, Oracle Net Configuration Assistant prompts you to configure a listener name and protocol address of your choice.

Oracle Net Configuration Assistant is primarily a post installation tool. Thereafter, Oracle provides other means for network configuration, which are the Oracle Enterprise Manager and Oracle Net Manager.

Enterprise Manager

Enterprise Manager enables you to manage your server-side network configuration with two pages: the Listener page and the Net Services Administration page.

The Listener page gives you listener status and enables you to shut it down. You can navigate to this page from the Home page by clicking the **Listener** link under General. See "Viewing Listener Configuration" in this chapter.

The Net Services Administration page enables you to do the following:

- Configure the following:
    - Local naming (`tnsnames.ora` files) on the machine running the database
    - Directory naming
    - Listeners
- Search and sort local and directory naming entries
- Export directory naming entries to `tnsnames.ora` file
- Perform the following administrative tasks for a selected listener:
    - Show current status
    - Change status (start/stop listener)
    - Change tracing settings
    - Change logging settings

You can navigate to the Net Services page from the Home page by clicking **Listener** under General. The Listener page appears. Under Related Links, click **Net Services Administration**.

Oracle Net Manager

You can access Oracle Net Manager from the command line or for Windows platforms through the Start menu.

- For command line (UNIX, Linux, or Windows) run `netmgr`.
- On Windows choose:

```
Start > Programs > Oracle - home-name > Configuration and Migration Tools >
Net Manager
```

Oracle Net Manager provides the same features as Oracle Enterprise Manager with the addition of configuring profiles. To learn how to use this tool, see "Configuring Oracle Networking on Client Machines" on page 4-5.

Viewing Listener Configuration

The listener runs on your database server machine and brokers incoming client requests. With Enterprise Manager, you can view the status of the listener, which is set up for automatic startup whenever your server machine is restarted.

To view the status of the listener, navigate to the Enterprise Manager home page. Under General, click **Listener**. The Listener: *listener_name* page appears. Under General, note the listener status, which is either Up or Down.

This page also gives the listener version, its net address, and start time.

You can additionally view the listener status at the command line using

```
lsnrctl status
```

Starting and Shutting Down the Listener

The Oracle listener is set up to start automatically whenever your server machine is restarted. However, when your system encounters unforeseen circumstances, or when you have manually stopped the listener, you can restart it at the command line. To do so, use the following:

```
lsnrctl start
```

You can use Enterprise Manager to stop the listener. To do so, navigate to the Listener: listener_name page by clicking **Listener** on the Home page. To shutdown the listener, click **Stop**.

You can also stop it at the command line using the following:

```
lsnrctl stop
```

Configuring Oracle Networking on Client Machines

Client machines need to be configured before they can connect to an Oracle database. To configure the client machine, you must first install Oracle client software, which includes Oracle Net software.

This example shows you how to configure a Windows client with local naming, which involves adding a connect descriptor entry to the local tnsnames.ora file. The connect descriptor specifies the host name, protocol address, and service name of the database to connect to.

On Windows, you can use the Oracle Net Manager program to create a new net service entry in your tnsnames.ora file as follows:

1. Start Oracle Net Manager by clicking Start->Programs->Oracle->Configuration and Migration Tools->Net Manager

2. Highlight Local Naming and click the plus (+) on the left hand side of the page. The Net Service Name Wizard appears.

3. Enter a net service name to identify it. The name you choose will appear under the Service Naming folder. Click **Next**.

4. Select the protocol to use to connect to your database. For example, you can accept the default TCP/IP (Internet Protocol). The database listener must be configured to use the same network protocol. Click **Next**.

5. Enter the host name of the database machine, such as `coconuts.island.com`.

6. Enter either the database service name or the database SID. In addition, select a Connection Type, either Shared or Dedicated. Shared connections require your database to be running in shared mode. If you choose Dedicated, the database dedicates a single server process to serve your connection. If you are unsure or would like to use the default database connection type, select **Database Default.** Click **Next**.

7. The last screen of the wizard enables you to test the connection using the information you entered. To do so, click **Test**. To log in to the database, you can use the `SYS` user and the password you specified when you configured your database. The wizard tells you if the connection test succeed or not.

8. Click **Finish**. The new service appears under the Service Naming folder. You can view or modify the configuration for the highlighted service under Service Identification and Address Configuration.

Note that you can additionally run Oracle Net Manager from the command line using `netmgr` on Windows, Linux, and Unix platforms.

Networking: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter and includes annotated screen shots.

To view the Networking OBE, point your browser to the following location:

`http://otn.oracle.com/obe/2day_dba/network/network.htm`

5

Managing the Oracle Instance

This chapter discusses the Oracle instance and database.

The following topics are covered:

- Overview of an Instance and Instance Management
- Shutting Down and Restarting the Instance and Database
- Viewing and Modifying Initialization Parameters
- Managing Memory Parameters
- Instances: Oracle by Example Series

Overview of an Instance and Instance Management

Your Oracle Database is comprised of a set of operating system files containing data entered by users or applications and structural information about the database itself called database metadata. Information is stored persistently in these files.

In order for you to view or update the data contained in the database, Oracle needs to start a set of processes, called background processes, and needs to allocate some memory to be used during database operation. The background processes and memory allocated by Oracle are together known as an instance. Therefore before the database can be used, the database instance must be started. When the database instance is not available, your data is safe in the database but it cannot be accessed by any user or application.

The properties of a database instance are specified using instance initialization parameters. When the instance is started, an initialization parameter file is read and the instance is configured accordingly.

The Oracle instance and the Oracle database are separate entities, although the term instance is often used to mean one or the other or both. Technically, they are distinguished as follows:

- An Oracle instance consists of the shared memory structures and background processes that run the Oracle database. You can have an instance without a database (for example, when you have not yet created a database), and if a database exists, it can be open or not.

- A database instance refers to the physical and logical components of a specific database, and its operation.

This section presents some of the concepts of an instance and its functioning. It is intended that these concepts can provide a basis for understanding instance management. The following topics are discussed:

- [Instance Memory Structure](#)
- [Oracle Background Processes](#)
- [Accessing the Database](#)
- [About Initialization Parameters](#)

Instance Memory Structure

This section briefly describes the memory structures of an instance. The size of these structures affects the performance of the Oracle database server and is controlled by initialization parameters. These initialization parameters can be categorized as memory parameters.

When a database is created with DBCA, the memory parameters are automatically set to optimal values based on your specification of the database workload. However, as your database usage expands, you might find it necessary to alter the settings of the memory parameters.

Oracle provides alerts and advisors to identify memory sizing problems and to help you determine appropriate values for memory parameters.

The System Global Area (SGA)

The SGA is a shared memory area that contains data and control information for the instance. Multiple users can share data within this memory area (controlled by Oracle) and information stored in the SGA can avoid repeated access from physical disk, a time consuming operation.

For optimal performance, the SGA should be large enough to avoid frequent disk reads and writes.

The SGA has several subcomponents as listed in the following table:

Component	Description
Buffer Cache	Before any data stored in the database can be queried or modified, it must be read from disk and stored in memory. The buffer cache is the component of the SGA that acts as the buffer to store any data being queried or modified. All user processes connected to the database share access to the buffer cache.
Shared Pool	The shared pool caches information that can be shared among users. Some examples: - SQL statements are cached so that they can be reused - Information from the data dictionary such as user account data, table and index descriptions, and privileges is cached for quick access and reusability - Stored procedures, which are executable code that is stored in the database, can be cached for faster access
Redo Log Buffer	This buffer improves performance by caching redo information (used for instance recovery) until it can be written at once and at a more opportune time to the physical redo log files that are stored on disk. Redo information and redo log files are discussed in "Redo Log Files" on page 6-2.
Large Pool	This is an optional area that is used for buffering large I/O requests for various server processes.
Java Pool	The Java pool memory is used for all session-specific Java code and data within the Java Virtual Machine (JVM)

Component	Description
Streams Pool	The Streams pool is used by the Oracle Streams product.

Program Global Area (PGA)

A program global area (PGA) is a memory area used by a single Oracle server process. A server process is a process that services a client's requests. Each server process has its own private PGA area that is a nonshared area of memory created by Oracle when a server process is started.

The PGA is used to process SQL statements and to hold logon and other session information.

The amount of PGA memory used and its content depends on the instance configuration, that is, whether the instance is running in dedicated server or shared server mode.

> **See Also:** "Managing Memory Parameters" on page 5-9

Oracle Background Processes

Oracle creates a set of background processes for an instance that interact with each other and with the operating system to manage memory structure, asynchronously perform I/O to write data to disk, and do general housekeeping. The background processes consolidate functions that would otherwise be handled by multiple Oracle programs running for each user process. They asynchronously perform I/O and monitor other Oracle processes to provide increased parallelism for better performance and reliability.

There are many background processes and not all may be present depending upon the features that are being used in the database. The most common background processes, and ones that most directly affect you, are the following:

Background Process	Description
Database Writer (DBWn)	The database writer writes modified blocks from the database buffer cache to the files on disk. Oracle allows a maximum of 20 database writer processes.
Log Writer (LGWR)	The log writer process writes redo log entries to disk. Redo log entries are generated in the redo log buffer of the SGA and the log writer process writes the redo log entries sequentially into an online redo log file.
Checkpoint	At specific times, all modified database buffers in the SGA are written to the datafiles by a database writer process (DBWn). This event is called a checkpoint. The checkpoint process is responsible for signalling DBWn at checkpoints and updating all of the datafiles and control files of the database to indicate the most recent checkpoint.
System Monitor (SMON)	The system monitor performs crash recovery when a failed instance starts up again.
Process Monitor (PMON)	The process monitor performs process recovery when a user process fails. It is responsible for cleaning up the cache and freeing resources that the failed process was using.
Archiver (ARCn)	When the database is running in archive log mode, one or more archiver processes copy the redo log files to archival storage when the log files are full or a log switch occurs. For more information, see Chapter 9, "Performing Backup and Recovery".

Accessing the Database

The database instance can be started (made available) only by an authorized database administrator who has a special type of connection privilege to the Oracle instance. After the database instance has been started, it is usually open for access by normal users for whom database user accounts have been created. Later, you will learn how to start up and shut down an Oracle instance and database in "Shutting Down and Restarting the Instance and Database" on page 5-6.

This section describes the process of starting the Oracle instance and database and the creation of server processes that enable user access.

How the Oracle Instance and Database are Started

The administrator who starts up the Oracle instance and database must connect to the instance with a special kind of connection privilege. There are two of these privileges: SYSDBA for fully empowered database administrators and SYSOPER for users who operate the database, but have no privileges to access user objects. Authorization to use these privileges occurs either through the operating system or through a special password file.

When you create an Oracle database, there are two primary administrative user accounts who are automatically created: SYS and SYSTEM. Both of these users have full database administration privileges granted to them, but only user SYS can initially connect with the SYSDBA privilege.

The Startup Process The process of starting the instance and database is as follows:

1. You start the instance in one of the following ways:

    ■   Use Oracle Enterprise Manager. See "Shutting Down and Restarting the Instance and Database".

    ■   Use the STARTUP statement using SQL*Plus. See *Oracle Database Administrator's Guide*

    ■   On Windows, you can start the Oracle services. See "Starting and Shutting Down the Database Instance on Windows".

    Oracle reads the initialization parameter file, allocates SGA memory, and starts the background processes for the instance.

2. If you also specify that the database is to be mounted, then the instance opens the database control file. The database is now said to be in the mount state. This state enables administrators to perform certain administrative functions that cannot be performed when other users are accessing the database.

3. If you specify that the database is to be opened, then the instance opens the redo log files and datafiles for the database. The database is now open and available for all user access.

    The default startup behavior is to complete the three stages as described earlier. Unless explicitly specified otherwise, the instance will be started, database mounted and opened.

The Shutdown Process The process is reversed when you shut down the database.

1. You issue the statement to shutdown the database by similar means as explained in "The Startup Process" on page 5-4. Any modified data blocks that are cached in the SGA and have not been written to disk are now written to disk. The redo log buffer is similarly flushed. Datafiles are checkpointed and their headers are

marked current as of the time the database was closed. Datafiles and log files are closed. Users can no longer access the database.

2. The Oracle instance dismounts the database and updates relevant entries in the control file to record a clean shutdown. The control file is closed. The database is now closed and only the instance remains.

3. The Oracle instance stops the background processes of the instance and deallocates the shared memory used by the SGA.

Server and Client Processes

In addition to background processes, Oracle creates server processes that handle the requests of user or client processes that connect to the instance. A user connection is composed of two distinct pieces:

- A client program acting on behalf of the user, such as Oracle Enterprise Manager, SQL*Plus, or an application

- A server process that handles the connection to the database on behalf of the client program

In some situations when the client and Oracle operate on the same machine, it is possible to combine the user process and corresponding server process into a single process to reduce system overhead. However, when the client and Oracle operate on different machines, a user process always communicates with Oracle through a separate server process.

Server processes can be either dedicated or shared. In dedicated server mode, each client process has its own server process. While a dedicated server process is good for long running queries and administrative tasks, an idle process or too many dedicated processes can result in an inefficient use of resources.

Shared server mode eliminates the need for a dedicated server process for each connection. A dispatcher directs multiple incoming network session requests to a pool of shared server processes. An idle shared server process from a shared pool of server processes picks up a request from a common queue, which means a small number of shared servers can perform the same amount of processing as many dedicated servers. Also, because the amount of memory required for each user is relatively small, less memory and process management are required, and more users can be supported.

Shared server mode is more efficient at supporting multiple users and clients making frequent short-running queries.

Network Connections

Oracle Net is a layer of software that allows different physical machines to communicate to accessing an Oracle database. A version of Oracle Net runs on the client machine and on the database server. The client machine and the database server are often the same machine.

Oracle Net enables the client and server to communicate over a network using many popular network protocols, and it provides location transparency such that the client machine does not need to know the server's location. When the database is moved to another location, you only need to reconfigure Oracle Net. No changes are necessary to client applications.

Oracle Net must be separately configured and started for it to handle client connections to the database. You can configure Oracle Net using Oracle Enterprise Manager or with a separately launched GUI tool called the Oracle Net Manager. To learn more, see Chapter 4, "Configuring the Network Environment".

When an instance starts, a listener process establishes a communication pathway to Oracle. When a user process makes a connection request, the listener determines whether it should use a shared server dispatcher process or a dedicated server process and establishes an appropriate connection.

About Initialization Parameters

Instance management involves configuring parameters that affect the basic operation of the database instance. These parameters are called initialization parameters. The Oracle database server reads these parameters at database startup and monitors them while the database is running. They are stored in memory, where many of them can be changed dynamically. There are two types of parameter files, and whether these dynamic changes are persistent across database shutdown and startup depends upon the type of parameter file you are using.

- Server parameter file

  This, the preferred form of initialization parameter file, is contained in a binary file that can be written to and read by the database server. It *must not* be edited manually. It resides on the machine that Oracle is running on, and is persistent across shutdown and startup.
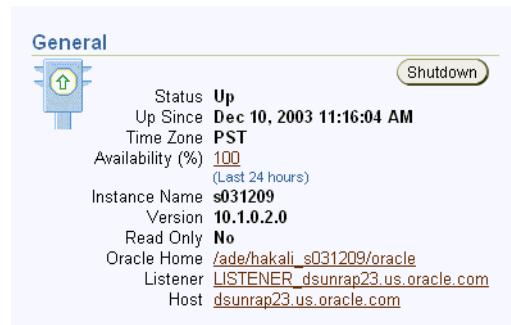
- Text initialization parameter file

  This type of initialization parameter file can be read by the database server, but it is not written to by the server. In this file, you can set initialization parameters with a text editor for them to be persistent across shutdown and startup.

Shutting Down and Restarting the Instance and Database

If you need to shut down and later restart your instance or database, you can do this with the Shutdown or Startup button, under the General heading on the Database Home page as shown in Figure 5–1, "Shutdown/Startup Button on Home Page".

Figure 5–1   Shutdown/Startup Button on Home Page

The first page to appear is Startup/Shutdown: Specify Host and Target Database Credentials. This page requires you to log in to the machine that Oracle is running on, and into the database itself with SYSDBA or SYSOPER privileges. The next screen enables you to shut down the database, or start up the database, specifying options.

Starting and Shutting Down the Database Instance on Windows

You can start and shut down your Oracle database with the Windows Services program. To start Oracle, you must start three services:

- OracleService<SID>, which is your Oracle Database instance

- Oracle<oracle_home><SID>TNSListener, which is your listener required to allow clients to connect to your database

- Oracle<oracle_home><SID>DBConsole, which enables clients to connect to Enterprise Manager

To start these services:

1. From the Control Panel, select Administrative Tools->Services. A list of all available services on you system appears.

2. Locate the three Oracle services listed earlier. For example, if your SID is *Sales1* and Oracle home is *OraHome10*, then locate the following services:

   - OracleServiceSales1

   - OracleOraHome10Sales1TNSListener

   - OracleOraHome10Sales1DBConsole

3. Start all three services. You can start a service by right-clicking the service, and selecting Start. You can also choose Start from the Actions menu. If you double click the service, the service properties page appears where you can Start or Stop the service, and select your Startup Type.

To shut down the database, you do the reverse: locate the services and stop them, by selecting Stop from the Actions menu.

Viewing and Modifying Initialization Parameters

When you install one of the preconfigured databases provided by Oracle, the initialization parameters are optimized for normal use in the environment that you specified. It is not necessary for you to alter any initialization parameters at this time. The steps described in this section will familiarize you with the initial parameter setting for your database and indicate how to modify parameters.

To view or modify the initialization parameters for your database:

1. From the Database Home page, click the **Administration** property page. The next page that you see is the Administration home page shown in Figure 5–2.

Figure 5–2   Database Administration Page

2.   Click **All Initialization Parameters** under the **Instance** heading.

Enterprise Manager displays the Initialization Parameters page comprised of a table listing the current value of each initialization parameter as seen by the database instance. You can also use this page to alter initialization parameter values, using the **Show SQL**, **Revert**, and **Apply** buttons.

There are two property pages shown on the Initialization Parameters page:

■   **Current**—The table on this property page displays all of the initialization parameter values currently seen by the database instance (in memory).

You can use this page to make dynamic changes to parameters in the current running instance. Only parameters marked dynamic can be changed. To do so, enter a new value and click **Apply**.

■   **SPFile**—This property page is present when you are using a server parameter file, whose location is displayed a the top of the table. This property page shows parameter settings in the server parameter file.

Changes to parameter settings in this file are persistent across instance startup and shutdown. To make persistent changes to an initialization parameter, enter the new value and click **Apply**. You can optionally apply changes to the current running instance by checking *Apply changes in SPFile to the current running instance*. If you do not check this box, your changes will not take effect until the database is shut down and restarted.

Parameters marked as **Basic** represent a small subset of all initialization parameters, and are considered necessary to keeping the database running and healthy.

As the number of database users grows and workload increases, you might need to alter some initialization parameters. You can make these changes directly using the Initialization Parameter page as described, or indirectly using one of the advisors provided by Oracle. One such advisor, the memory advisor, is described in "Managing Memory Parameters" on page 5-9.

Managing Memory Parameters

Some initialization parameters, referred to here as memory parameters, determine the total size of the system global area (SGA) and the program global area (PGA), and of the subcomponents of the SGA. The settings of memory parameters can affect the performance of your database. When you install your database, these parameters are tuned to meet the requirements of the environment that you specify.

If you enabled Automatic Shared Memory Configuration when you configured your database, Oracle automatically sizes the subcomponents of the SGA, which include the shared pool and buffer cache. Oracle recommends that you enable memory auto tuning.

If Automatic Shared Memory Management is disabled, you can enable it on the Memory Parameters page. You can navigate to this page from the Administration page by clicking *Memory Parameters* under the Instance heading. From this page you can enable or disable Automatic Shared Memory Management and view your SGA and PGA memory settings from their property pages.

Modifying Memory Parameters

To modify the size of an SGA subcomponent without bringing your instance down, you can disable Automatic Shared Memory Management on the Memory Parameters page. With the Advice button on the Memory Parameters page, the Memory Advisor (discussed in "Using the Memory Advisor" in Chapter 10, "Monitoring and Tuning the Database") gives you advice on optimal memory settings. If you choose to modify your memory settings manually, you can enter new sizes and apply changes dynamically while the instance is up. You can also do so from the Current property page on the Initialization Parameters page. However, there are some restrictions on dynamic modification of memory parameters. For more information, see *Oracle Database Administrator's Guide*.

To make changes to memory parameters persistent across instance startup and shutdown, you must alter your parameter file. You can do so by navigating to SPFile property page from the Memory Parameters page and making your changes there.

> **See Also:** *Oracle Database Performance Tuning Guide* for more information about memory parameters and their tuning

Instances: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter, and includes annotated screen shots.

To view the Instances OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/instance/instance.htm
```

6

Managing Database Storage Structures

This chapter discusses using Enterprise Manager to explore and manage the storage structures of your database. This chapter contains the following topics:

- Exploring the Storage Structure of Your Database
- Common DBA Tasks for Managing Database Storage
- Managing Undo for Your Database
- Making Changes to the Database Storage Structure
- Storage: Oracle by Example Series

Exploring the Storage Structure of Your Database

An Oracle database is comprised of physical and logical structures, which you, as a DBA, are responsible for managing. Physical structures are those that can be seen and operated upon from the operating system, such as the physical files that store data on disk.

Logical structures are created and recognized by the Oracle database server, and are not known to the operating system. The primary logical structure in a database, a tablespace, contains physical files. The applications developer or end user may be aware of the logical structure, but is not usually aware of this physical structure. The DBA must understand the correlation between the physical and logical structures of a database and control the mapping of the logical structure of the database to the physical structure.

Oracle has functionality within the database server to automate the management of its structure. Further, Oracle Enterprise Manager provide a Web-based graphical user interface to enable easy management and monitoring of your database.

You view database storage structure by clicking the links shown under the **Storage** heading on the Administration Home page (see Figure 5–2, "Database Administration Page"). Figure 6–1 shows these links.

Figure 6–1   Storage Options

Storage

Controlfiles
Tablespaces
Datafiles
Rollback Segments
Redo Log Groups
Archive Logs
Temporary Tablespace Groups

The following sections provide a closer look at the database storage structure:

- The Control File

- Redo Log Files

- Archive Log Files

- Rollback Segments

- Tablespaces

- Datafiles

- Other Storage Structures

The Control File

A control file tracks the physical components of the database and other control information. It is essential to the functioning of the database. Because of this, Oracle recommends that the control file be multiplexed. In other words, it should have multiple identical copies. For databases created using DBCA, three copies of the control file are automatically created.

If any control file fails, then your database becomes unavailable. But as long as you still have any of the control file copies intact, you can shut down your database and re-create the failed control file from one of the other control files, then restart your database. Another option is to delete the failed control file from the CONTROL_FILES initialization parameter and restart your database with the remaining control files. There are other options, depending upon circumstances. See *Oracle Database Administrator's Guide* for detailed information about control files.

Click the **Controlfiles** link. The Controlfiles General page shows that your database has a multiplexed control file. There are two other property pages, in addition to the General page, for viewing control file information. The Advanced and Record Section pages give you more detailed information about your control file. Explanations of this information are contained in the online help.

Redo Log Files

Every Oracle database has a set of two or more redo log files. The set of redo log files is collectively known as the redo log for the database. A redo log is made up of redo entries (also called redo records). The primary function of the redo log is to record all changes made to data. If a failure prevents modified data from being permanently written to the datafiles, then the changes can be obtained from the redo log, so work is never lost. To protect against a failure involving the redo log itself, Oracle allows a multiplexed redo log so that two or more copies of the redo log can be maintained on different disks.

Your database's redo log consists of groups of redo log files. A group consists of a redo log file, and its multiplexed copies. Each identical copy is said to be a member of that group, and each group is defined by a number, such as group 1. The database log writer process writes redo records from the memory buffer to a redo log group until the group fills up, or a log switch operation is requested, then it writes to the next group. This is done in a circular fashion, where the oldest group is overwritten by the most recent redo records.

Click **Redo Log Groups**. The Redo Log Groups page appears, showing the attributes of the redo log groups for your database. A database typically consists of three groups. Note the status attribute for the redo log groups. The CURRENT status group is the one currently being written to disk by the log writer process.

You can see that each group contains only one member. This is because the database does not have a multiplexed redo log. Select a redo log group and click **View** to see information about the redo log group members.

Multiplexing the Redo Log

While not multiplexing redo log groups is acceptable to a small database environment, as your database grows in users and applications, you should consider adding multiplexing. This provides better protection for data in the case of instance or media failure.

To multiplex your redo log, you must add members to each redo log group. This example will add one member to each redo log group. It is not required that redo log groups be symmetrical (all have the same number of members), but in practice, it is recommended that this be the case. It is required that a database have a minimum of two redo log groups.

To create a multiplexed redo log for your database, follow these steps:

1. Navigate to the Redo Log Groups page.

2. Select a group and click **Edit**, or click the group number link.

    The Edit Redo Log Group page appears.

3. In the **Redo Log Members** section, click **Add**.

    The Add Redo Log Member page appears.

4. Enter the file name. For example, if your existing member file name is REDO01, you might name this member REDO01a. Enter the file directory. Click **OK**.

    You can create this file in the same directory, but it is recommended that you store members on separate drives. That way, if there is a drive failure, you still have access to one member.

5. Repeat these steps for every existing group.

When you add the redo log member to a group, the group's status is marked INVALID. This is a normal state because there is a member that has not been written to yet. When a log switch occurs, and the invalid group becomes the current group, then its status is marked CURRENT.

To see the status change on a log switch:

1. Navigate to the Redo Log Groups page.

2. From the Actions pull down menu, select **Switch logfile**. Click **Go**.

    You can see that the next groups status changes from INVALID to CURRENT.

Archive Log Files

When you archive your redo log, you write redo log files to another medium location prior to their being overwritten. This location is called the archive log. You can archive to multiple locations, including a standby database.

These copies of redo log files extend the amount of redo information that can be saved and used for recovery. Archiving can be either enabled or disabled for the database, but Oracle recommends you enable archiving.The status of a log group that is actively being archived is `ACTIVE`.

Click **Archive Logs**. The Archive Logs page is displayed. This page lets you view archive log files and their attributes.

For placing the database into archive log mode, or adding archive log locations, see "Configuring Your Database for Basic Backup and Recovery" on page 9-4.

Rollback Segments

In earlier releases of the Oracle database, rollback segments were database structures used to track undo information for the database. Now, Oracle's preferred way of managing undo is with the undo tablespace. For more information, see "Managing Undo for Your Database" in this chapter.

Click **Rollback Segments**. You see the attributes of the system rollback segment. The system rollback segment is self managing.

Tablespaces

 A database consists of one or more tablespaces. A tablespace is a logical structure, or container, created by and known only to the Oracle database server in which database objects are stored. A tablespace consists of one or more datafiles or tempfiles.

There are various types of tablespaces, including undo tablespaces, temporary tablespaces, and permanent tablespaces.

A database running in automatic undo management mode transparently creates and manages undo segments. You can create an undo tablespace and determine the maximum retention time for undo data kept in that tablespace.

Temporary tablespaces improve the concurrence of multiple sort operations, reduce their overhead, or avoid Oracle space management operations altogether. Temporary tablespaces are the most efficient tablespaces for disk sorts. Space management (extent allocation and deallocation) is locally managed.

After temporary tablespaces, tablespaces of type `TEMPORARY` are the next best tablespaces to use for sort operations. Space management is dictionary managed. Permanent tablespaces (which are not of type `TEMPORARY`) are least efficient for performance of disk sorts.

Temporary and permanent tablespaces can be assigned at the user level. You can create a default temporary tablespace at database creation time. This tablespace is used as the default temporary tablespace for users who are not otherwise assigned a temporary tablespace.

> **See Also:** *Oracle Database Administrator's Guide*

Tablespaces Page

Click **Tablespaces**. The Tablespaces page provides high level information about tablespace structure and space usage. You can see how much space is allocated for the

tablespace and what portion of it is used. Select a tablespace and click **View** to drill down to see the underlying datafile attributes.

Some Tablespaces in the Database

Table 6–1 provides a description some of the tablespaces included in the database.

Table 6–1   Tablespaces and Descriptions

Tablespace	Description
SYSAUX	This is an auxiliary tablespace to the SYSTEM tablespace.
	Some components and products that prior to Oracle Database 10*g* used the SYSTEM tablespace or their own tablespaces now use the SYSAUX tablespace. This reduces the load on the SYSTEM tablespace and reduces maintenance because there are fewer tablespaces to monitor and maintain. Every Oracle Database 10*g* or higher level database must have a SYSAUX tablespace.
	Components that use this as their default tablespace during installation include Automatic Workload Repository, Oracle Streams, Oracle Text, and Enterprise Manager Repository. For more information, see *Oracle Database Administrator's Guide*
SYSTEM	The SYSTEM tablespace is always created at database creation. Oracle uses it to manage the database. It contains the data dictionary, which is the central set of tables and views used as a read-only reference describing a particular database. It also contains various tables and views that contain administrative information about the database. These are all contained in the SYS schema, and can only be accessed by user SYS, or other administrative users with the required privilege.
TEMP	This tablespace stores temporary data generated when processing SQL statements. It would, for example, be used for sort work space. Every database should have a temporary tablespace that is assigned to users as their temporary tablespace. In the preconfigured database, the TEMP tablespace is specified as the default temporary tablespace. This means that if no temporary tablespace is specified when the user account is created, then Oracle assigns this tablespace to the user.
UNDOTBS1	This is the undo tablespace used by the database server to store undo information. See "Managing Undo for Your Database" to understand Oracle's use of the undo tablespace. Every database must have an undo tablespace that can be created at database creation.
EXAMPLE	This tablespace contains the sample schemas that Oracle includes with the database. The sample schemas provide a common platform for examples. Oracle documentation and educational materials contain examples based on the sample schemas.

Table 6–1   (Cont.) Tablespaces and Descriptions

Tablespace	Description
USERS	This tablespace is used to store permanent user objects and data. Like the TEMP tablespace, every database should have a tablespace for permanent user data that is assigned to users. Otherwise, their objects will be created in the SYSTEM tablespace, and this is not good practice. In the preconfigured database, USERS is assigned the default tablespace, and space for all objects created by non-system users comes from this tablespace. For system users, the default permanent tablespace remains SYSTEM.

Temporary Tablespace Groups

Temporary tablespace groups lets users consume temporary space from multiple tablespaces. Using a tablespace group, rather than a single temporary tablespace, can alleviate problems caused when one tablespace is inadequate to hold the results of a sort, particularly on a table that has many partitions. A tablespace group enables parallel execution servers in a single parallel operation to use multiple temporary tablespaces.

A tablespace group name can appear where a tablespace name would appear when assigning a default temporary tablespace for the database or assigning a temporary tablespace for a user.

Click the **Temporary Tablespace Groups** link. You see the temporary tablespace groups that are defined for your database. Initially, there are none.

> **See Also:**   *Oracle Database Administrator's Guide* for details on using tablespace groups

Datafiles

Datafiles are the operating system files that hold the data within the database. The data is written to these files in an Oracle proprietary format that cannot be read by programs other than the database server. Tempfiles are a special class of datafiles that are associated only with temporary tablespaces.

Datafiles can be broken down into the following components:

- Segments and extents

  A segment contains a specific type of database object. For example, tables are stored in data segments, and indexes are stored in index segments.

  An extent is a contiguous set of data blocks (defined next) within a segment. Oracle initially allocates an extent of a specified size for a segment, but if that extent fills, then more extents can be allocated

- Data block

  Data blocks, also called database blocks, are the smallest unit of I/O to database storage. An extent consists of several contiguous data blocks. The default database block size is specified by a DBA at database creation or defaulted by Oracle.

  After the database has been created, it is not possible to change the default block size without re-creating the database. However, it is possible to create tablespace with a block size different than the default block size. For details on how to create

tablespaces with nondefault (or non-standard) block size, see *Oracle Database Administrator's Guide*.

Click **Datafiles**. Your database datafiles are displayed. Select a datafile and click **View** to see more attributes for the datafile, or select a tablespace link to see the attributes of the owning tablespace.

Other Storage Structures

Other storage structures that can exist in an Oracle database include the following:

Initialization Parameter File

Initialization parameters are used by the database server initially at startup to determine the runtime resources for the database. They are actively monitored by the database and can be set or modified while the database is running.

Initialization parameters and the initialization parameter file are discussed in Chapter 5, "Managing the Oracle Instance".

Password File

A database might use a password file to authenticate administrative users who can have SYSDBA connect privileges. SYSDBA connect privileges enable a DBA to startup and shutdown the database and perform other high level administrative tasks. This password file is outside of the database itself, because it must sometimes be referenced when the database is not yet running.

This is not the only form of administrator authentication, so not all databases require a password file.

The password file is discussed in "Accessing the Database" in Chapter 5, "Managing the Oracle Instance".

Backup Files

Backup files are technically not database files, but rather copies of the database in some form that can be used to recover the database should a failure causing loss of data occur.

Backup files are discussed in Chapter 9, "Performing Backup and Recovery".

Common DBA Tasks for Managing Database Storage

The remainder of this chapter presents examples of some common database administration tasks relating to the storage structure:

- Viewing Tablespaces in Your Database
- Creating a Tablespace
- Modifying a Tablespace
- Dropping a Tablespace
- Reclaiming Wasted Space

Viewing Tablespaces in Your Database

You can see all tablespaces in your database. From the Administration page, click **Tablespaces** under Storage. This takes you to the Tablespaces page.

The Results page shows all tablespaces, including the following information about each: type of tablespace, type of extent management, type of segment management, status, size, and percent used. Click the tablespace name for more information.

Creating a Tablespace

A particular user, groups of users, or an application may require its own tablespace for creating schema objects, rather than using the default USER tablespace. To create a tablespace, follow the steps listed in this section. This exercise will also familiarize you with the various attributes that you can set, and possibly later modify, for a tablespace.

1. From the Administration page, click **Tablespaces** under Storage. This takes you to the Tablespaces page.

2. Click **Create**. Or, if you want to create a tablespace that is like an existing tablespace, select an existing tablespace. Then from the **Actions** menu, select **Create Like**.

   The Create Tablespace General page appears.

   Did you notice the other options that are available from the Actions pull down menu? These are shortcuts for various tablespace operations.

3. Enter a name for the tablespace. For example: TB1

4. Under the **Extent Management** heading, select **Locally Managed**.

   A locally managed tablespace's extents are managed locally and efficiently within the tablespace by Oracle. For a dictionary managed tablespace, you must more actively manage extents and data dictionary access is required for tracking them. Dictionary managed tablespaces are being deprecated. Oracle does not recommend their use.

5. Under the **Type** heading, select appropriate type.

   Permanent tablespaces store permanent database objects created by the system or users.

   Temporary tablespaces are for storing temporary segments, as would be created when SQL statements perform sorts. The preconfigured database already has a temporary tablespace. You would create another temporary tablespace if you were creating a temporary tablespace group. Under normal circumstances, you should not need to create additional temporary tablespaces.

   Undo tablespaces are used by the system to store undo information. You might want to create another undo tablespace, but only one undo tablespace can be the active one.

   Again, you should not need to create additional undo tablespaces. However, if you want to switch the undo tablespace used by the database instance, then you can create a new undo tablespace and instruct the database to use it instead of the undo tablespace currently in use. The undo tablespace no longer in use can be dropped later.

6. Under **Status**, select **Read Write**.

   Read-write status means users can read and write to the tablespace after it is created. This is the default.

   If the tablespace was created read-only, then the tablespace could not be written to until its status is changed to read/write. You would probably not create a tablespace as read-only, but rather change it to that status after you have written data to it that you do not want modified.

If the tablespace was created offline, then no users could have access to it. You would probably not create a tablespace as offline, but rather you would change its status to offline when, for example, you wanted to perform some maintenance on its underlying files.

7. In the **Datafiles** section of the page, you can add one or more datafiles to the tablespace. However, if you select **Use bigfile tablespace**, then the tablespace can have only one datafile. Click **Add**.

    Bigfile tablespaces are used with ultralarge databases where Oracle's Automatic Storage Management or other logical volume managers support striping or RAID, and dynamically extensible logical volumes.

8. In the Add Datafiles page, enter a file name. For example: TS101.dbf. Use the File Directory and File Size defaults.

    At least one datafile is required for the SYSTEM and SYSAUX tablespaces of a database. Your database should contain several other tablespaces with their associated datafiles or tempfiles. The number of datafiles that you anticipate creating for your database can affect the settings of initialization parameters and the specification of CREATE DATABASE statement clauses. Be aware that your operating system might impose limits on the number of datafiles contained in your Oracle database. Also consider that the number of datafiles, and how and where they are allocated can affect the performance of your database.

9. To allow Oracle to automatically grow the datafile, under **Storage**, select **Automatically extend datafile when full (AUTOEXTEND)** and specify an amount in the **Increment** field by which you want to extend the datafile each time it fills. Leave the Maximum File Size set to Unlimited. Click **Continue**.

    You are returned to the Create Tablespace General page.

10. Click the Storage property page.

    The Create Tablespace Storage page appears.

    You can manually resize a datafile with the ALTER DATABASE statement. This lets you add more space to your database without adding more datafiles, if you are concerned about reaching the maximum number of datafiles allowed in your database. For a bigfile tablespace, use the ALTER TABLESPACE statement. You cannot add a datafile to a bigfile tablespace. Manually reducing the sizes of datafiles lets you reclaim unused space in the database. This is useful for correcting errors in estimates of space requirements.

11. Leave all of the defaults set on the Storage page. See online help for more information on these settings.

12. Click the Thresholds property page.

    The Thresholds page appears. This page lets you set monitored thresholds for space usage. You receive a warning when the threshold is reached. You can accept the default, specify different thresholds, or disable threshold warnings. See Chapter 10, "Monitoring and Tuning the Database" for more information about monitoring the database.

13. After determining thresholds, click **OK** to add the tablespace. The Tablespaces page appears where you can see your new tablespace in the **Results** section.

Modifying a Tablespace

With Enterprise Manager, you can modify a tablespace, such as extend it, set it to automatically extend, change its space thresholds, or change its status to offline. A

tablespace might need extending when you get a critical and warning alert. You might need to offline a tablespace to perform recovery operations on it.

This section shows you how to set a tablespace to automatically extend when it reaches its limit. To enable autoextend tablespace, do the following:

1. From the Administration page, click **Tablespaces** under Storage. This takes you to the Tablespaces page.

2. Select a tablespace to autoextend, such as USERS, and click **Edit**. The Edit Tablespace:USERS General property page appears.

3. Select the datafile associated with the tablespace, and click **Edit**. The Edit Datafile page appears.

4. Check **Automatically extend datafile when full (AUTOEXTEND)**. Set a suitable increment, such as 1 M. You can optionally set a maximum file size or set file size to unlimited depending on your resources.

    You can add datafiles and associate them with a tablespace with SQL statements. You can either specify the file specifications for the datafiles being created, or you can use the Oracle Managed Files feature to create files that are created and managed by the database server.

    You can also use SQL statements to manually resize a datafile. This lets you add space to your database without adding datafiles, if you are concerned about reaching the maximum number of datafiles allowed in your database. Manually reducing the sizes of datafiles lets you reclaim unused space in the database. This is useful for correcting errors in estimates of space requirements. For more information on managing datafiles, see *Oracle Database Administrator's Guide*.

5. Click **Continue**. This brings you back to the Edit Tablespace page. Click **Apply**. A confirmation message should appear.

Similarly, you can change tablespace thresholds by selecting a tablespace on the Tablespaces page and navigating to the Threshold property page. Here you can alter your space thresholds that generate the warning and critical alerts. For more information about thresholds and alerts, see Chapter 10, "Monitoring and Tuning the Database".

To offline a tablespace, on the Edit Tablespace page, under Status, select **Offline** and click **Apply**.

Dropping a Tablespace

After a tablespace has been dropped, the objects and data in it will no longer be available. To recover them can be a time consuming process. Oracle recommends a backup before and after dropping a tablespace.

1. From the Administration page, click **Tablespaces** under Storage. This takes you to the Tablespaces page.

2. Select the tablespace that you want to drop, and click **Delete**. Enterprise Manager will ask for confirmation that you want to delete the tablespace and associated datafiles from the operating system.

Reclaiming Wasted Space

Over time, DML operations to objects within a tablespace can create pockets of empty space that individually are not big enough to be reused. Collectively these pockets can combine to form large amounts of wasted space.

Oracle lets you reclaim wasted space through a shrink operation. In a shrink operation, data is compacted to the front of the table. Free space can either be given back to the tablespace or kept in the table for future inserts. A shrink operation on a table does not affect DML to the table during the operation.

Using the Segment Advisor

The Segment Advisor performs growth trend analysis on individual objects to determine if there will be any additional space left in the object in 7 days. It then uses the reclaim space target to select candidate objects to shrink. The Segment Advisor can be invoked in the comprehensive mode. In this mode, in addition to using the pre-computed statistics in the workload repository, the Segment Advisor performs sampling of the objects under consideration to refine the statistics for the objects. Although this operation is more resource intensive, it may be used to perform a more accurate analysis, when desired.

You can run the advisor at the tablespace or object level. To run the advisor from the tablespace level do the following:

1. From the Administration page, click **Tablespaces** under Storage. This takes you to the Tablespaces page.

2. On the Tablespaces page, select a tablespace and select **Run Segment Advisor** from the Actions pull-down menu. Note the used space before the operation for later comparison. Click **Go**. Segment Advisor can also be launched from the Advisor Central page.

3. The Segment Advisor: Advisor Mode page appears. On this page, accept the default **Complete Analysis of All Segments (Comprehensive)** mode. Click **Continue**.

4. The Segment Advisor:Options page appears. Accept **Unlimited** as the time limit for Analysis. Click **Next**.

5. The Segment Advisor:Schedule page appears. Accept **Immediately** as the Start. Otherwise, you can schedule the task to run later. Click **Next**.

6. The Segment Advisor:Review page appears summarizing the task and selected objects. Verify that the task will run as you want, and click **Submit**.

7. The Advisor Central page appears listing the advisor related tasks that have run. Select the Segment Advisor task that you submitted, and click **View Result**.

8. The Segment Advisor Task page appears showing the recommendations for your selected table.To implement one or more recommendations, select the recommendations to implement. This will launch the shrink wizard, which is described in the following section.

Shrinking Database Objects

Over time, it is common for segment space to become fragmented or for a segment to acquire a lot of free space as the result of update and delete operations. The resulting sparsely populated objects can suffer performance degradation during queries and DML operations.

The Segment Advisor provides advice on whether an object has space available for reclamation based on the level of space fragmentation within an object.

1. Select **Shrink Options** at the bottom of the page. You can choose to Compact Segments and Release Space, which releases freed space to the tablespace, or you can Compact Segments only. Click **Schedule Implementation** to run the shrink operation.

2. The Schedule Implementation page appears. Accept a start time of **Immediately**, and click **Submit**.

3. The scheduler Jobs page appears showing the status of the job. Click **Refresh** to update the page until the status of the job is Completed. On successful completion, you can navigate back to the Tablespace page and compare the current used space with the previous.

Managing Undo for Your Database

This section discusses undo management which involves storing the changes of database transactions long enough to accommodate rollback, read consistency, and flashback features.

When you install the database, Oracle automatically enables auto-tuning of the undo retention period. (This parameter is discussed in "Undo Tablespace Size and Retention Time" on page 6-12.) Auto-tuning of undo retention enables you to get started on building your database without immediately managing undo. Later as your database activity and transaction rate increase, understanding how to manage undo will become more useful to you.

About Undo

Undo data stores changes made to the database by transactions. Undo is necessary for the following:

- To undo any uncommitted changes made to the database in the event that a rollback operation is necessary. A rollback operation can be the result of a user specifically issuing a rollback statement to undo the changes of a misguided or unintentional transaction, or it can be part of a recovery operation.

- To provide read consistency. This means that each user can get a consistent view of data, even while other uncommitted changes may be occurring against the data. For example, if a user issues a query at 10:00 am and the query lasts for 15 minutes, the query results should reflect the entire state of the data at 10:00 am regardless of any updates or inserts by other users during the 15 minute query.

- To enable Oracle's flashback features, which enables you to view or recover data to a previous point in time. These features include Flashback Query and Flashback Table.

Undo Tablespace Size and Retention Time

In order for changes to be rolled back or undone, Oracle makes a copy of the original data before modifying it. The original copy of the modified data is called undo data. Undo data is stored in a logical database structure called an undo tablespace.

The undo tablespace is of a finite size. Space usage within the tablespace is such that records can be overwritten. The undo data needs to be saved at least until the transaction has been committed. Until then the undo data is said to be in the active state. The amount of space available in the undo tablespace should, therefore, be at least large enough to hold the active undo data generated by currently active transactions. Otherwise some of these transactions may fail. After the active undo data is stored in the undo tablespace, Oracle automatically ensures that it is never overwritten until the corresponding transaction has been committed.

Even after the transaction has been committed, the undo data still cannot be overwritten immediately. It is important to the success of flashback functionality, and

for read consistency for long running transactions, that records not be overwritten too soon. For example, if your longest query takes 15 minutes, the undo tablespace should be big enough to hold 15 minutes worth of undo information.

To control the retention of undo records, Oracle maintains an undo retention period, which in turn affects the size of the undo tablespace; the longer the retention period, the bigger the tablespace. The undo retention period should be at least as long as your longest running query. In addition, if you plan to use flashback features to recover from human errors such as unintentional changes, the undo retention should be set equal to the amount of time in the past you want to go back. For more details on Flashback features, see *Oracle Database Administrator's Guide*.

Automatic Undo Management

Oracle Database automatically determines how long undo data should be preserved after the transaction has been committed based on queries running against the database. During this time, the undo data is said to be in the unexpired state. Oracle will keep the undo data in the unexpired state as long as needed by queries currently running against the database or as long as any low undo retention threshold (whichever is longer). After this time, the state of undo data changes to expired. Undo data is a good candidate for overwriting only when it is in the expired state.

If the undo tablespace does not have adequate free or expired space to store active undo data generated by current transactions, Oracle may be forced to overwrite the unexpired undo data. As this may cause your queries to fail, this is not a desirable situation. In order to avoid this, Oracle recommends you allow the size of the undo tablespace to be automatically increased by enabling auto-extend. See "Extending the Undo Tablespace".

If you cannot enable auto extension for any reason and choose to size the undo tablespace manually, Oracle Database includes an undo advisor to help you determine the optimal size. See "Using the Undo Advisor" on page 6-14.

Undo Management with Enterprise Manager

With Enterprise Manager, you can access the Undo Management page from the database Administration page by selecting Undo Management under the Instance heading.

You can use the Undo Management page to view undo configuration, including the auto-tuned undo retention period, its low threshold retention period, and the name and size of the undo tablespace. Auto-tuning of undo retention cannot be disabled.

You can also use this page to set the low threshold retention or to extend your tablespace. While Oracle automatically tunes the undo retention period, low threshold undo retention lets you define the floor value under which Oracle should never bring down the undo retention. When you create a database, the low threshold undo retention is set to a default value. You may need to alter this value only when you need to build a recovery strategy using Flashback Query.

To change the low threshold, click its value. The Undo Advisor page appears, which can help you determine a better setting. See "Using the Undo Advisor" on page 6-14 for more information.

The Configuration section also shows if auto-extending the tablespace is enabled, which is the default. When auto-extend tablespace is enabled, Oracle automatically increases the size of the undo tablespace when more space is needed. By combining automatic extension of the undo tablespace with automatically tuned undo retention,

you can ensure that long-running queries will succeed by guaranteeing the undo required for such queries.

The methods for determining the undo retention period are described in the following sections:

- Using Oracle Recommendations for Managing Undo

- Using the Undo Advisor

> **See Also:** *Oracle Database Administrator's Guide* for more information about Automatic Undo Management and undo tablespaces.

Using Oracle Recommendations for Managing Undo

In the Recommendations section of the Undo Management page, you can specify a past time period for Oracle to analyze system activity. The current recommendations for your undo configuration are summarized. To update the analysis, click **Update Analysis**.

If Oracle recommends that you extend the undo tablespace, you can do so. See "Extending the Undo Tablespace" for more information.

To obtain more detailed information about the recommendations, click **Undo Advisor**. See also "Using the Undo Advisor" on page 6-14.

Extending the Undo Tablespace

When auto-extend tablespace is enabled, the system will automatically extend the undo tablespace if it is under space pressure.

However, when the auto-extend tablespace feature is disabled, you might need to manually extend the undo tablespace. You might discover this through pro-active planning with the undo advisor. See "Using the Undo Advisor" on page 6-14.

You might also need to extend it when you get an undo tablespace alert (warning or critical), or when you get long query alert or snapshot too old error.

To resize the tablespace, click **Edit Undo Tablespace**. The Edit Tablespace page appears. To extend the tablespace, select a datafile and click **Edit**. Enter the new size for File Size. You can also opt to have the system automatically extend the datafile by enabling **Automatically extend datafile when full** under Storage and specifying an increment size.

Click **Continue**. Click **Apply**. A confirmation message should appear.

Using the Undo Advisor

You can use the undo advisor to advise on the undo tablespace size and the low threshold undo retention setting. To navigate to the Undo Advisor page, do one of the following from the Undo Management page:

- Click **Undo Advisor**

- Click the value adjacent to **Low Threshold Undo Retention**.

The top of the page shows the current auto-tuned undo retention time and undo tablespace size.

The Undo Advisor helps you answer the following questions about your undo configuration:

1. **What is the maximum undo retention time required to guarantee successful query?** This time period corresponds to the duration of your longest running query. You can find **Longest Running Query** on the Undo Management page under System Activity and Tablespace Usage.

2. **Does my database have enough space to accommodate the maximum undo retention given the current tablespace configuration?** To figure this out, note the **Best Possible Undo Retention** under analysis on the Undo Advisor page. If this time period is longer than your longest running query, your undo tablespace is adequately configured.

3. **What if the best possible retention time is less than my longest running query?** If this is the case, your undo tablespace is too small. You need to either set your tablespace to auto-extend or manually extend it. To figure out the required new size of your tablespace, on the Undo Advisor page **New Undo Retention** field, plug in the value of your longest running query. Enter an appropriate Analysis Time Period and click **Update Analysis and Graph**. Under Analysis, note the Required Tablespace size for New Undo Retention. To learn how to alter your tablespace, "Extending the Undo Tablespace".

4. **How do I configure undo to accommodate Flashback operations?** If you are interested in using flashback features such as Flashback Query or Flashback Table, you can use the Undo Advisor to advise on how to configure your tablespace and undo retention. In order for flashback operations to go back in time, the database must ensure that undo data is not overwritten. To build a flashback recovery strategy, you can set the low threshold parameter, which determines the lowest value for automatic undo tuning. For example, if the low threshold is set to 15 minutes, Oracle never lowers the undo retention time to less than 15 minutes. Consequently, if your flashback recovery strategy requires you to go back 8 hours to recover from human errors, set the low threshold to 8 hours. To learn how to set a new retention time, see "Setting New Threshold Undo Retention Time".

The **Required Tablespace Size by Undo Retention Length** graph shows the relationship between retention period and undo tablespace size, highlighting key data points, such as the Auto-tuned and Best Possible Retention.

Setting New Threshold Undo Retention Time

To set a new low threshold retention time from the Undo Advisor page, enter the value in New Undo Retention and click **OK**.

Alternatively, you can set this parameter as you do for other system parameters by navigating to the All Initialization Parameters page. The parameter is called undo_retention. For more information about this page, see "Viewing and Modifying Initialization Parameters" in Chapter 5, "Managing the Oracle Instance".

Making Changes to the Database Storage Structure

The preconfigured database that you installed includes all of the database structures of a basic database. As the user base grows, you can expand existing database storage structures or create additional ones. For example, you might need to create additional tablespaces for users or applications, or you might want to create additional redo log groups to expand the redo log capacity.

Oracle provides alerts, advisories, and monitoring pages to help you make decisions regarding database storage. These are discussed in Chapter 10, "Monitoring and Tuning the Database". Oracle lets you create your own alerts for other events that you want to monitor.

The **Storage** heading links shown in "Storage Options" on page 6-2 are links that you can use to make changes to the database storage structure. Other pages also contain links, or drill down functionality, that enable you to perform specific actions that affect the database storage structure.

For example, there is a link for changing the archive log mode of the database and creating archive log files from the **Recovery** links on the database Maintenance page (see "Configuring Your Database for Basic Backup and Recovery" on page 9-4). Another example is changing the size of the undo tablespace, where you can use the Undo Advisor page to drill down to a page for changing the undo tablespace size (see "Managing Undo for Your Database" on page 6-12).

When you make changes to the storage structure of your database, these changes can be reflected in the data dictionary and in the control file. It is important both before and after making changes to the database's storage structure that you back up the database. To help you with this, Oracle has provided a backup strategy that you can use. This is discussed in Chapter 9, "Performing Backup and Recovery".

Storage: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter and includes annotated screen shots.

To view the Storage OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/storage/storage.htm
```

7

Administering Users and Security

For users to access your database, you must create user accounts and grant appropriate database access privileges to those accounts. Some user accounts are automatically included in the preconfigured database, but for security reasons, most of these accounts are locked and expired.

This chapter describes how to create and manage user accounts.

This chapter contains the following topics:

- Overview of Users and Security
- Administering Roles
- Administering Profiles
- Administering Database Users
- Users: Oracle by Example Series

Overview of Users and Security

Before users are allowed to connect to the database, they must have user accounts created for them. A user account is identified by a username and defines the user's attributes, including the following:

- How the user is to be authenticated
- Password (encrypted) for database authentication
- Tablespaces the user can access and quotas on them
- Whether the account is locked or not

These accounts are usually created by a database administrator or a person specifically assigned to create accounts and administer security.

After a user account is created, at least basic privileges must be granted to the account to enable the user to connect to the database and view and create database objects, known as schema objects.

Certain user accounts are automatically included when you create your database. All databases created by the Database Configuration Assistant include the SYS, SYSTEM, SYSMAN, and DBSNMP user accounts, but other accounts are included depending upon what features or options that are installed.

Most of the included accounts are administrative accounts, but user scott and Sample Schema accounts will also be present. User scott has long been used by Oracle in documentation to present examples that allow users to experiment with their

databases. However, the use of `scott` is gradually being phased out in favor of using the Sample Schemas.

The Oracle Database Sample Schemas are a set of interlinked schemas. This set of schemas provides a layered approach to complexity:

- A simple schema (Human Resources, HR) is useful for introducing basic topics. An extension to this schema supports Oracle Internet Directory demos.

- A second schema (Order Entry, OE) is useful for dealing with matters of intermediate complexity. Many datatypes are available in this schema, including nonscalar datatypes.

- The Online Catalog (OC) subschema is a collection of object-relational database objects built inside the OE schema.

- The Product Media schema (PM) is dedicated to multimedia datatypes.

- A set of schemas gathered under the main schema name IX (Information Exchange) can demonstrate Oracle Advanced Queuing capabilities.

- The Sales History schema (SH) is designed to allow for demos with large amounts of data. An extension to this schema provides support for advanced analytic processing.

The administrative accounts provided by Oracle should be used only by authorized individuals. To protect these accounts from being used by unauthorized persons familiar with their Oracle supplied passwords, these accounts are initially locked with their passwords expired. As the database administrator, you are responsible for the unlocking and resetting of these accounts as described in "Unlocking Accounts and Resetting Passwords" on page 7-10. Table 7–1 contains descriptions of some of these accounts.

Table 7–1 Administrative User Accounts Provided by Oracle

Username	Password	Description	See Also
CTXSYS	CTXSYS	The Oracle Text account	*Oracle Text Reference*
DBSNMP	DBSNMP	The account used by the Management Agent component of Oracle Enterprise Manager to monitor and manage the database	*Oracle Enterprise Manager Grid Control Installation and Basic Configuration*
MDDATA	MDDATA	The schema used by Oracle Spatial for storing Geocoder and router data	*Oracle Spatial User's Guide and Reference*
MDSYS	MDSYS	The Oracle Spatial and Oracle *inter*Media Locator administrator account	*Oracle Spatial User's Guide and Reference*
DMSYS	DMSYS	The data mining account. DMSYS performs data mining operations.	*Oracle Data Mining Administrator's Guide* *Oracle Data Mining Concepts*
OLAPSYS	MANAGER	The account used to create OLAP metadata structures. It owns the OLAP Catalog (CWMLite).	*Oracle OLAP Application Developer's Guide*

Table 7–1   (Cont.) Administrative User Accounts Provided by Oracle

Username	Password	Description	See Also
ORDPLUGINS	ORDPLUGINS	The Oracle *inter*Media user. Plugins supplied by Oracle and third party format plugins are installed in this schema.	*Oracle interMedia User's Guide*
ORDSYS	ORDSYS	The Oracle *inter*Media administrator account	*Oracle interMedia User's Guide*
OUTLN	OUTLN	The account that supports plan stability. Plan stability enables you to maintain the same execution plans for the same SQL statements. OUTLN acts as a role to centrally manage metadata associated with stored outlines.	*Oracle Database Performance Tuning Guide*
SI_INFORMTN_SCHEMA	SI_INFORMTN_SCHEMA	The account that stores the information views for the SQL/MM Still Image Standard	*Oracle interMedia User's Guide*
SYS	CHANGE_ON_INSTALL	The account used to perform database administration tasks	*Oracle Database Administrator's Guide*
SYSMAN	CHANGE_ON_INSTALL	The account used to perform Oracle Enterprise Manager database administration tasks. Note that SYS and SYSTEM can also perform these tasks.	*Oracle Enterprise Manager Grid Control Installation and Basic Configuration*
SYSTEM	MANAGER	Another account used to perform database administration tasks.	*Oracle Database Administrator's Guide*

> **See Also:**   *Oracle Database Sample Schemas* for a description of the Sample Schemas that are used for an example base in Oracle documentation and educational materials

Database Administrative Accounts

Two user accounts are automatically created when Oracle Database is installed:

- SYS

- SYSTEM

SYS

When you create an Oracle Database, the user SYS is automatically created and granted the DBA role.

All base tables and views for the database data dictionary are stored in the schema SYS. These base tables and views are critical for the operation of Oracle Database. To maintain the integrity of the data dictionary, tables in the SYS schema are manipulated only by the database. They should never be modified by any user or database administrator, and no one should create any tables in the schema of user SYS. (However, you can change the storage parameters of the data dictionary settings if necessary.)

Ensure that most database users are never able to connect to Oracle Database using the SYS account.

SYSTEM

When you create an Oracle Database, the user SYSTEM is also automatically created and granted the DBA role.

The SYSTEM username is used to create additional tables and views that display administrative information, and internal tables and views used by various Oracle Database options and tools. Never use the SYSTEM schema to store tables of interest to nonadministrative users.

A predefined DBA role is automatically created with every Oracle Database installation. This role contains most database system privileges. Therefore, the DBA role should be granted only to actual database administrators. The DBA role does not include the SYSDBA or SYSOPER system privileges.

Administrative Privileges

Administrative privileges that are required for an administrator to perform basic database operations, such as creating the database and instance startup and shutdown, are granted through two special system privileges, SYSDBA and SYSOPER. You must have one of these privileges granted to you, depending upon the level of authorization you require.

> **Note:** The SYSDBA and SYSOPER system privileges allow access to a database instance even when the database is not open. Control of these privileges is totally outside of the database itself.

The SYSDBA and SYSOPER privileges can also be thought of as types of connections that enable you to perform certain database operations for which privileges cannot be granted in any other fashion. For example, you if you have the SYSDBA privilege, you can connect to the database by specifying CONNECT AS SYSDBA.

> **See Also:** *Oracle Database Administrator's Guide* for more the operations authorized with each privilege and an example

Administering Roles

System privileges, object privileges, and roles provide a basic level of database security. They are designed to control user access to data and to limit SQL statements that users can execute.

Privileges and roles can be granted to other users by users who possess the privilege to do so. The granting of roles and privileges starts at the administrator level. At database creation, the administrative user SYS is created and granted all system privileges and predefined Oracle roles. User SYS can then grant privileges and roles to other users, and also grant those users the right to grant specific privileges to others.

The following are descriptions and examples of privileges and roles:

Privilege or Role	Description	Examples
System privilege	An Oracle-defined privilege usually granted only to and by administrators. System privileges allow users to perform specific database operations.	These are examples of system privileges that can be granted to users: CREATE TABLE: Allows grantee to create tables in grantee's schema. CREATE USER: Allows grantee to create users in the database. CREATE SESSION: Allows grantee to connect to an Oracle database to create a user session.
Object privilege	An object privilege controls access to a specific object.	These are examples of object privileges that can be granted to users: SELECT ON hr.employees TO myuser INSERT ON hr.employees TO myuser
Role	Roles are used to group privileges and other roles together, making them easier to manage as a group.	These are some examples of Oracle defined roles. You can create your own roles if you are granted the option to do so. CONNECT: When you create a user a shown in "Creating Users" on page 7-8, Enterprise Manager automatically grants this role to the new user. It grants several privileges to the user including CREATE SESSION and CREATE TABLE. RESOURCE: This role extends the privileges of a user beyond those granted by the CONNECT role. It includes CREATE PROCEDURE, CREATE TRIGGER, and other system privileges. DBA: This is the standard role that can be granted by an administrator to another DBA. It includes all system privileges, and should only be granted to the most trusted and qualified of users. Assigning this role to a user enables the user to administer the database.

See Also: *Oracle Database SQL Reference* for a list of Oracle-defined privileges and roles

Viewing Roles

You can use Enterprise Manager to view existing roles as follows:

1. From the Administration home page shown in Figure 5–2, click the **Roles** link shown under the security heading.

Security
Users
Roles ◄———
Profiles

2. The Roles page is displayed. From this page you can create, edit, view, or delete roles.

The structure and functionality of the Roles page is similar to that of the Users page shown in Figure 7–1.

3. Select the CONNECT role in the **Select** column. Click **View**. The View page appears and you can see all of the privileges and roles associated with the CONNECT role.

Creating a New Role

You can create a secure application role with all privileges necessary to run an application. You can then grant the role to other roles or users. A user can require several different roles, each granted a different set of privileges that allow for more or less data access while using the application.

To create a new role:

1. From the Administration Page, click **Roles** under the Security heading.

2. Click **Create**.

3. On the Create Role page, enter the name of your new role. Enter **APDEV** to create a new role for application developers in your organization. Click **OK**. A page appears with a list of all roles. You can now modify this new role by adding the required privileges.

Modifying Roles

You can add roles, privileges, and consumer groups to roles.

To modify the APDEV role:

1. From the Administration Page, click **Roles** under the Security heading.

2. Click **APDEV**. The Edit Role: APDEV page appears. This has links to the following property pages: Roles, System Privileges, Object Privileges, and Consumer Groups. Click **Roles**.

3. Click **Modify**.

4. In the Available Roles list, double click **CONNECT** to add the CONNECT role to the APDEV role. Click **OK**.

5. You are returned to the Edit Role: APDEV page. Click **System Privileges**.

6. Click **Modify**.

7. In the Available System Privileges list, double click **EXECUTE ANY PROCEDURE** and **SELECT ANY TABLE** to add these privileges to the APDEV role. Click **OK**.

8. You are returned to the Edit Role: APDEV page. Click **Object Privileges**.

9. From the Select Object Type pulldown list, select **Table**. Click **Add**.

10. From the Select Table Objects box, click the browse button (flashlight). Select the **HR** schema in the pulldown list. To see a list of all tables in the selected schema, clear the search box and click **Go**. Select the **JOBS** table. You are returned to the Add Table Object Privileges page.

11. Double click **DELETE** to add the DELETE privilege to the APDEV role for the HR.JOBS table. Click **OK**. To learn how to grant this role to a user, see "Granting Roles" in this chapter.

Dropping Roles

To drop the APDEV role:

1. From the Administration page, click **Roles** under the Security heading.

2. Select the APDEV role, and click **Delete**. At the confirmation page, click **Yes**.

Administering Profiles

A user profile establishes the password management policy for a user and defines and limits the user's access to certain database resources. When you create the user in "Creating Users" on page 7-8, you specify that the user be assigned the Oracle-supplied default profile. This default profile is quite liberal in its resource specifications and does not provide tight restrictions on password usage. Until you are more familiar with your database and its users, it is not necessary to be concerned about creating new profiles.

Follow these steps to view the attributes of the default profile:

1. From the Administration home page shown in Figure 5–2, click the **Profiles** link shown under the security heading.

Security
Users
Roles
Profiles ◄———

2. The Profiles page is displayed. From this page you can create, edit, view, or delete profiles.

   The structure and functionality of the Profiles page is similar to that of the Users page shown in Figure 7–1.

3. Select the DEFAULT profile in the **Select** column. Click **View**. The View page appears and you can see all of the attributes associated with the DEFAULT profile.

Database resource usage and limits are managed by the Database Resource Manager, which is not a subject of this book. You can read about the Database Resource manager in online help and view its pages when you click the links under the Resource Manager heading on the Database Administration page.

> **See Also:** *Oracle Database Administrator's Guide* for a description of the Database Resource Manager and its use

Administering Database Users

To create new users or administer existing ones select the Administration property page on the Database Home page (Figure 5–2).

From the Database Administration page, click **Users** shown under the Security heading.

Security
Users ◄———
Roles
Profiles

The Users page shown in Figure 7–1 is displayed. From this page you can view, add, edit, or delete database users.

Figure 7–1   Users Page

Creating Users

In this section, you create a user named `MYUSER`, set the password, and assign `MYUSER` to the `USERS` tablespace. This enables `MYUSER` to log in to the database

Later, you will learn how to edit and assign roles to this user.

To create a database user, do the following as user `SYS` or `SYSTEM`:

1. Click **Create** on the right hand side of the Users page. The Create Users General page shown in Figure 7–2 is displayed.

   The General page is one of a series of pages in the Users property page.

Figure 7–2 Create User General Page

2. Enter a user **Name**. For example: `MYUSER`

3. Leave Profile set to `DEFAULT`. The profile specifies the resource limit

   Note that this profile does not allow the user to log in to Enterprise Manager. To do so, the user needs to be assigned the DBA role. For more information, see "Administering Roles" on page 7-4.

4. Leave Authentication set to password. For advanced authentication schemes, see *Oracle Database Security Guide*.

5. Enter the password of your choice in the **Enter Password** and **Confirm Password** boxes.

6. Do not check **Expire Password** now. If the account status to set to expired, then the user or the database administrator must then change the password before the user can log in to the database.

7. For the **Default Tablespace** field, click the browse button (flashlight) and select the `USERS` tablespace. While users can specify different defaults for each user, it is easiest to define a default permanent tablespace and a default temporary tablespace at the database level, instead of the user level. For more information on the `USERS` tablespace, see "Some Tablespaces in the Database" on page 6-5.

8. For the **Temporary Tablespace** field, click the browse button (flashlight) and select the `TEMP` tablespace. For more information on the `TEMP` tablespace, see "Some Tablespaces in the Database" on page 6-5.

9. For **Status**, select **Unlocked**. For suspicious users who you do not necessarily want to delete, you can lock them out using this.

10. Click **OK**. The Users page is displayed, and you can now see an entry for the new user, `MYUSER`.

The Users General page has links to several other property pages. For example, you can specify user roles, privileges, quotas, consumer groups, and proxy users. Consumer groups are groups of users, or sessions, that are grouped together based on their processing needs.

> **Note:** This section creates a regular database user. If you want to grant the user system administrator privileges, you need to complete additional steps described in Chapter 3, "Getting Started with Oracle Enterprise Manager"

> **See Also:** *Oracle Database Administrator's Guide* and *Oracle Database Security Guide* for more information on roles, privileges, quotas, consumer groups and proxy authentication

Using a Shortcut to Create Users

If you later want to create other users similar to user MYUSER, or another user, Oracle provides a shortcut for doing so. On the Users page, you can create a user with the same attributes as another user as follows:

1. Select the user that you want to duplicate.

2. From the Actions menu, select **Create Like**.

3. Click **Go**. The Create User General page appears for a new user with the same attributes as the duplicated user.

The **Actions** box also provides shortcuts for other actions, as well as providing a means to display the SQL DDL used to create a user.

Editing Users

You can change the attributes of users by clicking **Edit** on the Users page (Figure 7–1) and selecting the properties page with the attributes that you want to change. The following procedure changes the tablespace quotas assigned to user MYUSER.

1. From the Users page, select MYUSER in the results list, then click **Edit**. The Edit User General page is displayed.

2. Click the Quotas property page to display the Quotas page.

   You can see that this user has 0 Mbytes quota on all tablespaces. This means that he has no quota in any tablespace. Since MYUSER belongs to the users tablespace, he must have quota to create tables and other schema objects in this tablespace.

3. Pull down the box in the Quota column for tablespace users. Select **Value**.

4. In the corresponding **Value** column, enter 100.

5. Click **Apply**.

   User MYUSER can now create objects in the users tablespace.

6. Click the **Database** property page to return to the Administration home page.

Unlocking Accounts and Resetting Passwords

Locked accounts cannot be accessed. During installation, you were provided the opportunity to unlock and reset the Oracle supplied database user accounts. If you did not choose to unlock those accounts at that time, you can do so now as follows:

1. From the Users page shown in Figure 7–1, in the **Select** column of the table, mark a user whose **Account Status** is shown as EXPIRED AND LOCKED and that you want to unlock and reset.

2. From the pull down menu in the **Actions** pull-down menu, select **Unlock User** and click **Go**. Click **Yes** to the confirmation message.

   This unlocks the user account (the **Account Status** is now `EXPIRED`), but does not reset the password. The user would still be unable to log in to the database.

3. From the Users page, again mark the user in the **Select** column, then click **Edit**.

   The Edit User General page is displayed. You can also display this page for a user by clicking the username in the **UserName** column.

4. Use the General page to change the password for the account. Note that you can also use this page to unlock the account.

You must follow the preceding steps individually for each account that you want to unlock and reset.

Locking and account is similar. You select the user and then choose Lock User from the Actions pull-down menu. Locking an account denies access to the account.

To better understand the ramifications of unlocking and resetting accounts, see the other property pages available on the Edit user page. Specifically, if you explore the **Roles**, **System Privileges**, and **Object Privileges** property pages, you can see the privileges of the user whose account you are enabling.

Granting Roles

As a system administrator, you can use Enterprise Manager to grant roles to users. For example, you can grant a user the `DBA` role, which allows the user to administer the database. You can also grant user-created roles, such as the APDEV role you created in "Creating a New Role".

To grant the `DBA` role to `MYUSER`:

1. From the Administration Page, click **Users** under the Security heading.

2. From the list of users, select `MYUSER` and click **Edit**. The Edit User: `MYUSER` page appears.

3. In the top navigation bar, click **Roles**. A page appears with the list of roles for `MYUSER`.

4. Click **Modify** on the right hand side of the page. The Modify Roles page appears.

5. In the Available Roles list, select the `DBA` role and click **Move** to make it part of the Selected Roles list. You can also select the APDEV role you created earlier in this chapter.

6. Click **OK**. Then click **Apply**.

Revoking Roles

You can also revoke roles from users. To revoke the `DBA` role from `MYUSER`:

1. From the Administration page, click **Users** under the Security heading.

2. From the list of users, select `MYUSER` and click **Edit**. The Edit User: `MYUSER` page appears.

3. In the top navigation bar, click **Roles**. A page appears with the list of roles for `MYUSER`.

4. Click **Modify** on the right hand side of the page. The Modify Roles page appears.

5. In the Available Roles list, select the `DBA` role and click **Remove** to make it part of the Available Roles list.

6. Click **OK**. Then click **Apply**.

Dropping Users

Enterprise Manager enables you to drop users. However, you must excerise caution when doing so, since dropping user will drop all schema objects owned by the user including tables and indexes. If you want to deny user access to the database, it is better to lock the user account or expire the user password. See "Unlocking Accounts and Resetting Passwords".

To drop a user such as MYUSER:

1. From the Administration page, click **Users** under the Security heading.

2. Select the **MYUSER** user, and click **Delete**. At the confirmation page, click **Yes**.

Users: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter and includes annotated screen shots.

To view the Users OBE, point your browser to the following location:

`http://otn.oracle.com/obe/2day_dba/users/users.htm`

8

Managing Schema Objects

This chapter discusses the management of schema objects. You are shown how to explore the schema objects in the preconfigured database. The chapter contains the following sections:

- Overview of Schemas and Common Schema Objects
- Managing Tables
- Managing Indexes
- Managing Views
- Managing Database Resident Program Units
- Other Schema Objects
- Loading Data Into Tables
- Managing Access to Schema Objects
- Some Schemas of Interest
- Schemas: Oracle by Example Series

Overview of Schemas and Common Schema Objects

A schema is a collection of database objects. A schema is owned by a database user and has the same name as that user. Schema objects are logical structures created by users. Objects may define areas of the database to hold data, such as tables or indexes, or may consist just of a definition, such as a views or synonyms.

> **Note:** There is no relationship between a tablespace and a schema. Objects in the same schema can use storage in different tablespaces, and a tablespace can hold contain storage from different schemas.

Schema objects can be created and manipulated using SQL. As an administrator, you can create and manipulate schema objects, just as you can the logical and physical structures of your database, using Oracle Enterprise Manager. The underlying SQL is generated for you by Oracle Enterprise Manager.

This section contains the following topics:

- Exploring Your Database's Schema Objects
- Managing Database Objects

> **See Also:** *Oracle Database SQL Reference* for more detailed
> information about schema objects, object names, and data types

Exploring Your Database's Schema Objects
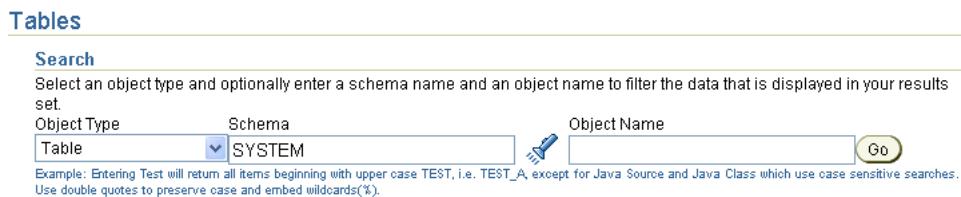
To explore and manage the schema objects in your database, open the Database
Administration property page (see Figure 5–2, "Database Administration Page") select
one of the links under the Schema heading. Figure 8–1 shows these links.

Figure 8–1   Schema Links

Click each link under the Schema heading in turn to see the all of the objects in your
database. You can also access any type of schema object from the Search section of any
schema object home page.

Figure 8–2   Search Section for Schema Object Pages

When you select a schema link, a standard *Object_Name* page appears. In the **Search**
section of the page, shown in Figure 8–2, you can access other schema objects,
including ones with no direct link under the Schema heading. To select a specific
schema object, use the drop down menu under the Object Type heading.

As you can see, you can further qualify your search for objects by specifying the
Schema and Object name fields.

The following are descriptions of some of the schema objects in your database.

Managing Database Objects

Some object types have many more management options than others but most have a
number of similarities. Every object in the database belongs to just one schema and has
a unique name within that schema. When you create an object, therefore, you must
ensure it is in the schema where you intend to locate it. Generally, you place all of the
objects that belong to a single application in the same schema.

A database object name must abide by certain rules as well as being unique within its
schema. For example, object names cannot be longer than 30 bytes and must begin
with a letter. If you attempt to create an object with a name that violates any of these
rules, Oracle raises an error.

The following sections describe how to view, create, and manage the various types of object in your database schemas:

- Managing Tables

- Managing Indexes

- Managing Views

- Managing Database Resident Program Units

- Other Schema Objects

> **See Also:** For more information about naming database objects, see *Oracle Database SQL Reference*

Managing Tables

Tables are the basic unit of data storage in an Oracle database. They hold all user-accessible data. Each table has columns and rows. In an employee table, for example, there can be (vertical) columns called employee and employee number. Each (horizontal) row in the table would contain a value for employee and employee number.

Viewing Tables

To view a table, click **Tables** on the Administration page of Enterprise Manager. The **Tables** property page appears as in Figure 8–3.

Figure 8–3   Tables Property Page

Initially, you see the tables for the schema that you are logged into. To see tables in other schemas, click the flashlight adjacent to the Schema box to select another schema, then click **Go** for the results to be displayed. For example, you can specify HR as your schema and EMPLOYEES as the table.

To view a table definition, from the Results list, either click the link in the **Table Name** column or select a table, such as HR.EMPLOYEES, then click **View**.

Viewing Table Data

To view data in a table from the Tables property page, select the table, then select **View Data**, from the Actions drop down menu, and click **Go**. A View Data For Table page appears showing the row data in the Results section.

In Figure 8–4, the Query box displays the SQL query that was executed in order to view the data for the HR.EMPLOYEES table.

Figure 8–4   View Data for Table Page

To sort data in a column, you can click any column name.

To change the query, click **Refine Query**. The Refine Query page lets you select the columns to display. It also lets you specify a WHERE clause for the SQL statement to limit the results.

You can write your own SQL query using a SELECT statement to see the contents of a table. You can execute SQL statements by starting an iSQL*Plus session from Enterprise Manager's Database Maintenance property page. A detailed description of the SELECT statement and its options is contained in the *Oracle Database SQL Reference.*

Creating Tables

You can use Enterprise Manager to create tables. You can do so as a system administrator or user with the CONNECT role.

Types of Table

The most common type of table in an Oracle database is a relational table, structured like the employee table described at the beginning of this section, "Managing Tables" on page 8-3. Two other types of table are supported: object tables and XMLType tables. Any of the three table types can be defined as permanent or temporary, the latter with the option of making the table structure available to multiple users or only to the user who creates the table.

Relational tables can be built in either heap or index organized structures. In the former, the rows are not stored in any particular order but in index organized tables, the row order is determined by the values in one of more columns which you select.

This chapter concentrates on permanent, heap organized tables. For information about other table types and when to use them, see the *Oracle Database Administrator's Guide*, *Oracle Database Concepts*, and the *Oracle Database Performance Tuning Guide*, and for the syntax required to create and alter them, see the *Oracle Database SQL Reference*.

Column Attributes

Each column in a table is defined with one or more characteristics as follows:

■ Data type: a required element which defines the nature of the data to be stored in the column and may a value to indicate the longest value that can be place in the column.

- Constraint: one or more optional elements that determine what values are valid in the column. Some constraints can be defined at the column level or at the table level. In Enterprise Manager, the only constraint defined at the column level on the Create Table property page is the Not Null constraint, which requires that a value is included in this column for every row in the table.

- Default value: A value that is automatically stored into the column whenever a new row is inserted without a value being provided for the column.

These column characteristics are described in more detail in the following sections.

Data Types When you create a table, you must specify a datatype for each of its columns. When you create a procedure or stored function, you must specify a datatype for each of its arguments.

These datatypes define the domain of values that each column can contain or each argument can have. For example, DATE columns cannot accept the value February 29 (except for a leap year) or the values 2 or 'SHOE'. Each value subsequently placed in a column assumes the column's datatype. For example, if you insert '17-JAN-2004' into a date column, then Oracle treats the '17-JAN-2004' character string as a DATE value after verifying that it translates to a valid date.

Table 8–1 is a summary of Oracle's built-in datatypes.

Table 8–1   Common Datatypes

Datatype	Description
VARCHAR2(size) [BYTE | CHAR]	Variable-length character string having maximum length size bytes or characters. Maximum size is 4000 bytes, and minimum is 1 byte or 1 character. You must specify size for VARCHAR2.
	BYTE indicates that the column will have byte length semantics; CHAR indicates that the column will have character semantics.
NVARCHAR2(size)	Variable-length character string having maximum length size characters. Maximum size is determined by the national character set definition, with an upper limit of 4000 bytes. You must specify size for NVARCHAR2.
NUMBER (p,s)	Number having precision p and scale s. The precision p can range from 1 to 38. The scale s can range from -84 to 127.
LONG	Character data of variable length up to 2 gigabytes, or 231 -1 bytes.
DATE	Valid date range from January 1, 4712 BC to December 31, 9999 AD
TIMESTAMP (fractional_seconds_ precision)	Year, month, and day values of date, as well as hour, minute, and second values of time, where fractional_seconds_ precision is the number of digits in the fractional part of the SECOND datetime field. Accepted values of fractional_ seconds_precision are 0 to 9. The default is 6.
TIMESTAMP (fractional_seconds_ precision) WITH TIME ZONE	All values of TIMESTAMP as well as time zone displacement value, where fractional_seconds_precision is the number of digits in the fractional part of the SECOND datetime field. Accepted values are 0 to 9. The default is 6.
TIMESTAMP (fractional_seconds_ precision) WITH LOCAL TIME ZONE	All values of TIMESTAMP WITH TIME ZONE, with the following exceptions: ■ Data is normalized to the database time zone when it is stored in the database. ■ When the data is retrieved, users see the data in the session time zone.

Table 8–1   (Cont.) Common Datatypes

Datatype	Description
INTERVAL YEAR (year_ precision) TO MONTH	Stores a period of time in years and months, where year_ precision is the number of digits in the YEAR datetime field. Accepted values are 0 to 9. The default is 2.
INTERVAL DAY (day_ precision) TO SECOND (fractional_seconds_ precision)	Stores a period of time in days, hours, minutes, and seconds, where: ■ day_precision is the maximum number of digits in the DAY datetime field. Accepted values are 0 to 9. The default is 2. ■ fractional_seconds_precision is the number of digits in the fractional part of the SECOND field. Accepted values are 0 to 9. The default is 6.
RAW(size)	Raw binary data of length size bytes. Maximum size is 2000 bytes. You must specify size for a RAW value.
LONG RAW	Raw binary data of variable length up to 2 gigabytes.
ROWID	Base 64 string representing the unique address of a row in its table. This datatype is primarily for values returned by the ROWID pseudocolumn.
UROWID [(size)]	Base 64 string representing the logical address of a row in a regular or index-organized table. The optional size is the size of a column of type UROWID. The maximum size and default is 4000 bytes.
CHAR(size) [BYTE | CHAR]	Fixed-length character data of length size bytes. Maximum size is 2000 bytes. Default and minimum size is 1 byte. BYTE and CHAR have the same semantics as for VARCHAR2.
NCHAR(size)	Fixed-length character data of length size characters. Maximum size is determined by the national character set definition, with an upper limit of 2000 bytes. Default and minimum size is 1 character.
CLOB	A character large object containing single-byte or multibyte characters. Both fixed-width and variable-width character sets are supported, both using the CHAR database character set. Maximum size is (4 gigabytes - 1) * (database block size).
NCLOB	A character large object containing Unicode characters. Both fixed-width and variable-width character sets are supported, both using the NCHAR database character set. Maximum size is (4 gigabytes - 1) * (database block size). Stores national character set data.
BLOB	A binary large object. Maximum size is (4 gigabytes - 1) * (database block size).
BFILE	Contains a locator to a large binary file stored outside the database. Enables byte stream I/O access to external LOBs residing on the database server. Maximum size is 4 gigabytes.

In most tables, you should only need columns of NUMBER, VARCHAR2, and DATE datatypes. When defining numeric data, you may want to use the precision option, to set the maximum number of digits in the number, and the scale option, to define how many of them are to the right of the decimal point.

For example, a field to hold monetary values might be defined as NUMBER(12,2) providing ten digits for the primary unit of currency (dollars, pounds, marks, and so on), and two digits for the secondary unit (cents, pennies, pfenigs, and so on). To define a VARCHAR2 field for character data, you must include the size value.

Set the size to the maximum number of bytes (or, optionally, characters) to be stored in the column. A column to hold postal codes for different countries, for example, might be restricted to 12 bytes by defining it as `VARCHAR2(12)`. `DATE` columns are automatically formatted by Oracle to include a date and time component.

Not Null Column Constraint The Not Null constraint on a column requires that column must contain a value whenever a row is inserted or updated. Unlike other constraints, described in "Table Level Constraints" on page 8-8, which may be defined as part of the column definition or part of the table definition, the Not Null constraint must be defined as part of the column definition.

Use a Not Null constraint when the data is required for the integrity of the database. For example, if all employees must belong to a specific department, then the column that contains the department identifier should be defined with a Not Null constraint. On the other hand, do not define a column as Not Null if the data may be unknown or may not exist when rows are added or changed, for example, the second, optional line in a mailing address.

A primary key constraint automatically adds a Not Null constraint to the column or columns included in the primary key.

Default Values When you define a column with a default value, any new rows inserted into the table will store the default value unless the row contains an alternate value for the column.

Table Level Constraints

In an Oracle database, you can apply rules to preserve the integrity of your data. For example, in a table containing employee data, the name column cannot accept NULL as a value. Similarly, in this table you cannot have two employees with the same ID.

Oracle enables you to apply data integrity rules called constraints to tables at the column level or, with the exception of the Not Null constraint discussed earlier, at the table level. Any attempt to insert or update a row that violates a constraint results in an error and the statement is rolled back. Likewise any attempt to apply a new constraint to a populated table will also result in an error if any existing row violates the new constraint.

The types of constraints you can apply at the table level are as follows:

- *Primary Key*---Requires that a column (or combination of columns) be the unique identifier of the row. A primary key column does not allow NULLs.

- *Unique Key*--Requires that every value in a column be unique. That is, no two rows can have duplicate values in a specified column or combination of columns. The set of columns is said to be the unique key.

- *Check*--Requires that a column (or combination of columns) satisfy a condition for every row in the table. A check constraint must be a boolean expression that is evaluated using the column value about to be inserted or updated to the row.

- *Foreign*---Requires that all column values in the child table exist in the parent table. The table that includes the foreign key is called the dependent or child table. The table that is referenced is called the parent table. An example of a foreign key constraint is when the department column of the employees table (child) must contain a department id that exists in the parent department table.

Constraints can be created and, in most cases, modified with a number of different status values. The options include enabled or disabled, which determine if the constraint is checked when rows are added or modified, and deferred or immediate,

which cause constraint validation to occur at the end of a transaction or at the end of a statement, respectively.

> **See Also:** *Oracle Database Concepts* for more information on constraints

Storage Attributes

When you create a table in a locally managed tablespace, using automatic segment space management, you should allow Oracle to build the table using its own storage algorithms. In case you need to store a table in a dictionary managed tablespace, you can set the storage characteristics for the table. However, the preconfigured database you installed (see Chapter 2, "Installing Oracle and Building the Database") and extended as described in Chapter 6, "Managing Database Storage Structures"should not contain such tablespaces. Therefore, these options are not discussed here but are described in the *Oracle Database Administrator's Guide* and *Oracle Database Concepts* manual if you need to use them.

Other Table Creation Considerations

When creating a table using the Enterprise Manager Create Table page, you may click the Show SQL button to see the SQL statement that has been constructed. This option will show the statement even if it is incomplete, so you need to complete your input to see the complete SQL statement when using this option.

You may also create a new table with the same the structure and contents as an existing table. This can be useful to build a test table that resembles a production table. To do this, select SQL in the Define Using list and complete the statement that is begun for you in the CREATE TABLE AS box. Your query can include all columns and rows from the original table or use the full power of the SQL SELECT statement to identify which rows you want, rename columns with column aliases, or exclude certain columns.

Your new table may include a column defined with an abstract data type (ADT). ADTs are created with the CREATE TYPE statement and are described in detail in the *Oracle Database SQL Reference*.

If you are creating a table containing one or more LOB columns, you may define different storage options for the LOB components of the table. In most cases, you should use the default settings but, if you want to override them, you click **Set Default LOB Attributes** and complete the entries in the Set Default LOB Attributes page. Additionally, you can set LOB attributes for a single column by selecting the column name and clicking the Advanced Attributes button.

You may also create tables with multiple partitions. Each partition is a separate database object with many of the characteristics of an individual table, although each partition contains the same column definitions as its parent table. Partitioning a very large table can simplify the management of the table because each partition can be configured independently of the other partitions. Also, some operations against a partitioned table may be more efficient than against the table if it were not partitioned. This is because the partitions may be spread across more disk drives than would be the case for a non-partitioned table and because the optimizer may be able to avoid processing the contents of every partition when processing certain SQL statements.

For some applications, particularly data warehousing, with large tables that are frequently queried but very rarely updated, you may create compressed tables. These require less disk storage than uncompressed tables (which are the default). To create a compressed table, you will need to write your own CREATE TABLE statement, because the option is not available in Enterprise Manager.

> **See Also:** *Oracle Database Concepts* and *Oracle Database Administrator's Guide* for design and management considerations of different table types and *Oracle Database SQL Reference* for the syntax to create tables.

Estimating the Size of New Tables

During table creation with Enterprise Manager, you can estimate the size of the table, allowing you to determine if you have sufficient room in your database, or on the available disks, to store the table. If you do not have room, you can still create the table but not populate it until you have obtained the necessary storage.

To estimate the table size, you must first enter your column names and datatypes, then click **Estimate Table Size** in the Create Table page. The Estimate Table Size page then prompts you for the projected number rows, which you enter. After clicking **Estimate Table Size**, Enterprise Manager returns its estimate in MB.

Table Creation Example

In the following example, we create a table called purchase_orders in the MYUSER schema we created in Chapter 7, "Administering Users and Security". The table will have columns *po_number*, *po_description*, *po_date*, and *po_item*.

To create a table in MYUSER's schema, log in to Oracle Enterprise Manager as either SYS or MYUSER and follow these steps:

1. Under the Administration page, click **Tables**. The Tables page appears listing the tables in the MYUSER schema. Note if you logged in as SYS, you need to click the flashlight and select MYUSER to manage tables in this schema.

2. Click **Create**. The Create Table: Table Organization page appears.

3. Select the box marked *Standard, Heap Organized*. Click **Continue**.

4. The Create Table page appears. See Figure 8–5, "Create Table Page".

Figure 8–5   Create Table Page

5. In the Name field, enter purchase_orders as the table name. In the Schema field, accept the MYUSER schema, and in the Tablespace field, accept <Default>.

6. In the Columns section, enter the following columns with the datatypes:

    ■   *po_id* as *NUMBER*

    ■   *po_date* as *DATE*

    ■   *po_desc* as *VARCHAR2(200)*

    ■   *po_item* as *VARCHAR2(100)*

    You can leave Scale, Not NULL, and Default Value blank.

    Note: If you want to the create table with partitions, you can do so at this step by clicking **Partitions**.

    ---

7. Click **OK**. An update message appears indicating the table has been successfully created.

Modifying Table Attributes

You can use Enterprise Manager to modify tables, such as adding and deleting columns or adding constraints. The following sections show how to add and delete a column, and how to add, change the status of, and drop a table constraint.

Adding a Column

In this example, we modify the *PURCHASE_ORDERS* table by adding a new column called po_item2. To add the new column:

1. In the Tables page, in the MYUSER schema, select the *PURCHASE_ORDERS* table in the results list and click **Edit**.

2. In the Edit Table page, click **Add 5 Table Columns**. An editable columns list appears.

3. Enter po_item2 as VARCHAR2(100) as the new column. You can leave the Scale, Not Null, and Default Value blank. Click **Apply**.

4. An Update Message appears indicating that the table has been modified successfully.

Dropping a Column

This example shows you how to drop column, in this case, the po_item2 column that was added to the *PURCHASE_ORDERS* table in the previous section ("Adding a Column"). To delete the column:

1. In the Tables page, in the MYUSER schema, select the *PURCHASE_ORDERS* table in the results list and click **Edit**.

2. In the Columns table on the Edit Table page, select the column you want to delete, po_item2.

3. Click **Delete**.

4. The row for that contained the information for the deleted row, po_item2, becomes blank to indicate that the table has been modified successfully.

Creating a New Table Constraint

You can add constraints to a table with Enterprise Manager. In our current example, we want to enforce that our purchase orders always be entered with an ID and that this column (po_id) be the primary key of the table. We will add the not null and primary key constraints to this column.

> **Note:** You can also add constraints during table creation.

To add these constraints to the purchase_orders table, do the following:

1. In the Tables page, in the MYUSER schema, select the PURCHASE_ORDERS table in the results list and click **Edit**.

2. To apply the Not Null constraint, check the Not Null box next to the po_id column. Click **Apply**.

3. To apply the primary key constraint, click the Constraints property page.

4. On the Constraints page, select PRIMARY from the drop-down menu, and click **Add**. The Add PRIMARY Constraint page appears.

5. In the available columns list, select po_id and move it to the selected columns list. Do not change the check boxes in the Attributes area of the page because so that the default settings for the constraint are retained.

6. Click **OK**. On the Edit Table page, click **Apply**. A confirmation message appears. You can view the new constraints by navigating to the View Table page

Modifying an Existing Constraint

You can change the status of an existing constraint, for example, from an enabled to a disabled state. Using the primary key constraint created in the previous section, "Creating a New Table Constraint", here is an example of how to disable a constraint:

1. In the Tables page, in the MYUSER schema, select the PURCHASE_ORDERS table in the results list and click **Edit**.

2. To locate the constraint to be disabled, click the **Constraints** property page.

3. On the Constraints page, select the row containing the PRIMARY constraint from the Constraints table, and click **Edit**.

4. Under the Attributes section of Edit PRIMARY Constraint window, select the Disabled box.

5. Click **OK**. On the Edit table page, click **Apply**. A confirmation message appears and the value in the Disabled column for the primary key constraint should be Yes.

Dropping a Constraint

You can drop constraints to a table with Enterprise Manager. Although you do not have to disable a constraint before dropping it, you can determine whether the constraint can be dropped by attempting to disable it first. When a constraint in a parent table is used to enforce a foreign key constraint in a child table, the constraint cannot always be disabled or dropped if the child table contains dependent rows.

Continuing the current example, we drop the primary key constraint that we created earlier in the section, "Creating a New Table Constraint", by completing the following steps:

1. In the Tables page, in the MYUSER schema, select the PURCHASE_ORDERS table in the results list and click **Edit**.

2. To locate the constraint to be disabled, click the **Constraints** property page.

3. On the Constraints page, select the row containing the PRIMARY constraint from the Constraints table, and click **Delete**.

4. On the Edit table page, click **Apply**. A confirmation message appears.

Dropping a Table

If you no longer need a table or its contents, you can drop the table using Enterprise Manager. Be certain that you really do not need the data in the table before you drop it because it may be difficult and time-consuming to retrieve the records, if they can be retrieved, after you execute the drop statement.

To drop a table, follow these steps:

1. From the Tables page, identify the table you want to delete by selecting the required schema and the required table name.

2. Select the table you need to delete from the Results table and click **Delete**.

3. The Confirmation page lets you choose whether to continue the operation. If you are certain you have identified the correct table, click **Yes**.

4. The Edit table page displays a confirmation message if the table is successfully deleted.

Managing Indexes

Indexes are optional structures associated with tables. They can be created to improve query performance. Just as the index in this book helps you to quickly locate specific information, an Oracle index provides a quick access path to table data.

Indexes can be created on one or more columns of a table. After an index is created, it is automatically maintained and used by Oracle. Changes to a table's data or structure, such as adding new rows, updating rows, or deleting rows, are automatically incorporated into all relevant indexes with complete transparency to the user.

Some indexes are created implicitly through constraints that are placed on a table. For example, a column with the constraint that its values be unique causes Oracle to create a unique key index.

When you click the **Indexes** link under the **Schema** heading of the Administration page, an Indexes page appears, which is similar in content and functionality to the Tables page shown in Figure 8–3.

Viewing Indexes

To find out what indexes exist in your database, use the search options in the Indexes page. The search works like the one for tables described earlier except that, when searching by object name, you can use the name of the index or the name of the table on which the index is built. Select whether the search is for the name of an index or of a table by clicking the appropriate entry in the Search By list.

To view the details of an index, click its name in the Results list on the Indexes page. You can find basic information about the index, such as its status and the column or columns on which it is built, in the General area of the main page for each individual index. Other areas include information about the space consumed by the index and the options used in its definition.

Creating a New Index

Oracle automatically creates the indexes necessary to support data integrity defined with constraints when you add or enable those constraints. For performance purposes, you might want to add an index to the columns you define in a child table when adding a foreign key constraint. Before you add any other indexes, you should be ready to examine the performance of your database after they are added.

After index creation, Oracle automatically synchronizes the index with any subsequent inserts, updates or deletes to the base table.

Indexes are generally of value to queries and to SQL statements that need to operate on a single, existing row or a small number of existing rows. However, too many indexes can cause serious problems by increasing the processing overhead for statements that add, modify, or delete rows. In some cases, a statement could use two or more indexes and the optimizer will pick just one of them. Unless other statements can take advantage of the unused indexes, they are not providing any benefit. Therefore, you might find yourself deleting indexes if you create too many.

Index Types

Indexes can be categorized in a number of ways, the primary options are listed in the following section:

Standard (B-tree) and Bitmap: A standard, B-tree index contains an entry for each value in the index key along with an address to the row where the value is stored. A B-tree

index is the default and most common type of index in an Oracle database. A bitmap index uses strings of bits to encapsulate values and potential row addresses. It is more compact than a B-tree and can perform some types of retrieval more efficiently. For general use, however, a bitmap index requires more overhead during row operations on the table and should be used primarily for data warehouse environments as described in the *Oracle Data Warehousing Guide*.

Ascending and Descending: The default search through an index is from lowest to highest value, where character data is sorted by ASCII values, numeric data from smallest to largest number, and date from the earliest to the latest value. This default behavior is performed in indexes created with as ascending indexes. You can cause index searches to reverse the search order by creating the related index with the descending option.

Column and Functional: Typically, an index entry is based on the value or values found in the table's column or columns. This is a column index. Alternatively, you can create a function-based index in which the indexed value is derived from the table data. For example, to find character data which can be in mixed case, you could use a function-based index to look for the values as if they were all in uppercase characters.

Single Column and Concatenated: You can create an index on just one column (single column index) or on multiple columns (concatenated index). Concatenated indexes are useful when all of the columns are likely to be included in the WHERE clause of frequently-executed SQL statements. For concatenated indexes, you should define the columns used in the index carefully so that the column with the fewest duplicate values is named first, the column with next fewest duplicate values is second, and so on. Columns with many duplicate values or many rows with no value (Nulls) should not be included or should be the last-named columns in the index definition.

Non-partitioned and Partitioned: As with tables, you can create your indexes without or with partitions. In most situations, it is most useful to partition an index when the associated table is also partitioned and the index uses the same partitioning scheme as the table, known as a local index. This does not mean you must partition an index like the underlying table. You may even create a non-partitioned, or global, index on a partitioned table.

> **See Also:** *Oracle Database Concepts* and *Oracle Database Administrator's Guide* for design and management considerations of different index types and *Oracle Database SQL Reference* for the syntax to create tables.

Index Creation Example

You can create indexes with Enterprise Manager. To create an index, you specify one or more columns to be indexed and the type of index you want to create.

The following example creates a standard B-tree index on the SUPPLIER_ID column in the SH.PRODUCTS table, which is part of the sample schema. To create this index, do the following:

1. On the Administration page under Schema, click **Table**. The Tables page appears.

2. In the Search section, enter SH as the Schema and click **Go**. Under Results, the list of tables in the SH sample schema appears.

3. Select the PRODUCTS table and select **Create Index** from the Actions drop-down menu. Click **Go**.

4. The Create Index page appears. Name the new index PRODUCTS_SUPPLIER_
   IDX. For *Tablespace*, accept <Default>, and select Standard B-tree as the index type.

5. In the list of table columns, select the SUPPLIER_ID column by entering 1 in the
   Order column. Accept ASC as the Sorting Order. Click **OK** to create the index.

6. After the index is created, the Indexes page appears with a confirmation message.
   The new index is listed under Results.

Dropping Indexes

To drop the table using Enterprise Manager, follow these steps:

1. From the Indexes page, identify the index you want to delete by selecting the
   required schema and the required index name.

2. Select the index you need to drop from the Results table and click **Delete**.

3. The Confirmation page lets you choose whether to continue the operation. If you
   are certain you have identified the correct index, click **Yes**.

4. The Edit table page displays a confirmation message if the index is successfully
   deleted.

Note: You cannot drop an index that is currently used to enforce a
constraint. You must disable or drop the constraint and then, if the
index is not dropped as a result of that action, drop the index.

Managing Views

Views are customized presentations of data in one or more tables or other views. They
can be thought of as stored queries. Views do not actually contain data, but instead
they derive their data from the tables upon which they are based. These tables are
referred to as the base tables of the view.

Like tables, views can be queried, updated, inserted into, and deleted from, with some
restrictions. All operations performed on a view actually affect the base tables of the
view. Views provide an additional level of security by restricting access to a
predetermined set of rows and columns of a table. They also hide data complexity and
store complex queries.

Click the **Views** link under the **Schema** heading of the Administration page to look at
the views in the database. The Views page appears and has functionality similar to the
Tables page shown in Figure 8–3.

Look at the views in the SYS schema. These views enable you to look at data
dictionary information or ongoing database activity. The views in the SYS schema are
also the source of information for many of the Oracle Enterprise Manager pages.

The data dictionary views are called static, because they change only when a change is
made to the data dictionary (for example, when a new table is created or a user is
granted new privileges). The views that monitor ongoing database activity are called
dynamic performance views.

Many data dictionary tables have three corresponding views:

- A DBA_ view displays all relevant information in the entire database. DBA_ views
  are intended only for administrators.

- An ALL_ view displays all the information accessible to the current user, including
  information from the current user's schema as well as information from objects in

other schemas, if the current user has access to those objects by way of grants of privileges or roles.

- A USER_ view displays all the information from the schema of the current user. No special privileges are required to query these views.

The columns of the ALL_, DBA_, and USER_ views corresponding to a single data dictionary table are usually nearly identical.

The names of dynamic performance views, available only to administrators, start with the characters V$ and are often referred to as the V$ views.

Complete descriptions of the views in the SYS schema are contained in the *Oracle Database Reference*.

View Creation Example

The following example creates a view derived from the HR.EMPLOYEES table, which is part of the sample schema. The view filters all direct reports for the manager Stephen King, whose employee_id is 100. In an application scenario, this view adds an additional level of security to the HR.EMPLOYEES table while providing a suitable presentation of relevant information for manager Stephen King.

To create this view, do the following:

1. On the Database Administration page, under Schema, click **Views**. The Views page appears.

2. Click **Create**. The Create View page appears.

3. Enter the following fields:

    - Name: *SKING_VIEW*

    - Schema: *HR*

    - Query Text:

    ```
    SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,
      PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT,
      MANAGER_ID, DEPARTMENT_ID
      FROM HR.EMPLOYEES WHERE MANAGER_ID =  100
    ```

4. Click **OK**. The Views page appears with a confirmation message. The new view is listed under results.

5. To see the new view, select it and choose **View Data** from the drop-down menu. Click **Go**. The selected data in the view appears under Result.

Managing Database Resident Program Units

Oracle offers the capability to store programs in the database. This allows commonly-required code to be written and tested once and then accessed by any application that requires the code. Database resident program units also ensure that the same processing is applied to the data when the code is invoked, making the development of applications easier and providing consistency between developers.

Database resident programs can be written in PL/SQL or Java and you can use Enterprise Manager to manage the source types such as PL/SQL packages, procedures, triggers, and functions and Java sources and classes. The actions include creating, compiling, creating synonyms for, granting privileges on, and showing dependencies for these source types.

This book describes the main types of PL/SQL program units. You should have experience with the PL/SQL language before attempting to create or modify these units. The following sections cover:

- Managing Packages
- Managing Package Bodies
- Managing Standalone Subprograms
- Managing Triggers

> **See Also:** *PL/SQL User's Guide and Reference* for help in understanding and writing PL/SQL code and program units.

Managing Packages

A package is a structure that contains definitions, blocks of PL/SQL code, or both in a single unit. The contents of the package are invoked by naming the package and the element within it using a dot notation. For example, to execute the procedure called STORE_SQE in the CTX_QUERY package owned by the CTXSYS user, you would invoke it with the statement CTXSYS.CTX_QUERY.STORE_SQE(<arg1>,<arg2>) where <arg1> and <arg2> are the values for the two procedure arguments. To hide the complexity of packaged element names, you might want to use synonyms, described later in "Other Schema Objects" on page 8-20

You can examine packages that already exist in the database by clicking the **Packages** link on the Enterprise Manager Administration property page to open the Packages property page. You can then search for packages by schema, name, both schema and object name, or neither (this latter option will include every package in the result set). To see a package definition, click its name in the Results list or select the package in the Results list and click **Edit**. The Edit Package property page contains the package name, schema, status, and source.

The source contains the code for a package and may contain the following elements:

- Authentication information: determines which security schema is used when the package contents are executed
- Procedure definitions: the name and list of arguments for one or more procedures
- Function definitions: the name, list of arguments, and return data type for one or more functions
- Variables: definitions of variables that globally available to the user of the package
- Other public elements: definitions of such structures as types, constants, and exceptions that are needed by package users

From the Edit Package property page, you can also perform the following actions:

- Modify the package definition by typing new contents in the Source box and clicking **Apply**.
- Compile the package by clicking **Compile**, an action that may be required if the package status has become invalid for some reason.
- Modify the package body (see next section) by clicking **Modify Package Body**.

> **Note:** You can also navigate to the Edit Package property page from the Edit Package Body property page, described in the next section, by clicking **Edit Package**.

As with other schema objects, you can create a new object by clicking **Create** on the object page. In this case, you use Package property page. The Create Package property page provides fields to enter the package name, schema name, and the PL/SQL code for the package definition.

To delete a package, select the package from the Results list on the Packages property page and click **Delete**.

Managing Package Bodies

Package bodies contain the PL/SQL code for the procedures and functions defined in the definitions of the packages to which they belong. The same navigation paths and options for adding (creating), editing (modifying), compiling, and deleting package bodies are available in Enterprise Manager as for packages, as described in the previous section. This includes direct navigation between the Edit Package Body and the Edit Package property pages as mentioned in the previous section.

In some cases, the developer of the package may have wrapped one or more of the individual procedures or functions. This converts the code into non-readable strings which makes it difficult for other developers to misuse the application or for competitors to see the algorithms. However, wrapped code cannot be edited and the Source box in the **Edit Package Body** page will contain a message such as `Source code could not be displayed, because it is wrapped` and will not be available for modification.

Managing Standalone Subprograms

Procedures and functions do not have to belong to a package. Program units that are created outside of a package are called standalone subprograms. You can use Enterprise Manager to create, modify, compile, and delete standalone subprograms just like packaged subprograms except that they do not have a parent package (and, consequently, no navigation path in Enterprise Manager from or to the package property pages). There are separate property pages for procedures and for functions which you can find from the Enterprise Manager Administration property page.

To execute a standalone subprogram, you only need to include its schema name and object name: there is no package name. Even so, you may want to create synonyms for subprograms that are used regularly.

Note that functions return a value, whether or not they use an argument list, the datatype for which must be defined as part of the function code.

Managing Triggers

A database trigger is a stored subprogram associated with a database table, view, or event. For instance, you can have Oracle fire a trigger automatically before or after an `INSERT`, `UPDATE`, or `DELETE` statement affects a table. One of the many uses for database triggers is to audit data modifications. For example, the following table-level trigger fires whenever salaries in the emp table are updated:

```
CREATE TRIGGER audit_sal
AFTER UPDATE OF sal ON emp
FOR EACH ROW
BEGIN
INSERT INTO emp_audit VALUES...
END
```

The executable part of a trigger can contain procedural statements as well as SQL data manipulation statements. Besides table-level triggers, there are instead-of triggers for views and system-event triggers for schemas.

> **See Also:** *Oracle Database Application Developer's Guide - Fundamentals* and *PL/SQL User's Guide and Reference* for more information about the uses for and creation of triggers.

Triggers are another type of schema object that you can manage through Enterprise Manager. They have their own property page which you can open by clicking **Triggers** under the Schema heading on the Administration page. From the Triggers property page, you can create new triggers, find existing triggers and either delete them or open an Edit Triggers page.

The **Create Triggers** and **Edit Triggers** property pages look very similar to the corresponding pages for packages, package bodies, and standalone subprograms. The major difference is that there two check boxes: one to check if you want to replace an existing trigger and one to immediately enable a trigger after it is created or edited. The other difference is that the box containing the PL/SQL code is labeled *Trigger Body* instead of *Source*. Otherwise, the creation, editing, and compilation of triggers is exactly the same as for other database PL/SQL units.

Because triggers execute automatically as a result of the triggering DML on the table, there is no way to execute them directly.

Other Schema Objects

Other schema objects are not discussed in detail in this book, but are described in Oracle Enterprise Manager online help. As with the schema objects described in the preceding sections, they can be managed from Enterprise Manager by clicking the required link under the Schema heading on the Administration property page. The schema object pages provide:

- Search capabilities to find existing objects

- Links to drill down to individual objects, with further drill down to details as appropriate

- Navigation to screens where you can add, edit, or delete an object

search capabilities to find existing objects (with the ability to drill down to details as appropriate), These other objects include:

- Clusters

    Clusters are groups of one or more tables physically stored together because they share common columns and are often used together. Because related rows are physically stored together, fewer disk accesses may be required to retrieve records defined with a WHERE clause in a SQL statement.

    Like indexes, clusters do not affect application design. Whether a table is part of a cluster is transparent to users and to applications. Data stored in a clustered table is accessed by SQL in the same way as data stored in a nonclustered table.

- Synonyms

    A synonym is an alias for any schema object such as a table or view. Synonyms provide an easy way to hide the underlying database structure from an application or a user. For example, you can create a synonym named PERSONNEL

as an alias for the EMPLOYEES table in the HR schema. With the synonym in place, the following query:

```
SELECT employee_id, salary
FROM hr.employees
ORDER BY salary
```

can be rewritten more simply as:

```
SELECT employee_id, salary
FROM personnel
ORDER BY salary
```

An additional benefit of synonyms is that you can use the same synonym in a development database as in the production database, even if the schema names are different. This allows the application code to run unmodified in both environments. For example, the second version of the preceding query would work in a development database where the EMPLOYEES table was in the DEV1 schema if the PERSONNEL synonym were defined.

Because a synonym is simply an alias, it requires no storage other than its definition in the data dictionary.

■   Sequences

A sequence is a database object from which multiple users can generate unique integers. Using a sequence generator to provide the value for a primary key in a table is an easy way to guarantee that the key value is unique.

■   Database links

A database link is a schema object that instructs Oracle to connect to a remote database to access an object there. They are used in distributed database environments which are described in the *Oracle Database Administrator's Guide*.

Loading Data Into Tables

You can use Enterprise Manager to load data into tables in batch. Batch loading is useful when you have a lot of data. You can load data from operating system files or from other databases. You can also export data into files.

For loading data with Enterprise Manager, one method is to use control (.ctl) and data (.dat) files. These files are formatted as standard SQL*Loader files.

The following example loads data into the *PURCHASE_ORDERS* table we created with *MYUSER* in "Managing Tables" on page 8-3. For simplicity, this example loads only three rows. Your control and data files can be different, but the idea is the same for this method of loading.

This example requires us to create two files: *load.ctl* and *load.dat*.

The control file *load.ctl* looks like:

```
LOAD DATA
 INTO TABLE myuser.purchase_orders
 REPLACE
 FIELDS TERMINATED BY ';'
(po_id,
po_date date 'yyyymmdd',
po_desc char(25),
po_item char(25),
po_item2 char(25))
```

Our data file *load.dat* looks like:
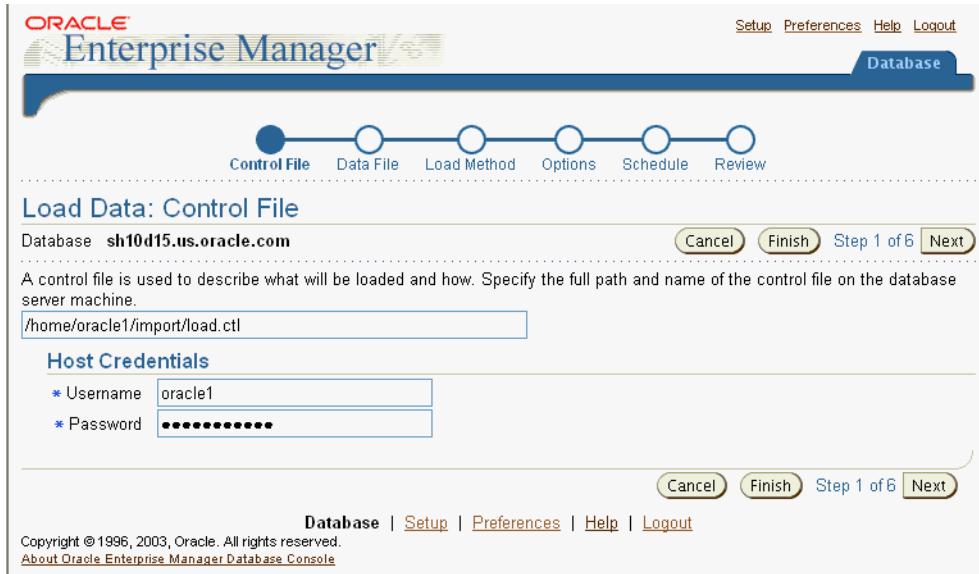
```
1; 20031025; DSL/Wireless Expense Oct; Home DSL; Wireless
2; 20031125; Office Equipment; Optical Mouse; Network Router
3; 20031225; Computer System; Dell Pentium 4 Laptop; Docking Bay
```

> **See Also:** *Oracle Database Utilities* for more information about SQL*Loader control files

Figure 8–6   Load Data page

To load the data file into the `myuser.purchase_orders` table, you must log in to Enterprise Manager as `SYS` or as `myuser`, who has administrator privileges:

1. From the Maintenance page, click **Load Data from File** under Utilities to begin a six step loading wizard. Figure 8–6, "Load Data page" shows the first step.

2. In the Load Data: Control File page, enter the full path of your control file on the database server machine. Also enter the username and password for the host machine. Click **Next**.

3. In the Load Data: Data File page, select **Provide the full path and name on the database server machine** and type in the path. Click **Next**.

4. In the Load Data: Load Method page, accept the default **Conventional Path** as the loading method. If you have a lot of data, you can use the Direct Path method.

5. In the Load Data: Options page, under Optional Files, select **Generate log file where logging information is to be stored**. You can accept the default file name and path or enter a different one. Note that this page gives you the option of limiting the number of rows loaded. Since we are only loading three rows in this example, we leave this un-selected. Click **Next**

6. In the Load Data: Schedule page, enter a **Job Name** and **Description**. Select **Immediately** to run the job now. Click **Next**.

7. The Load Data: Review page lets you review your file names and loading methods. If you want to change something, you can click the Back button. Otherwise, click **Submit Job** to start the loading.

8. The Status page appears. It should say Load Data Submit Successful. Click **View Job** to view the job summary. This summary page should indicate that the job has succeeded. If not, you can view the log file by clicking your job under the Logs heading or by viewing the log file directly.

9. You can confirm the loaded data by navigating to the Tables page, selecting the table, and choosing *View Data* as the action. See Figure 8–7, "View Data: MYUSER.PURCHASE_ORDERS".

Figure 8–7   View Data: MYUSER.PURCHASE_ORDERS

Managing Access to Schema Objects

By default, only the owner of an object or users with advanced system privileges can access the object. For each object you want to make available to other users, you need to include the object and the access methods you want to allow in a role. The role can then be assigned to the users who require the access as described in the "Granting Roles" on page 7-11.

To add an object privilege to a role, first create or identify an existing role that you want to use. Find or create the role by clicking **Roles**, under the Security heading, on the Database Administration page. To use an existing role, click its name; to create a new role, click **Create** on the **Roles** page.

On the Edit Role or Create Role property page, depending on whether you are using an existing role or creating a new one, follow these steps to add an object privilege to the role:

1. Click **Object Privileges**.

2. Choose the object type from the **Select Object Type** then click **Add**.

3. Enter the schema name and object name, with a period separator, in the **Select <Object_Type> Objects** window, or click the flashlight icon to initiate a search for the required object.

4. Use the move and remove arrows to place the required privileges into the **Selected Privileges** window.

5. Click **OK** to add the privileges to the role.

To revoke object privileges, you can either delete the role if it contains only the object privileges you want to remove, or you can remove individual privileges from a role. To perform either of these actions, you must first find the required role as described earlier in this section. To delete the role, select it from the Results table and click **Delete**. To remove an individual privilege, click the role name in the Results table to navigate to the Edit Role page then click **Object Privileges**. Then select, in turn, each privilege you want to revoke and click **Delete** then **Apply** to remove it from the role.

Some Schemas of Interest

Some schemas with tables of particular interest are discussed in the sections that follow.

SYS Schema

All of the base tables and views for the database's data dictionary are stored in the schema SYS. The data dictionary is a collection of tables that describe the Oracle database. The data dictionary is created in the SYSTEM tablespace at database creation and it is updated by the database server every time that a data definition language (DDL) statement is issued. Oracle accesses the data dictionary to find information about users, schema objects, and storage structures. The data dictionary is critical for the operation of Oracle.

Oracle users can use the data dictionary as a read-only reference for information about the database. They do not access the tables directly, but rather they access a set of views built over the tables.

Objects in the SYS schema should never be modified by any user or database administrator, and no one should create any tables in the schema of user SYS. Special system privileges protect access to the SYS schema.

SYSTEM Schema

The SYSTEM schema contains additional tables and views that display administrative information, and internal tables and views used by various Oracle options and tools. Never create in the SYSTEM schema tables of interest to individual users.

Sample Schemas

The sample schemas provide a common platform for examples in Oracle releases. They are a set of interlinked schemas aimed at providing a layered approach to complexity:

- HR (Human Resources)—This is a simple schema for introducing basic topics. An extension to this schema supports Oracle Internet Directory demos

- OE (Order Entry)—This schema is for dealing with matters of intermediate complexity. A multitude of datatypes is available in the schema.

The OC (Online Catalog) subschema is a collection of object-relational database objects built inside the OE schema.

- PM (Product Media)—This schema is dedicated to multimedia datatypes.

- QS (Queued Shipping)—This is a set of schemas gathered under the main schema name QS to demonstrate Oracle Advanced Queuing capabilities.

- SH (Sales History)—This schema is designed to allow demos with larger amounts of data. An extension to this schema provides support for advanced analytic processing.

Schemas: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter and includes annotated screen shots.

To view the Schemas OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/schema/schema.htm
```

9

Performing Backup and Recovery

This chapter introduces you to Oracle backup and recovery as performed through Enterprise Manager. The goals are to introduce the basic concepts of Oracle backup and recovery, show how to configure your database for backup and recovery using the Oracle-suggested disk-based backup strategy, and then walk you through recovery of a full database backup.

> **Note:** The instructions in "Configuring Your Database for Basic Backup and Recovery" on page 9-4 describe how to set up your database to take advantage of Oracle-suggested backup strategies. If you chose to configure automated backups when creating the database using the Database Configuration Assistant, you do not need to perform the steps in this section. See "Using DBCA to Create and Configure a Database" on page 2-5 for more details on creating a database already configured for automated daily backups using the Oracle-suggested backup strategy.

This chapter includes the following topics:

- Overview of Oracle Backup and Recovery
- Configuring Your Database for Basic Backup and Recovery
- Backing Up Your Database
- Restore and Recovery Operations
- Managing Your Backups
- Backup and Recovery: Oracle by Example Series

Overview of Oracle Backup and Recovery

Backup of an Oracle database generally refers to physical backup, protecting the files that make up your database. The files protected by the backup and recovery facilities built into Enterprise Manager include datafiles, control files, server parameter files (SPFILEs), and archived redo log files. With these your database can be reconstructed. The backup mechanisms that work at the physical level protect against damage at the file level, such as the accidental deletion of a datafile or the failure of a disk drive.

Logical-level backups, such as exporting database objects like tables or tablespaces, may be a useful supplement to physical backups for some purposes but cannot protect your entire database. The focus in Oracle backup and recovery is generally on the physical backup of database files, which permit the full reconstruction of your database.

Oracle's flashback features provide a range of physical and logical data recovery tools as efficient, easy-to-use alternatives to physical and logical backups. This chapter will introduce two of the flashback features that operate at a logical level: Oracle Flashback Table, which lets you revert a table to its contents at a time in the recent past; and Oracle Flashback Drop, which lets you rescue dropped database tables. Neither requires advance preparation such as creating logical-level exports to allow for retrieval of your lost data, and both can be used while your database is available. *Oracle Database Backup and Recovery Advanced User's Guide* discusses the flashback features of the Oracle database at greater length.

Oracle Enterprise Manager's physical backup and recovery features are built on Oracle's Recovery Manager (RMAN) command-line client. Enterprise Manager carries out its backup and recovery tasks by composing RMAN commands and sending them to the RMAN client. Enterprise Manager makes available much of the functionality of RMAN, as well as providing wizards and automatic strategies to simplify and further automate implementing RMAN-based backup and recovery.

After you are familiar with the basics of backup and recovery through Oracle Enterprise Manager, refer to *Oracle Database Backup and Recovery Basics*, and *Oracle Database Backup and Recovery Advanced User's Guide*, for more details on the full range of Oracle's backup capabilities.

Oracle Backup, Restore and Recovery Concepts

To back up your database is to make backup copies of your datafiles, control file, and archived redo logs if any. Restoring a database from backup is simply copying the physical files that make up the database from some backup medium (disk or tape) to their locations during normal database operation. Recovery of your database is the process of updating database files restored from a backup with the changes made to the database since the backup, typically using redo log files.

Consistent and Inconsistent Backups

A backup can be consistent or inconsistent. A consistent backup is one in which there are no changes in the redo log that have not already been applied to the datafiles at the time of the backup. In an inconsistent backup, the online and archived redo logs can contain changes that have not yet been applied to the datafiles.

After being restored from a consistent backup, the database can be opened immediately. There is no need to preserve the redo logs to be able to use a consistent backup. However, in order to create a consistent backup of your database, you must carry out a normal shutdown of your database, and then back up your datafiles. This process is known as an offline backup, because the entire database is offline for the duration of the backup process.

When your datafiles have been restored from an inconsistent backup, the database cannot be opened until the restored datafiles are updated with any pending changes in the redo log. This step brings the datafiles to a consistent state, so that all datafiles reflect all changes as of a specific point in time. After the datafiles are consistent, the database can be opened. The process in which changes from the redo log are applied to the datafiles is called media recovery, and it is described in the next section. To use inconsistent backups, however, you must run your database in ARCHIVELOG mode. This preserves your redo log files for use in media recovery.

In spite of the name, an inconsistent backup is as robust a form of backup as a consistent backup. The great advantage to taking inconsistent backups is that you do not have to take your database offline to create an inconsistent backup. Users can continue to update the database during the backup process.

Media Recovery

If you restore the archived redo logs and the datafiles from backup, Oracle will perform the media recovery process when you try to open the database. Any database transactions in the online and archived redo logs not already reflected in the datafiles are applied to the datafiles, bringing them to a transaction-consistent state before the database is opened.

Media recovery can be either complete recovery or point-in-time recovery. In complete recovery datafiles are restored from backup, and all changes from the archived and online redo logs are applied to the datafiles. The database is returned to its state at the time of failure, and can be opened with no loss of committed changes.

In point-in-time recovery, you return your database to its contents at a target time of your choosing in the past. Starting with a set of datafile backups created prior to the target time and a complete set of archived redo log files from the time of the backup through the target time, you can have the database re-apply to the datafiles each change since the datafiles were backed up. When all changes as of the target time have been re-applied, the datafiles are returned to their contents as of the target time. Provided that you keep all archived redo logs back to the time of a given database backup, point-in-time recovery can return your whole database to its state at any time between the time of that backup and the most recent change in the archived redo logs. Point-in-time recovery is sometimes referred to as incomplete recovery, since not all changes up through the present are applied.

Media recovery requires a control file, datafiles (typically restored from backup) and all online and archived redo logs containing changes since the time the datafiles were backed up. It is typically used only in the case of database failure (caused either by a media failure, such as the loss of a file or disk, or a user error, like the deletion of the contents of a table).

Complete recovery will be the focus of the examples in this book. Incomplete recovery is an advanced technique discussed in *Oracle Database Backup and Recovery Advanced User's Guide*.

The RMAN Repository

RMAN's record of all backup and recovery files and activities for your database is known as the RMAN repository. Every backup action you perform through RMAN or Enterprise Manager is recorded in the repository, along with the location of and other information about every backup created or altered on disk or tape. At recovery time, you can simply issue a command like RESTORE DATABASE and Oracle uses the record in the repository to select backups on disk and tape needed to complete the recovery.

The primary store for the RMAN repository is the control file for your database. This is one more reason why protecting your control file is a vital part of your backup strategy. In some installations, a second copy of the RMAN repository is stored in a separate Oracle database called the recovery catalog. Use of a recovery catalog is optional and is beyond the scope of this book.

The Flash Recovery Area

To simplify the management of backup and recovery related files, Oracle lets you create a flash recovery area for your database. Designate a location (typically, a directory on disk) and a maximum disk quota for the flash recovery area, and a retention policy to specify your database recoverability goals. You can then direct backup-related activities (including archiving of redo logs) to store their files there, and Oracle will automatically manage this storage, deleting files no longer needed to

meet your recoverability objectives due to their age or due to their having been backed up to tape. Use of a flash recovery area simplifies backup storage management tasks and is therefore strongly recommended.

Because the flash recovery area is so useful, most examples in this chapter will assume the use of a flash recovery area.

Configuring Your Database for Basic Backup and Recovery

To take maximum advantage of Oracle's features for automatic managing backup and recovery, configure your database as follows:

- Use a flash recovery area, to automate storage management for most backup-related files.

- Run your database in ARCHIVELOG mode, so that you can perform online backups and have data recovery options such as complete and point-in-time media recovery.

- Use the flash recovery area as an archive log destination for your database.

You must also set a number of policies governing which files are backed up, what format is used to store backups on disk, and when files become eligible for deletion from the flash recovery area. Finally, you should make a record of two pieces of information about your database: the DBID and the DB_UNIQUE_NAME. This information is needed in some disaster recovery situations, such as restoring a lost database control file.

Planning Space Usage and Location for the Flash Recovery Area

To better protect your data, put the flash recovery area on a separate disk from the working set files of your database, to avoid the disk being a single point of failure for your database.

The amount of disk space to allocate for the flash recovery area depends upon the size and activity levels of your database, which determine the size of your datafiles and redo log files, and your recovery objectives, which dictate what kinds of backups you take, when you take them, and how long you must keep them.

Retention Policy and the Flash Recovery Area

Space management in the flash recovery area is governed by a backup retention policy. Retention policy determines when files are obsolete, meaning that they are no longer needed to meet your data recovery objectives.

Retention policies can be based on redundancy of backups, or a recovery window. Under a redundancy-based policy, the flash recovery area considers a backup of a file to be obsolete only when a specified number of more recent backups of that file are present. For example, if you specify a retention policy requiring two backups of each file be kept, and make backups nightly starting on a Monday night, then after the Wednesday night backup succeeds, the Monday night backup becomes redundant because the Tuesday and Wednesday backups are available.

Under a recovery window-based policy, you specify a time interval measured in days, and files become obsolete only when they are no longer needed for successful complete recovery or point-in-time recovery to any point within that number of days into the past. For example, if you specify a recovery window of three days, a backup of all datafiles from at least three days ago must be retained, along with a full set of archived redo logs since that backup.

> **Note:** Depending upon your backup strategy, a recovery window-based retention policy can require the retention of large quantities of data. If you specify a retention policy requiring a three day recovery window, and you take a full database backup only on the first night of each month and no other backups, on the 29th of the month you must retain the full database backup from the first night, as well as 28 days of archived redo logs.

A redundancy-based retention policy makes it easier to predict space usage in the flash recovery area, but do not allow you to predict how far into the past you can recover your database. A recovery window-based policy offers better protection for your data, but can make storage requirements for backups harder to predict. As already noted, even a short recovery window combined with a poorly designed backup strategy can cause unexpectedly high space requirements. (A reasonable backup strategy would include more frequent datafile backups.) Oracle recommends the use of a recovery-window-based backup retention policy as part of your backup strategy.

Even after they are obsolete, files are generally not deleted from the flash recovery area until space is needed to store new files. As long as space permits, files recently moved to tape will remain on disk as well, so that they will not have to be retrieved from tape in the event of a recovery.

The automatic deletion of obsolete files and files moved to tape from the flash recovery area makes the flash recovery area a very convenient redo log archiving destination. Other archiving destinations require manual cleanup of archived redo logs no longer needed on disk for recovery.

Sizing the Flash Recovery Area

An approach for sizing the flash recovery area is described at length in *Oracle Database Backup and Recovery Basics*, but as a general rule, the larger the flash recovery area, the more useful it becomes. Ideally, the flash recovery area should be large enough to hold a copy of all of your datafiles and control files, the online redo logs, and the archived redo log files needed to recover your database using the datafile backups kept under your retention policy. If your backup strategy includes incremental backups, described in "Incremental Backups of Datafiles" on page 9-10, add enough space to the flash recovery area to accommodate these files as well. If you can move some backups to tape, you can reduce the size of the flash recovery area somewhat, although retrieving those files from tape will cause longer database restore and recovery times.

Credentials for Performing Oracle Enterprise Manager Backup and Recovery

You must have the proper credentials to perform some of the configuration tasks for backup and recovery, and to schedule backup jobs and perform recovery. Two sets of credentials are required: the Oracle user you use when you log in to Enterprise Manager, and the host operating system user whose credentials you provide when performing backup and recovery tasks. The host operating system credentials are used when RMAN is started behind the scenes, to actually perform the tasks you specified or scheduled through Enterprise Manager.

If you log in to Enterprise Manager with SYSDBA privileges, any valid host operating system user who has execute permission for the RMAN command line client will suffice for scheduling and running RMAN tasks.

For those tasks where host operating system credentials are required, a Host Credentials form where they can be entered appears at the bottom of the page used to perform the task. This form always includes a checkbox labelled **Save as Preferred Credential**. If you check this box before performing your action, the credentials you provide will be stored persistently for the currently logged-in Oracle user, and re-used whenever you log in as that user and perform operations requiring host credentials. (If the database is down, however, as is required for some database recovery operations, you will have to enter host credentials manually.)

Configuring the Flash Recovery Area and ARCHIVELOG Mode

It is possible to configure a flash recovery area and set the database archiving mode when first creating the database. If you did not perform these tasks at database creation time, however, you can add them to an existing database with the procedure described here.

Configuring the flash recovery area and setting `ARCHIVELOG` mode requires the following steps:

1. Under the host operating system, create a directory to hold the flash recovery area. Make sure that the permissions on this directory allow Oracle to create files here.

2. From the Database Home page, go to the Maintenance page. On this page, under the Backup/Recovery heading, select **Configure Recovery Settings**. The Configure Recovery Settings page appears.

3. Under the Flash Recovery Area heading, enter the path to the flash recovery area location (the path to the directory on disk you created in step 1), and your desired flash recovery area size. Make sure the **Apply changes to SPFile Only** box is not checked, then click **Apply** to make these changes take effect.

4. Under the Media Recovery heading, check **ARCHIVELOG Mode** if it was not already checked. Below the **ARCHIVELOG Mode** checkbox is a list of up to ten possible log archiving locations. Enter `USE_DB_RECOVERY_FILE_DEST` as one of the destinations, to specify the flash recovery area as a destination. For ease of database management, the best practice is to use the flash recovery area as your only redo log archiving destination. You can, however, specify other locations on disk, filling them in starting with the first one.

5. Click **Apply** to make these changes take effect.

> **Note:** If your database was not previously running in `ARCHIVELOG` mode, you will be prompted at this time to restart your database so that the switch to `ARCHIVELOG` mode can take effect. You must also take an offline backup of your whole database immediately after switching your database into `ARCHIVELOG` mode. See "Performing and Scheduling Backups with Enterprise Manager" on page 9-11 for more details on performing an offline database backup.

Configuring Backup Settings

Assuming you have a flash recovery area configured and are running in `ARCHIVELOG` mode, you can configure a number of settings and policies that determine how backups are stored, which data is backed up, how backups perform, and how long backups are retained before being purged from the flash recovery area. There are also settings you can configure to improve backup performance. This section provides

information on concepts underlying the available settings, and information on how to change them through Enterprise Manager.

Understanding Backup Device Settings for Disk

The settings available on the Device property page of Configure Backup Settings affect how backups will be written to disk and to tape. For disk-based backups, you can configure the default format for storing backups, the location on disk where backups are stored, and whether backup tasks are run in parallel for improved performance.

Backup File Types: Image Copies, Backup Pieces, and Backup Sets Database backups created by RMAN can be stored in one of two forms: image copies or backup sets.

Image copies are exact byte-for-byte copies of the files they back up. You can create an image copy by copying a file at the host operating system level. However, unlike copying files at the operating system level, creating image copy backups through Recovery Manager or Enterprise Manager causes a record of those image copies to be made in the RMAN repository. This allows RMAN to use these copies during database restore and recovery. RMAN can only use files in restore operations if they are recorded in the RMAN repository.

Backup sets are logical entities containing the results of running various backup tasks. In fact, a backup set consists of several physical files called backup pieces. Backup pieces use an Oracle-proprietary format to store the contents of one or several files from a backup job. Backup pieces cannot be meaningfully manipulated individually; they can only be accessed as part of backup sets. When datafiles are backed up into backup sets, data blocks that have not yet been used to store data are not written into the backup pieces, saving space. This is referred to as unused block compression. Unused block compression is fundamental to how backup sets are created, and cannot be disabled.

An additional binary data compression algorithm can be applied during the writing of backup sets to save more space, at some cost in performance. Using binary compression for backup sets stored on disk is recommended if conserving disk space is more important than backup speed.

> **Note:** Use of binary compression of backup sets is supported for backup sets written to a media manager, but if the media manager supports its own binary compression, then Oracle Corporation recommends using the media manager's compression instead.

Oracle backups to disk can be saved as image copies or backup sets. Image copy backups can only be created on disk. Backups to tape and similar media management devices must be stored as backup sets.

Parallelism and RMAN Backups RMAN depends upon server sessions, processes that run on the database server, to actually perform backup and restore tasks. Each server session in turn corresponds to an RMAN channel, representing one stream of data to or from a backup device. RMAN supports parallelism, the use of multiple channels and server sessions to carry out the work of one backup or recovery task. If a task can be usefully executed in parallel on your hardware, you can set the level of parallelism to use on that task. You can also set a default level of parallelism to use when working with specific devices. Proper exploitation of parallelism can greatly increase performance on backup and recovery tasks.

For more conceptual material on channels and parallelism in RMAN backups (whether through Enterprise Manager or the command line client) see *Oracle Database Backup and Recovery Advanced User's Guide*.

Configuring Backup Device Settings for Disk From the Database Home page, go to the Maintenance property page, and then, under the Backup/Recovery heading, choose **Configure Backup Settings**.

The Configure Backup Settings page contains three property pages: Device, Backup Set and Policy.

The settings you pick here are defaults which can apply to all backup jobs. When performing individual backup tasks, you can override these defaults.

The Device property page is shown first by default. Review the following fields under the Disk Settings section:

- **Parallelism**: For now, set this value to 1. Later, when you have had time to review the information in *Oracle Database Backup and Recovery Advanced User's Guide* on parallelism and performance in RMAN, you may want to change this value.

- **Disk Backup Location**: Should be blank, to direct backups to the flash recovery area.

- **Disk Backup Type**: Make sure that Backup Set is selected. One advantage to backing up Oracle datafiles to backup sets is that RMAN uses unused block compression to save space in backing up datafiles. Only those blocks in your datafiles that have been used to store data are included in the backup set.

You can also provide the host credentials for the backup. Enter a username and password for a host operating system account that is a member of the DBA group.

After these settings are filled in, you can click **Test Disk Backup** to make sure the credentials and backup location are correct.

The settings on the Backup Set property page should not be altered at this time.

Configuring Backup Policy Settings

From the Backup Settings page, choose the **Policy** property page. On this page, you can set the backup policies governing control file and SPFILE backups, tablespaces to exclude from whole database backup, and the backup retention policy.

Configuring Backup Policies For now, check the box to **Automatically back up the control file and server parameter file (SPFILE) with every backup and database structural change**. The SPFIle and control file are critical to the operation of your database and RMAN, and are also relatively small compared to typical datafiles. Backing them up frequently imposes relatively little disk space overhead. Leave the **Autobackup Disk Location** field blank, so that the autobackups are sent to the flash recovery area.

Check the option to **Optimize the whole database backup by skipping unchanged files such as read-only and offline datafiles that have been backed up**. This will save space in the flash recovery area.

Check the **Enable block change tracking for faster incremental backups** box to take advantage of the block change tracking feature of Oracle, which substantially improves performance of incremental backups at a small cost of overhead during normal operations.

Configuring Exclusions from Backup   You can list tablespaces to exclude from backup. If, for example, you have a read-only tablespace, you do not need to back it up. For now, make sure this list is empty, so that all tablespaces are backed up.

Configuring Backup Retention Policy You can choose between three forms of retention policy:

- **Retain All Backups** (in effect, no retention policy at all, so that all backups are kept in the flash recovery area until you explicitly delete them)

- **Retain backups that are necessary for a recovery to any time within the specified number of days (point-in-time recovery)** for a recovery window-based retention policy

- **Retain at least the specified number of full backups for each datafile** for a redundancy-based retention policy

 For now, choose the recovery window-based retention policy, with a recovery window of 31 days.

Verify that the **Host Credentials** section at the bottom of the page contains proper credentials. Then click **OK** to save the new settings.

Recording Your DBID and DB_UNIQUE_NAME

If you lose your database control file or SPFILE, Enterprise Manager can restore them from backup, as long as you can provide the DB_UNIQUE_NAME and DBID for your database.

To find out your DB_UNIQUE_NAME, from the database home page, from the Administration page, under Instance, click **All Initialization Parameters**. On the Current property page, for **Filter** enter DB_UNIQUE_NAME and click **Go**. The resulting page displays a row with DB_UNIQUE_NAME listed in the Name column, and the DB_ UNIQUE_NAME value for your database in the Value column.

To find out your DBID, from the database home page, select the Administration page, then under Storage, click **Controlfiles**, then select the Advanced property page. The Database ID field contains the value of the DBID.

Record these two pieces of information offline in case they are needed for disaster recovery.

Backing Up Your Database

This section discusses performing database backup using Enterprise Manager. It introduces several types of Oracle database backup, then explains how to take the different backup types, how to take advantage of Enterprise Manager's Oracle-suggested backup strategy to implement a useful basic backup regimen permitting fast recovery, and how to schedule your own backups.

> **Note:**   The Oracle-suggested strategy for disk-only backups, as described in this section, provides efficient daily backup of your entire database to disk. This provides the ability to quickly return your database to its state at any point during the preceding 24 hours. If you need more flexible backup options than this, you will find more details on the full range of backup options in *Oracle Database Backup and Recovery Basics*.

Database Backup Concepts

To understand the Oracle-suggested backup strategy and other backup types provided through Enterprise Manager, you need some conceptual background on database backups as supported by Oracle.

Full Backups of Datafiles

A full backup of a datafile is a backup which includes all used blocks of the datafile. This can be either an image copy backup, which is an exact copy of the datafile as if copied with a host operating system file copy command, or a backup into a backup set created by RMAN. Regardless of the form in which the backup is stored, the entire datafile is backed up, even if only a few blocks have changed.

Incremental Backups of Datafiles

Incremental backups are based around capturing only those blocks that change between backups in each datafile of your database. In a typical incremental backup strategy, a level 0 incremental backup, capturing all blocks in the datafile, is taken as a starting point. Subsequent level 1 incremental backups, typically taken at regular intervals, capture images of each block in a datafile that changed. Level 1 backups can be cumulative, in which case all blocks changed since the most recent level 0 backup are included, or differential, in which case only those blocks changed since the most recent level 0 or level 1 incremental backup are included.

Recovering changed blocks from incremental backups is used to improve performance of the media recovery process. Since an incremental level 1 backup captures the final contents of all datafile blocks changed during the period covered by the incremental, the recovery process can skip reapplying individual updates from the redo logs of that period and simply update each block with its final contents. The redo logs are only used for the period not covered by level 1 incremental backups.

Incrementally Updated Backups: Rolling Forward Image Copies of Datafiles

The **incrementally updated backups** feature of Oracle lets you use one or more level 1 incremental backups with an older image copy backup of your datafiles, to roll the copy forward to the SCN at which the last level 1 incremental backup was taken. All blocks changed since the image copy was created are overwritten with their new contents as of the time of the last level 1 incremental backup. The effect is to roll the file forward in time, so that its contents are equivalent to an image copy full datafile backup taken at the time of the last incremental level 1 backup. This feature lets you implement strategies with shorter recovery times, because you need only perform recovery starting at the SCN of the last level 1 incremental applied to your datafiles.

Tagging Backups

All RMAN backups, including incremental backups, can be tagged with a label, a text string identifying that backup, either uniquely or as part of a group of backups. (For instance, if you performed a weekly full database backup on Saturday nights, you could use the tag FULL_SATURDAY to identify all such backups.) These tags can be used for referring to specific backups in RMAN commands; for example, you could issue a command to move the latest FULL_SATURDAY backup to tape.

Because you can use tags to refer to different groups of backups, they are useful if you want to create several different routines in your overall backup strategy which do not interfere with each other.

When you schedule a backup job and give the job a name, the job name is used to tag the backup.

Performing and Scheduling Backups with Enterprise Manager

Enterprise Manager lets you take backups of all types supported by Oracle. It also lets you schedule backups for use in a backup strategy.

Performing a Whole Database Backup with Oracle Enterprise Manager

Whole backups of a database are based on backing up the entire contents of the database at the time of backup. Full backups of all datafiles are created. The results may be stored as image copies or as backup sets, but in either case the complete contents of all datafiles of the database are represented in the backup, as well as the control file, archived redo log and server parameter file. With this set of files, the database can be recovered completely.
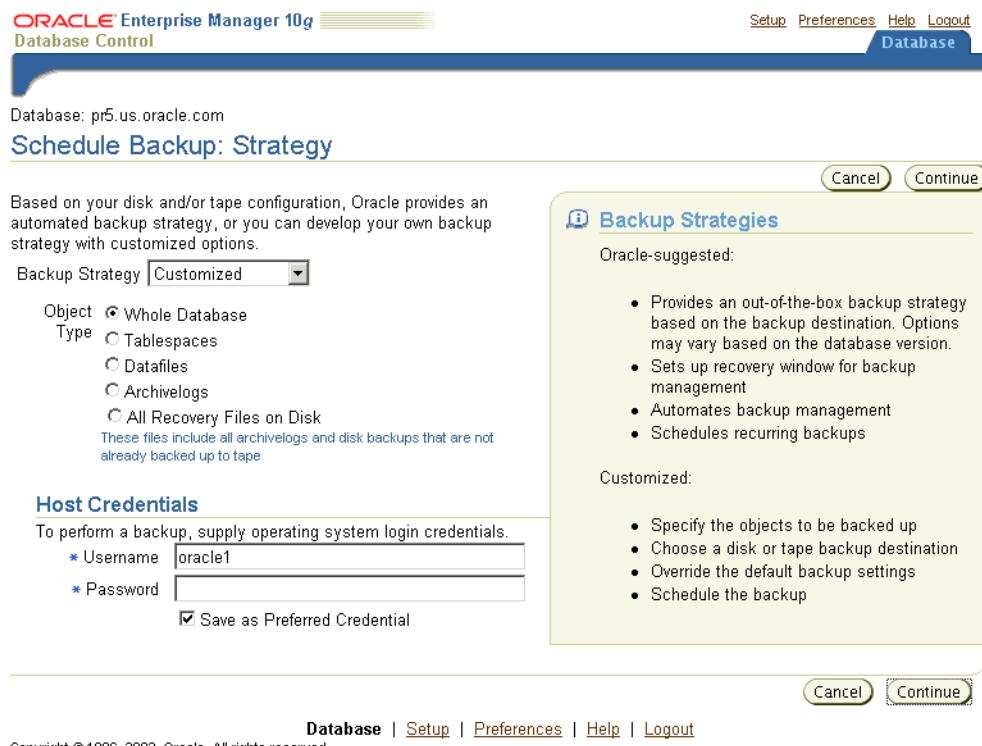
While whole database backups can be an important element in your overall backup strategy, they are also a required step in some situations, such as when you switch ARCHIVELOG mode on or off.

> **Note:** If you had to switch your database into ARCHIVELOG mode as part of initial configuration for backup and recovery, you *must* shut the database down and take an offline whole database backup *immediately* after making the switch, to use as the basis for future recoveries. Do not allow users to perform transactions against the database until you have completed the backup. Because you cannot apply archived redo logs from after the switch to any backups created before the switch, you are effectively running without a backup if you do not create an offline whole database backup as soon as you switch into ARCHIVELOG mode.

Go to the Maintenance property page and under the Backup/Recovery column, choose **Schedule Backup**. The Schedule Backup: Strategy page shown in Figure 9–1, "Schedule Backup: Strategy Page" appears:

Figure 9–1 Schedule Backup: Strategy Page

The first field is a **Backup Strategy** field, where you can choose between **Oracle-suggested** (the default) and **Customized**. For a whole database backup, choose **Customized**. The page refreshes itself, to offer a choice of objects to back up and a space to enter host credentials. Select **Whole Database** as the set of objects to back up. Enter your host operating system credentials if required, and click **Continue**.

On the Schedule Backup: Options page, you specify the options for this whole database backup. Under **Backup Type**, choose **Full Backup**. For **Backup Mode**, choose between an online and offline backup. Typically, you will want to perform online backups, to maximize database availability. (Note that, as explained in "Configuring Your Database for Basic Backup and Recovery" on page 9-4, the database must be run in ARCHIVELOG mode to enable online backups.)

Under the **Advanced** section, if this is an online backup, check the box to back up all archived logs. (There is no need to back up archived logs when performing an offline backup, as the database will be in a consistent state at the time of backup and does not require media recovery if you restore from this backup.) For now, leave the option for proxy copies unchecked. Do not specify a maximum number of files for each backup set. Click **Next** to move on to the Backup: Settings page.

On the Schedule Backup: Settings page, select the appropriate backup destination. Oracle Corporation recommends backing up to disk, to minimize recovery time. Then, click **Next** to move on to the Schedule Backup: Schedule page.

On the Schedule Backup: Schedule page, enter the schedule for the backup job, or specify to perform it immediately. A **Job Name** is generated for you, but if you want to tag this backup, you can edit the assigned job name, which is used as the tag. You can also edit the **Job Description** as needed for your own reference. Then, in the **Schedule** fields, you specify when to start the backup, and how often to repeat it. Leave the default start time of **Immediately** selected to run a backup immediately, or set **Later** and enter a later time. For jobs to be run regularly, set the **Repeat** and **Repeat Until** parameters as appropriate. (You probably want to change the **Job Name** in this case, to identify these scheduled backups as part of an ongoing series, such as WEEKLY_FULL_BACKUP. When finished, click **Next** to move on to the Schedule Backup: Review page.

On the Schedule Backup: Review page, you have one last chance to use the **Back** button and change these options.You can also click **Edit RMAN Script** to see the RMAN commands that will be executed to perform your specified backup jobs. Click **Submit Job** to add the specified backup job to the schedule (or to run it immediately).

To monitor the progress of the backup job, you can click **View Job**.You will arrive on a page that shows a summary of the job submitted, and a Logs section where you can follow the progress of the various steps of the backup job and drill down to see the RMAN output from the running job.

Performing Offline Database Backups

When performing an offline backup, the database instance shuts down, then restarts and enters a MOUNTED state for the duration of the offline backup. The offline backup runs in the background, generating no user-visible output in the browser. The fact that the database is not open affects the pages you see from Enterprise Manager while the offline backup runs.

After you submit the backup job, the Backup Submit Successful page appears, and suggests that you wait long enough for the offline backup to complete, then click View Job to be redirected to the Job Status page.

In this situation, because you know that the database is down in order to run its offline backup, do not choose either option. Wait long enough for the offline backup to complete, then reload the browser page again. At that time you will be prompted for database login credentials, and then returned to the job status page, where you can view the final results of the offline backup job.

If you attempt to reload the status page while the database is shutting down and coming back up to a MOUNT state, you may encounter an error page in your web browser. Wait a short period and refresh the page again. This error does not indicate a problem with the offline backup. Enterprise Manager must be restarted when the database is shut down and brought to a MOUNT state. While Enterprise Manager is being restarted, you may receive this error page.

If you attempt to reload the status page while the offline backup is in progress, you may encounter a page with a message that reads: *The database status is currently unavailable. It is possible that the database is in mount or nomount state.* Your choices on this page are to click **Startup** or **Perform Recovery**. Do not click either button. The offline backup is still running in the background and should not be interrupted. Instead, reload this page periodically. After the offline backup completes and the database opens, reloading this page will bring you to the database login page again.

Using the Oracle-Suggested Backup Strategy

Enterprise Manager makes it easy to set up an Oracle-suggested backup strategy for backups to disk, that protects your data and provides efficient recoverability to any

point in the a recovery window of your choosing. (In the simplest case, examined in this section, this window is 24 hours.) The Oracle-suggested strategy leverages Oracle's incremental backup and incrementally-updated backup features to provide faster backups than whole database backups, and faster recoverability than is possible through applying database changes from the archived log to your datafiles.

The Oracle-suggested backup strategy is based on creating an image copy of your database which is rolled forward using incrementally updated backups. Oracle Enterprise Manager schedules RMAN backups jobs for you to run during the overnight hours.

For each datafile, the strategy calls for backups to be made as follows:

- At the beginning of day 1 of the strategy (the time the first scheduled job actually runs), an incremental level 0 datafile copy backup. It contains the datafile's contents at the beginning of day 1.

  In a restore-and-recovery scenario, the redo logs from day 1 can be used to recover to any point during day 1.

- At the beginning of day 2, an incremental level 1 backup is created, containing the blocks changed during day 1.

  In a restore-and-recovery scenario, this incremental level 1 can be applied to quickly roll forward the level 0 backup to the beginning of day 2, and redo logs can be used to recover to any point during day 2.

- At the beginning of each day n for days 3 and onwards, the level 1 backup from the beginning of day n-1 is applied to the level 0 backup. This brings the datafile copy to its state at the beginning of day n-1. Then, a new level 1 is created, containing the blocks changed during day n-1.

  In a restore-and-recovery scenario, this incremental level 1 can be applied to the datafile rolled forward on day n-1 to the beginning of day n, and redo logs can be used to recover the database to any point during day n.

The datafile copies used in the Oracle-suggested backup strategy are tagged with the tag ORA$OEM_LEVEL_0. The level 1 incremental backups for use in this strategy are created for use with datafile copies that are so labelled. You can safely implement other backup strategies without concern for interference from the backups for the Oracle suggested strategy.

There are also Oracle-suggested strategies that use tape backups along with disk backups, but those are beyond the scope of this document.

Backing Up Your Database With the Oracle-Suggested Disk Backup Strategy Go to the Maintenance property page and under the Recovery column, choose **Schedule Backup**. The Schedule Backup: Strategy page appears.

The **Backup Strategy** field lets you select between **Oracle-suggested** (the default) and **Customized** backup strategies, as well as backup destinations of disk, tape or both. For now, choose **Oracle-suggested** as a strategy and **Disk** as the destination. Choose **Continue**. The Schedule Backup: Setup page appears describing the setup. There are, however, no settings to change on this page. Click **Next** to continue to the Schedule Backup: Schedule page.

On the Schedule Backup: Schedule page, you are prompted for **Start Date**, **Time Zone** and **Daily Backup Time** for the daily backups. Based upon your expected usage patterns, choose times for the nightly backup during which database activity is low. Choose **Next** to continue to the Schedule Backup: Review page.

On the Schedule Backup: Review page, the backup script RMAN will run is displayed (although you cannot edit the script directly), and you are presented with a chance to confirm or alter your settings. In the listing you can see the tag `ORA$OEM_LEVEL_0` assigned to the backup. Assuming you do not need to change the schedule assigned, click **Submit Job** to add the job for the Oracle-suggested strategy to your schedule.

Your database will now be backed up once daily, using incremental backups and incrementally applied backups, allowing quick recovery to any time in the preceding 24 hours.

Scheduling Other Backup Tasks

After taking some time to understand the full range of available backup options as described in *Oracle Database Backup and Recovery Basics*, you may decide to schedule backup tasks beyond those used to implement the Oracle-suggested backup strategy. While the particulars of specifying the job to perform differ for each type of backup, all backups, whether performed immediately or scheduled, begin from the Schedule Backup: Strategy page, shown in "Performing and Scheduling Backups with Enterprise Manager" on page 9-11. Here you can select any of several object types to back up: the whole database, individual tablespaces, datafiles, or archived logs. (You can also back up existing backups in one destination to some other destination, such as moving backups from disk to tape.) For any object you select to back up, there will be options governing how that backup is taken. These will, however, differ by the type of object being backed up.

Then, you arrive at the Schedule Backup: Schedule page. You can either schedule a job to run once, immediately or at a future time, or set it to run multiple times, using the options in this form.

If you schedule a regular backup task as part of a strategy of your own devising, it is recommended that you give the resulting backup job a distinctive name, such as WEEKLY_FULL_BACKUP or DAILY_SALES. All backups produced by this strategy will have this job name appended with a timestamp as their tag. The common tag will permit you to more easily refer to the backups from this strategy during recovery activities.

Validating Backups

A backup strategy is not complete until you validate that the backups produced can actually be used to recover the database. The RMAN command line client provides the RESTORE... VALIDATE command, which lets you confirm that a particular restore operation can be performed with the backups you have available.

For example, if the RMAN command RESTORE DATABASE restores your entire database from backup, RESTORE DATABASE VALIDATE examines the available backups and redo logs to determine whether the RESTORE DATABASE operation would complete successfully given the currently available set of backups. See *Oracle Database Backup and Recovery Basics* for more details about validating backups.

Restore and Recovery Operations

Restoring and recovering your whole database from a backup is completely automated, thanks to the record of your existing backups kept in the RMAN repository and the intelligence built into Recovery Manager. As long as you have taken your backups through Recovery Manager (whether at the command line or through Enterprise Manager), the entire restore and media recovery operation can be performed through the Enterprise Manager.

To access the restore and recovery tasks, from the database home page, choose the **Maintenance** property page, then under the heading Backup/Recovery choose **Perform Recovery** to start the Recovery wizard.

On the Perform Recovery: Type page, in the **Type** field, select the type of restore operation to perform. You can choose to recover a whole database or selected tablespaces, datafiles, archived logs, or tables.

Note: In some recovery scenarios, such as a complete restore and recovery of your database, the database state will be altered by steps you take during the wizard. Changes, some of them irrevocable, are made to your database at certain steps. For example, the database may be shut down and brought to MOUNTED state, or datafiles may be overwritten with versions from backup.

Oracle Enterprise Manager will display warnings each time a significant database change will result from pressing **Continue** during the recovery process. Pay close attention to these warnings.

Recovering a Whole Database from Backup

Note: The discussion which follows assumes that you are restoring and recovering your database after the loss of one or more datafiles, but still have a usable SPFILE and control file. Enterprise Manager can also be used to restore a lost SPFILE or control file. See "Recovering from a Lost SPFILE or Control File" on page 9-17 for details.

On the Perform Recovery: Type page, set the **Type** to **Whole Database**. The **Operation Type** drop-down menu will update to offer you the choices appropriate to a whole database restore: restoring files only, recovery only, and both restoring and recovering your database. Select **Restore and Recover**. Also, provide the requested host credentials at this time if necessary and click **Continue**.

At this point, Oracle shuts down your database, and you arrive at a Database page where you are given the opportunity to **Startup** or **Perform Recovery**. Choose **Perform Recovery**.

Note: As already noted in the discussion of offline backups in "Performing a Whole Database Backup with Oracle Enterprise Manager" on page 9-11, Enterprise Manager may respond with an error page during the period when the database is being shut down and brought to the MOUNTED state. Wait a while and refresh the page until the error no longer occurs.

After the database reaches the MOUNTED state, Enterprise Manager responds with a page warning that the state of the database is unavailable (meaning that Enterprise Manager cannot determine the state), but that the database may be in a NOMOUNT or MOUNTED state. At this time you will be offered two choices: **Startup** and **Perform Recovery**. Because you know that the database is down so that you can perform your recovery, do not choose **Startup**. Click **Perform Recovery** to resume your recovery session. You may be prompted for host and database credentials. Then you will reach

the Perform Recovery: Type page again, only now the database is in a MOUNTED state (as is required for this operation) instead of being open. At this point, as you did before, select **Whole Database** for Object Type and **Restore and Recover** for Operation, and click Continue.

Next, you see the Perform Recovery: Credentials page, where you are again prompted for operating system-level and Oracle database credentials. Provide these, and click **Continue** to begin the recovery process.

In the first step of performing whole database recovery, Perform Recovery:Point in Time, you specify whether to recover all transactions to your database as of the present time (which is called complete recovery), or only transactions up through some point in time in the past (which is called point-in-time recovery). Point-in-time recovery is an advanced recovery technique that lets you return the database to its state before some unwanted major change. For more details about point-in-time recovery, see *Oracle Database Backup and Recovery Advanced User's Guide*. For this example, however, recover the database to the current point in time. Click **Continue** after making your selection.

In the next step, **Perform Recovery: Rename**, you can rename the files being restored from backup. You can specify a new path for the files, restoring them to a different directory. For this example, however, restore the files to the default location, which is its location before the restore operation. Click **Next** to continue.

Finally, on the Perform Recovery: Review page, you can review the options you chose, and view the RMAN script that will be run to carry out your requested restore and recovery action. Click **Submit** to start the recovery process.

Recovering from a Lost SPFILE or Control File

The process for recovering a database with a lost SPFILE or control file is similar to recovery of all datafiles. If you have lost a control file, your database is definitely down. On attempting to start an Enterprise Manager session to your database, you will encounter the warning that the database state is unavailable, and you must choose between **Startup** and **Perform Recovery**. Because you do not know why your database is down, click **Startup**. Enterprise Manager attempts to open the database. In the process, it will detect a lost or unusable SPFILE or control file. If the startup fails, choose **Perform Recovery** and Enterprise Manager will prompt you to restore the lost files. If you are not using a recovery catalog, you will have to restore these files from autobackup. For this process, you will need the DBID and DB_UNIQUE_NAME you recorded in You may also need to provide Enterprise Manager the location where you directed control file autobackups, which can be the location of the flash recovery area or some other location on disk of your choosing.

Returning a Table to a Past State: Flashback Table

Oracle Flashback Table lets you revert one or more tables back to their contents at a previous time, without affecting other objects in your database. This recovery technique lets you recover from logical data corruptions, such as erroneously inserting rows into a table or deleting data from a table. Flashback Table lets you return tables you select to their state at a past point in time, without undoing desired changes to the other objects in your database, as would be required by a point-in-time recovery of the entire database. Also, unlike point-in-time recovery, your database remains available during the operation.

For this example, we will perform Flashback Table on the EMPLOYEES table in the HR schema. Assume that an erroneous update shortly after 23 October 2004, 15:30:00

has changed the LASTNAME column for all employees to an empty string and you need to return the original LASTNAME values to the table.

Before you can perform Flashback Table, you must ensure that row movement is enabled on the table to be flashed back.

Enabling Row Movement on a Table

To enable row movement, or if you do not know whether row movement is enabled, follow these steps:

1. From the database home page, choose the Administration page. Then, under Schema, click **Tables** to administer tables. To find the target table for flashback table, you can enter one or both of the schema name in the **Type** field and the table name in the **Object Name** field. Then click **OK** to search for the table. You may need to page through the search results to find your table.

2. After you find your table in the schema, click the name of the table in the Table Name column of the search results. Enterprise Manager displays a page with several property pages for administering different aspects of the table. Select the Options property page.

3. Make sure **Enable Row Movement** is set to **Yes**, and click **Apply** to update the options for the table. When the page has refreshed, you can click **Tables** in the breadcrumb at the top of the page to return to the search results, and enable row movement on more tables by repeating these steps for each table.

Performing Flashback Table

To perform the Flashback Table operation, use the following steps:

1. From the Maintenance property page of the database home page, under the Backup/Recovery heading, select **Perform Recovery**, and on the Perform Recovery: Type page, select **Tables** for the object type. Choose the **Flashback Existing Tables** option and click **Continue**.

2. On the Perform Recovery: Point-in-time page, you must choose the target time for your Flashback Table operation. If you do not know the time at which the unwanted changes occurred, you can investigate the history of transactions affecting this table by choosing **Evaluate row changes and transactions to decide upon a point in time**. A feature called Oracle Flashback Versions Query lets you review all recent changes to the target table. However, the use of this feature is beyond the scope of this manual.

   For this example, assume that the time of the corruption is known to be Oct. 3, 2004, 11:53AM. In the form offered, select **Flashback to a timestamp**, and enter your target time. Click **Next** to continue with the Flashback Table process.

3. Now, on the Perform Recovery: Flashback Tables page, specify the target tables for Flashback Table, by entering table names (one on each line) in the **Tables to Flashback** text box. You can also click **Add Tables** and search for more tables to add. For this example, manually enter the HR.EMPLOYEES table in the **Tables to Flashback** text box. Click **Next** to continue with the Flashback Table process.

4. If your table has other dependent tables, you next see the Dependency Options page, where you are asked how they should be handled. Your choices are **Cascade** (flashing back any dependent tables), **Restrict** (flashing back only the target table), or **Customize** (selecting which dependent tables to flashback and which to leave as they are). You can click **Show Dependencies** to see which tables will be affected. How you proceed at this point will depend upon your application.

HR.EMPLOYEES has dependent tables HR.JOBS and HR.DEPARTMENTS. For this example, assume that it is safe to cascade any changes, flashing back those two tables as well as the HR.EMPLOYEES table. Note that row movement must be enabled on all affected tables, not just the initial target table. Click **Next** to continue. The next page to appear is the Perform Recovery: Review page, where you have a chance to confirm the Flashback Table operation to be performed. The target timestamp and tables to be flashed back are displayed. Click **Submit** to perform the actual Flashback Table operation. When the operation is completed, a Confirmation page appears announcing the results. Click **OK** to return to the database home page.

Recovering Dropped Tables: Flashback Drop

Oracle Flashback Drop lets you reverse the effects of dropping a table, returning the dropped table to the database along with its dependent objects such as indexes and triggers. It works by storing dropped objects in a Recycle Bin, from which they may be retrieved until the Recycle Bin is purged, either explicitly or because space is needed for new database objects.

As with Flashback Table, Flashback Drop can be used while the rest of your database remains open, and without undoing desired changes in objects not affected by the Flashback Drop operation. It is more convenient than forms of recovery that require taking the database offline and restoring files from backup.

> **Note:** For a table to be recoverable using Flashback Drop, it must reside in a locally managed tablespace. Also, tables in the SYSTEM tablespaces cannot recovered using Flashback Drop regardless of the tablespace type.

Follow these steps to perform Flashback Drop:

1. From the Maintenance property page of the database home page, under the Backup/Recovery heading, select **Perform Recovery**, and on the Perform Recovery: Type page, select **Tables** for the object type. For the **Operation Type**, choose **Flashback dropped tables**. Then click **Continue**. This takes you to the Perform Recovery: Dropped Objects Selection page.

2. On the Perform Recovery: Dropped Objects Selection page, you can gain access to the objects in the Recycle Bin. The Search form lets you search among the dropped objects in the Recycle Bin for the objects you want to recover. Provide values for one or both of the **Schema Name** and **Table** fields, and click **Go** to search.

   When the page refreshes, the Results section lists the objects matching your search. If you only see the Recycle Bin listed, click the arrow next to the Recycle Bin to expand its contents by one level, showing dropped tables matching your search but not their dependent objects. You can also click **Expand All** to see all objects in the Recycle Bin, including both dropped tables and dependent objects such as indexes and triggers. For each table listed, you can click **View Content** in the Operation column, to see its contents.

   To select one or more tables for Flashback Drop, click the checkbox next to each table. (When a table is retrieved from the Recycle Bin, all of the dependent objects for the table that are in the recycle bin are brought back as well. They cannot be retrieved separately.) When you have selected all of your objects to restore, click **Next**.

3. On the Perform Recovery: Rename page, you have the opportunity to specify new names for any dropped objects you are returning to your database. The primary reason for renaming objects being retrieved from the recycle bin is if you have created new tables with the same names as tables being retrieved. Enter new names as needed in the **New Name** field in the list of tables being flashed back. Then click **Next** to continue.

4. On the Perform Recovery: Review page, you are offered an impact analysis, showing the full set of objects to be flashed back, including the dependent objects, as well as the names they will have when the Flashback Drop operation is complete. If you are satisfied with the changes listed, click **Submit** to perform the Flashback Drop.

When the process is complete, you arrive at a confirmation page indicating the success of the operation. Click **OK** to return to the database home page.

Managing Your Backups

Managing backups consists of two things: managing the backups themselves as they exist on disk or tape, and managing the record of backups kept in the RMAN repository. A backup recorded in the RMAN repository can be in one of three states:

- Available, meaning that the backup is still present on disk or tape, as recorded in the repository;

- Expired, meaning that the backup has been deleted from disk or tape but is still listed in the repository;

- Unavailable, meaning that the backup is temporarily not available for data recovery operations (because, for example, it is stored on a tape that is stored off-site or a disk that is not mounted at the moment).

Backups can also be obsolete or not. An obsolete backup is one which, based on the currently configured retention policy, is no longer needed to satisfy data recovery goals.

Backups that no longer exist (because they were deleted from disk, because a tape was lost, and so on) must be removed from the RMAN repository. Otherwise, RMAN may discover in the middle of a database recovery operation that a needed backup is not in fact available.

Datafiles or archived redo logs copied at the operating system level can be cataloged, which adds them to the RMAN repository so that they can be used in data restore and recovery operations just as if they had been created with RMAN.

Backup maintenance functions provided through Enterprise Manager includes the following:

- Viewing lists of backups (backup sets and image copies) recorded in the RMAN repository

- Crosschecking your repository, which checks whether backups listed in the repository exist and are accessible, and marks as expired any backups not accessible at the time of the crosscheck;

- Deleting the record of expired backups from your RMAN repository;

- Deleting obsolete backups from the repository and from disk.

Note that if you use a flash recovery area for your backup storage, many maintenance activities are reduced or eliminated because of the flash recovery area's automatic

management of disk space and which files are needed according to the retention policy.

To access backup management functions, from the database home page, choose the Maintenance property page, and then in the Backup/Recovery category, click **Manage Current Backups**.

Using the Manage Current Backups Page in Enterprise Manager

The Manage Current Backups page has two property pages you can choose: **Backup Set** (the initial view) and **Image Copy**. Each serves a similar purpose, listing the backups stored as backup sets or image copies, according to the record in the Recovery Manager repository. Figure 9–2, "Manage Current Backups Page" shows a typical view of the Backup Set page:

Figure 9–2   Manage Current Backups Page

Backup sets are identified by their tags and completion times. To view information about which files are backed up in a backup set, drill down by clicking the value in the

Contents column. For information about the individual backup pieces in a backup set, click the number of pieces in the **Pieces** column. Individual pieces will be listed by filename.

You can filter the list of backup sets by their contents, their status, and completion time. This makes it easier, for example, to determine what backups you have available for your control file and SPFILE.

You can also crosscheck or delete individual backups, or mark individual backups as unavailable if you know that they are temporarily not accessible by RMAN. Simply click the **Select** checkbox next to the file, and click the appropriate action button at the top of the Results list.

The Image Copy property page presents similar functionality to the Backup Sets property page. The focus in this discussion will be on commands in the Backup Sets property page, where they are substantially similar to those for image copies.

Like the backup and restore commands, the commands to crosscheck, delete and change the status of backups are ultimately RMAN commands and are submitted as RMAN jobs. Upon selecting these tasks, you will go to a page where you schedule and submit the RMAN job for execution, just as you would schedule a backup job. In some cases executing a maintenance task, such as crosschecking files stored on tape, may take considerable time. Depending on the task, scheduling tasks for later execution may be preferable to running them immediately and waiting. Also, some tasks, such as periodic crosschecks of your backups, should be among the regularly scheduled components of your backup strategy, as a way of being certain that your records of your backups in the RMAN repository are correct. Without that record, RMAN's recovery capabilities are severely limited.

Crosschecking Backups

Crosschecking a backup causes RMAN to verify that the actual physical status of the backup matches the record of the backup in the RMAN repository. For example, if a backup on disk has been deleted with an operating system command and is therefore no longer available for use in restore operations, then crosschecking that file detects this condition. After the crosscheck operation, the RMAN repository correctly reflects the state of the backups on disk or tape.

Backups to disk are marked AVAILABLE if they are still present on disk in the location listed in the RMAN repository, and if they have no corruption in the file header. Backups on tape are listed as AVAILABLE if they are still found on tape (though the file headers are not checked for corruption). Backups that are missing or corrupt are marked EXPIRED.

Click **Crosscheck All** at the top of the page to crosscheck all files in the RMAN repository. Note that, especially in installations where tape backups are in use, crosschecking may take a long time, because it requires access to all storage media containing backups listed in the repository. Because it can take a long time, crosschecking all files is handled as a scheduled job. When you click **Crosscheck All** you arrive at the Crosscheck All: Specify Job Parameters page, from which you can schedule regular crosschecks or run one immediately.

You can also crosscheck individual files, by clicking the **Select** checkbox next to them in the Results list and clicking **Crosscheck** at the top of the Results list. Unlike **Crosscheck All**, the crosscheck of individual files is performed immediately rather than scheduled.

Deleting Expired Backups

Deleting expired backups removes from the RMAN repository those backups which are marked EXPIRED, that is, those which RMAN found to be inaccessible during a crosscheck operation. (No attempt is made to delete the files containing the backup from disk or tape; this command updates only the RMAN repository.)

To delete expired backups, click **Delete All Expired** at the top of the Manage Current Backups page. Note that this will delete both expired backup sets and expired image copies from the RMAN repository, regardless of whether you are viewing the Backup Sets or Image Copies property page when you click **Delete All Expired**.

When you select **Delete All Expired**, you arrive at a page titled Delete All Expired: Specify Job Parameters. Along with the usual scheduling options for an RMAN job, you have a checkbox **Perform the operation 'Crosscheck All' before 'Delete All Expired'**. Checking this box will cause the operation to take longer, but by performing the crosscheck operation immediately before deleting expired backups from the repository, RMAN will have the most up-to-date information possible about which backups are expired.

Deleting Obsolete Backups

To delete obsolete backups, that is, those backups no longer needed to meet your retention policy, click **Delete All Obsolete** at the top of the Manage Current Backups page. All obsolete backups (both backup sets and image copies) will be deleted, regardless of whether you clicked **Delete All Obsolete** while viewing the Backup Set or Image Copy property page on the Manage Current Backups page.

When you click **Delete All Obsolete**, you arrive at the Delete All Obsolete: Specify Job Parameters. You can run the deletion job immediately or schedule it as you would a backup job.

Note that if you use a flash recovery area as your sole disk-based backup destination, you will never need to delete obsolete backups from disk. The automatic space management of the flash recovery area will keep files as specified by the backup retention policy, and then only delete them when space is needed.

Marking Backups Unavailable

If you know that some individual backup is unavailable because of a temporary condition, such as a disk drive that is temporarily offline or a tape stored off-site, you can mark the backups stored there as UNAVAILABLE. RMAN will keep these backups in the RMAN repository (and not delete them when you delete expired backups) but will not try to use them in recovery operations. When the backup becomes available again you can change its state back to AVAILABLE.

To mark backups as unavailable, click the **Select** checkbox next to each backup in the Results list of backups, and select **Change to Unavailable**.

 Note that you cannot mark backups stored in the flash recovery area as UNAVAILABLE.

Cataloging Additional Backups

You may have backups in your flash recovery area or in some other destination which are not in your RMAN repository but which could be used by RMAN. For example, you might have made image copies of some or all of your datafiles at the operating system level. RMAN can use such image copies once they are recorded in the repository.

To add backups to the catalog, select **Catalog Additional Files** at the top of the Manage Current Backups page. On the Catalog Additional Files page, you can either select **Catalog all files in the recovery area into the Recovery Manager repository**, or **Catalog files in the specified disk location into the Recovery Manager repository**. In the latter case, you must enter a path to a disk location in the **Starts With** field.

Note: This path is not necessarily a complete directory name. You can enter a complete filename here to catalog a single file, such as /tmp/dfcopy1.df, or, if you have a series of datafile copies /tmp/dfcopy1.df, /tmp/dfcopy2.df, and so on, you could enter /tmp/dfcopy and match them all. Likewise, if you had a series of directories /tmp/backups1, /tmp/backups2, and so on through /tmp/backups9, each of which contained files to be cataloged, you could enter /tmp/backups and all files whose full path name began with /tmp/backups would be cataloged.

Note also that this could also match directories and files you did not intend, such as /tmp/backupslist.txt or /tmp/backups/ignore. Use care in choosing your value for **Starts With**.

When you click **OK**, RMAN immediately catalogs the files you specified.

Backup and Recovery: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter, and includes annotated screen shots.

To view the Backup and Recovery OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/backup/backup.htm
```

10

Monitoring and Tuning the Database

Monitoring the health of a database and ensuring that it performs optimally is an important task for a database administrator. the This chapter discusses the features and functionality included in Oracle Database that make it easy to proactively monitor database health, identify performance problems, and implement any corrective actions.

The following topics are covered:

- Proactive Database Monitoring
- Diagnosing Performance Problems
- Using Advisors to Optimize Database Performance
- Monitoring and Tuning: Oracle by Example Series

Proactive Database Monitoring

The Oracle Database makes it easy to proactively monitor the health and performance of your database. It monitors the vital signs (or metrics) related to database health, analyzes the workload running against the database, and automatically identifies any issues that need your attention as an administrator. The identified issues are either presented as alerts in Enterprise Manager or, if you prefer, can be sent you through email.

This section discusses the following topics:

- Alerts
- Performance Self-Diagnostics: Automatic Database Diagnostics Monitor
- Monitoring General Database State and Workload
- Managing Alerts

Alerts

Alerts help you monitor your database proactively. Most alerts are notifications when particular metrics thresholds are crossed. For each alert, you can set critical and warning threshold values. These threshold values are meant to be boundary values that when crossed indicate that the system is in an undesirable state. For example, when a tablespace becomes 97% full this can be considered undesirable and have Oracle generate a critical alert.

Other alerts correspond to database events such as Snapshot Too Old or Resumable Session suspended. These types of alerts indicate that the event has occurred.

In addition to notification, you can set alerts to perform some action such as running a script. Scripts that shrink tablespace objects can be useful for a Tablespace Usage warning alert.

By default, Oracle enables the following alerts: Table Space Usage (warning at 85% full, critical at 97% full), Snapshot Too Old, Recovery Area Low on Free Space, and Resumable Session Suspended. You can modify these alerts or enable others by setting their metrics.

Performance Self-Diagnostics: Automatic Database Diagnostics Monitor

Oracle Database includes a self-diagnostic engine called the Automatic Database Diagnostic Monitor (ADDM). ADDM makes it possible for the Oracle Database to diagnose its own performance and determine how any identified problems can be resolved.

To facilitate automatic performance diagnosis using ADDM, Oracle Database periodically (once an hour by default) collects information about the database state and workload it is running. This information is gathered in form of snapshots, which is a statistical summary of the state of the system at any given point in time. These snapshots are stored in the Automatic Workload Repository, residing in the SYSAUX tablespace. The snapshots are stored in this repository for a set time (a week by default) before they are purged in order to make room for new snapshots.

ADDM examines data captured in AWR and performs analysis to determine the major issues on the system on a proactive basis and in many cases recommends solutions and quantifies expected benefits.

Generally, the problems detected by ADDM include the following:

- CPU Bottlenecks
- Poor connection management
- Excessive parsing
- Lock contention
- I/O capacity
- Under sizing of Oracle memory structures such as the PGA, buffer cache, or log buffer
- High load SQL statements
- High PL/SQL and Java time
- RAC specific issues

For more information about using ADDM, see "Diagnosing Performance Problems".

Monitoring General Database State and Workload

The Enterprise Manager home page enables you to monitor the health of your database. It provides a central place for general database state information and is updated periodically. This page reports information that is helpful for monitoring database state and workload.

The **General** section provides a quick view of the database, such as whether the database is Up or Down, the time the database was last started, instance name, host name, and the time of the most recent entry in the alert log.

The **Host CPU** section shows the percentage of CPU time used in the overall system. This chart breaks down CPU percentage into time used by the database and time used by other processes. If your database is taking up most of the CPU time, you can explore the cause further by looking at the **Active Sessions** summary. This summary tells you what the database processes are doing, such as which ones are using CPU, or waiting on I/O. You can drill down for more information by clicking a link, such as **CPU**.

If other processes are taking up most of your CPU time, this indicates that some other application running on the database machine may be the cause of the performance problems. To investigate this further, click the **Host** link under the General section. This link takes you to machine overview page where you can see some general information about the machine such as what operating system it is running, how long has the machine been up, and any potential problems. Clicking the Performance property page takes you to the machine performance summary page.

If you view the Performance Summary, you can see CPU utilization, memory utilization, and disk utilization over time. Below these graphs, you can also view the top ten processes in the CPU. The type of actions you can take to relieve this kind of load depends on your system, but can include eliminating unnecessary processes, adding memory, or adding CPUs.

On the Home page, the **Diagnostic Summary** summarizes the latest ADDM performance findings, This section also summarizes any critical or warning alerts listed in the Alerts section.

The **Alerts** table provides information about any alerts that have been issued along with the severity rating of each. An alert is a notification when a metric threshold is crossed. For example, an alert can be triggered when a tablespace is running out of space.

When an alert is triggered, the name of the metric causing it is displayed in the Name column. The severity icon (Warning or Critical) is displayed, along with time of alert, alert value, time the metric was last checked. You can click the message to learn more about the cause. For more information, see "Alerts".

The **Performance Analysis** section provides a quick summary of the latest ADDM findings, highlighting the issues that are causing the most significant performance impact. This analysis can identify problems such as SQL statements that are consuming significant database time. For more information, see "Performance Self-Diagnostics: Automatic Database Diagnostics Monitor".

Managing Alerts

The following sections describe how to manage alerts.

Viewing Metrics and Thresholds

Metrics are a set of statistics for certain system attributes as defined by Oracle. They are computed and stored by the Automatic Workload Repository, and are displayed on the All Metrics page, which is viewable by clicking **All Metrics** under Related Links heading on the Database Home page (and some other pages). Figure 10–1 shows a portion of the All Metrics page, displaying some of the metrics that AWR computes. When you click a specific metric link, a detail page appears, with more information about the metric. Online help for this page gives you a description of the metric.

Figure 10–1   All Metrics Page

For each of these metrics, you are able to define warning and critical threshold values, and whenever the threshold is crossed, Oracle issues an alert.

Alerts are displayed on the Database Home page under the Alerts heading (or Related Alerts for non-database alerts such as a component of Oracle Net) as shown in Figure 10–2.

Setting thresholds is discussed in "Setting Metric Thresholds" on page 10-5. Actions you might take to respond to alerts are discussed in "Responding to Alerts" on page 10-7.

Figure 10–2   Alerts Section of Database Home Page

When the condition that triggered the alert is resolved and the metric's value is no longer outside the boundary, Oracle clears the alert. Metrics are important for

measuring the health of the database and serve as input for self-tuning and recommendations made by Oracle advisors.

Setting Metric Thresholds

Oracle provides a set of predefined metrics, some of which initially have thresholds defined for them. There may be times when you want to set thresholds for other metrics, or you want to alter existing threshold settings.

One means of setting a threshold was introduced in "Creating a Tablespace" on page 6-8, where you could set warning and critical thresholds on the amount of space consumed in a tablespace.

A more general means of setting thresholds is available using the Edit Thresholds page.

The following steps describe how to set metric thresholds:

1. From the Database Home page, click **Manage Metrics** under the Related Links heading.

   The Manage Metrics page is displayed. It displays the existing thresholds for metrics and any response actions that have been specified. You cannot edit any thresholds on this page.

2. Click **Edit Thresholds**.

   The Edit Thresholds page, shown in Figure 10–3, is displayed. On this page you can enter new **Warning Threshold** and **Critical Threshold** values, or you can modify existing values. In the **Response Actions** field, you can enter an operating system command or a script that you want executed when an alert is issued because a threshold has been crossed.

   For example, to modify the warning threshold for *Tablespace Space Used (%)* metric, you can enter 87 as the percentage. Under *Response Actions*, you can optionally specify a fully qualified path to an operating system script that will cleanup or increase the size of the tablespace.

   Oracle provides a host of alerts that are not enabled by default. You can enable them by specifying threshold values. For example, to enable the alert for *Cumulative Logons (for each second)*, enter 10 for warning and 25 for critical. This will cause the system to warn you when the number of logons for each second exceeds 10.

   You must click **OK** to save your changes.

Figure 10–3   Edit Thresholds Page

3. For more comprehensive management of threshold settings for the different alerts, click the radio button in the **Select** column for that metric, then click **Manage Metric Indexes**.

   The Manage Metric Indexes:*metric_name* page is displayed that enables you to add or delete specific metric threshold and response action settings.

   For example, to set the **Tablespace Space Used (%)** metric thresholds for individual tablespaces, select this metric then click *Manage Metric Indexes*. Enter the tablespace name and its warning and critical values. Click **OK**.

Setting up Notification

You can optionally direct Enterprise Manager to provide notification when events that require your intervention arise. For example, if you specify that you want email notification for critical alerts, and you have a critical threshold set for the *system response time for each call* metric, then you could be sent an email containing a message similar to the following:

```
Host Name=mydb.us.mycompany.com
Metric=Response Time per Call
Timestamp=08-NOV-2003 10:10:01 (GMT -7:00)
Severity=Critical
Message=Response time per call has exceeded the threshold. See the
lattest ADDM analysis.
Rule Name=
Rule Owner=SYSMAN
```

The host name is a link to the Database Home page and in the message there is a link to the latest ADDM analysis.

By default, alerts in critical state such as DB Down, Generic Alert Log Error Status, and Tablespace Used are set up for notification. However, to receive these notifications, you must set up your email information.You can do so as follows:

1. From any Database Control page, click the **Setup** link, which is visible in the header and footer area.

2. On the Setup page, select **Notification Methods**.

3. Enter the required information into the Mail Server portion of the Notifications Methods page. See the online help for assistance.

   There are other methods of notification, including scripts and SNMP (Simplified Network Management Protocol) traps. The latter can be used to communicate with third-party applications.

   > **Note:** So far, this procedure has set up a method of notification, but has not set up an email address to receive the notification. You must also complete the following steps.

4. From any Database Control page, click the **Preferences** link, which is visible in the header and footer area.

5. On the Preferences page, select **General**. Enter your email address in the **E-mail Addresses** section.

6. You can optionally edit notification rules, such as to change the severity state for receiving notification. To do so, select **Notification Rules**. The Notification Rules page appears. For more information about configuring notification rules, see *Oracle Enterprise Manager Advanced Configuration*

Responding to Alerts

When you receive an alert, follow any recommendations it provides, or consider running ADDM or another advisor, as appropriate to get more detailed diagnostics of system or object behavior.

For example, if you receive a Tablespace Space Usage alert, you might take a corrective measure by running the Segment Advisor on the tablespace to identify possible objects for shrinking. You can then shrink the objects to free space. See "Reclaiming Wasted Space" in Chapter 6, "Managing Database Storage Structures".

Additionally, as a response, you can set a corrective script to run as described in "Setting Metric Thresholds".

Clearing Alerts

Most alerts such as an Out of Space are cleared automatically when the cause of the problem disappears. However, other alerts such as Generic Alert Log Error are sent to you for notification and need to be acknowledge by you, the system administrator.

After taking the necessary corrective measures, you can acknowledge an alert by clearing or purging it. Clearing an alert sends the alert to the Alert History, which is viewable from the home page under Related Links. Purging an Alert removes it from the Alert History.

To clear an alert such as Generic Alert Log Error, from the Home page under Alerts, click the alert link. The Alert Log Errors page appears. Select the alert to clear and click **Clear**. To purge an alert, select it and click **Purge**. You can also **Clear Every Open Alert** or **Purge Every Alert** using these buttons.

Diagnosing Performance Problems

At times database performance problems arise that require your diagnosis and correction. Sometimes problems are brought to your attention by users who complain

about slow performance. Other times you might notice performance spikes in the Host CPU chart on the home page.

In all cases, these problems are flagged by the Automatic Database Diagnostics Monitor (ADDM), which does a top-down system analysis every half hour by default and reports its findings on the Oracle Enterprise Manager Home page.

Viewing Performance Analysis

ADDM runs automatically every 30 minutes to coincide with the snapshots taken by AWR. Its output consists of a description of each problem it has identified, and a recommended action.

Findings are displayed in two places on the home page:

- Under the **Performance Analysis** section on the Database Home page, as shown in Figure 10–4.

  You can drill down by clicking the finding. The Findings Details page appears describing the findings and recommended actions.

- Below the Diagnostic Summary heading next to Performance Findings shows the number of findings if any. Clicking this link takes you to the ADDM page.

Figure 10–4 Performance Analysis

Responding to Performance Findings

To respond to a performance finding, click the finding and follow the recommended actions, if any. A recommendation can include running an advisor.

For example, Figure 10–4 shows a performance finding of *SQL statements consuming significant database time were found*, with an impact of 76.53% and recommended summary of SQL Tuning.

Clicking this link takes you to the Performance Finding Details page. Here the recommended action is to run the SQL Advisor, which you can do by clicking **Run Advisor Now**. The advisor runs and gives a recommendation in the form of precise actions for tuning the SQL statements for better performance.

Modifying Default ADDM Behavior

ADDM behavior and analysis is based on the Automatic Workload Repository (AWR), which collects system performance statistics and stores the data in the database. After default installation, the AWR captures data every hour and purges data over seven days old. You can modify both the snapshot frequency and the data retention period as desired.

You can view and alter these settings on the Workload Repository page:

- The retention period for snapshots. This is initially set to 7 days.

- The interval for snapshots. The default and recommended value is 30 minutes.

To navigate to this page, from the Database Administration page, under Workload, select **Automatic Workload Repository**.

To change either of these settings, click **Edit** on the Workload Repository page. The Edit Settings page appears. Enter a new Snapshot retention period or new System Snapshot Interval. Click **OK**.

Running ADDM Manually

By default Oracle runs ADDM every hour. Performance findings from the last snapshot are listed on the Oracle Enterprise Manager Home page. This is described in "Performance Self-Diagnostics: Automatic Database Diagnostics Monitor".

You can also invoke ADDM manually. Reasons for doing so include running it as a recommended action associated with an alert, running it in the middle of a snapshot period, or running it across multiple snapshots.

> **Note:** If you need more frequent ADDM reporting, you can also modify the default snapshot interval. To do so, see "Modifying Default ADDM Behavior".
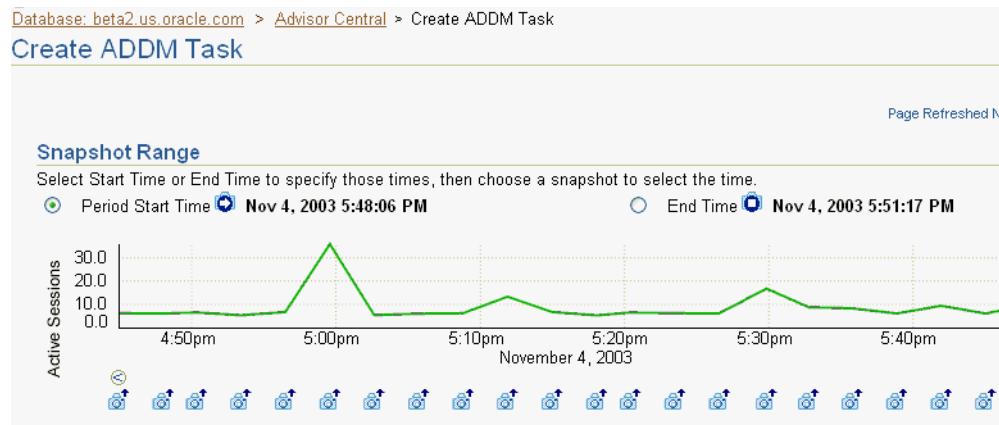
From the Home page, under Related links you can navigate to the ADDM page by clicking **Advisor Central**, then **ADDM**. The Create ADDM Task page appears.

Figure 10–5 is a screen shot of Create ADDM Task page.

Increased session activity shows up as peaks in the graph. To analyze a period across multiple snapshots, select a start time and click **OK**, then choose and end time and click **OK**. The ADDM Task page appears detailing any findings.

Figure 10–5   Create ADDM Task

Using Advisors to Optimize Database Performance

Advisors are powerful tools for database management. They provide specific advice on how to address key database management challenges, covering a wide range of areas including space, performance, and undo management. In general, advisors produce more comprehensive recommendations than alerts. This is because alert generation is intended to be low cost and have minimal impact on performance, whereas advisors are user-invoked, consume more resources and perform more

detailed analysis. This along with the what-if capability of some advisors provides vital information for tuning that cannot be procured from any other source.

About Performance Advisors

This chapter deals primarily with the advisors that can improve performance. These advisors include the SQL Tuning, SQL Access, and Memory Advisors. Table 10–1, " Performance Advisors" describes these advisors.

Other advisors such as the Undo and Segment Advisors are listed in Table 10–2, " Other Advisors".

For example, the shared pool memory advisor graphically displays the impact on performance of changing the size of this component of the SGA.

Examples of situations in which an advisor might be invoked include:

- You want to resolve a problem in a specific area, for example, why a given SQL statement is consuming 50% of CPU time and what to do to reduce its resource consumption.You would use the SQL Tuning Advisor here.

- During application development, you want to tune new schema and its associated SQL workload for optimal performance. You would use the SQL Access Advisor here.

- You want to tune memory usage to find the optimum size for your memory structures such as the shared pool or buffer cache. You can use the memory advisors or ADDM here.

Table 10–1 describes the performance advisors that Oracle provides. These advisors are described in this chapter.

Table 10–1   Performance Advisors

Advisor	Description
Automatic Database Diagnostic Monitor (ADDM)	ADDM makes it possible for the Oracle Database to diagnose its own performance and determine how any identified problems can be resolved. See "Performance Self-Diagnostics: Automatic Database Diagnostics Monitor" and "Diagnosing Performance Problems".
SQL Tuning Advisor	This advisor analyzes SQL statements and makes recommendations for improving performance. See "Using the SQL Tuning Advisor".
SQL Access Advisor	Use this advisor to tune schema to a given SQL workload. For example, the access advisor can provide recommendations for creating indexes and materialized views for a given workload. See "Using the SQL Access Advisor".
Memory Advisor ■ Shared Pool Advisor (SGA) ■ Buffer Cache Advisor (SGA) ■ PGA Advisor	The Memory Advisor is the main advisor for system memory and is responsible for optimizing memory on the instance as a whole. You have the option of having Oracle auto-tune memory. If you choose not to have Oracle auto-tune memory, then you can invoke the SGA Advisors or the PGA Advisor to obtain optimal settings for the components and total size of the SGA or PGA. See "Using the Memory Advisor".

Table 10–2, " Other Advisors" describes other advisors Oracle provides. These are described elsewhere in this book.

Table 10–2   Other Advisors

Advisor	Description
Segment Advisor	The Segment Advisor provides advice on whether an object is a good candidate for a shrink operation based on the level of space fragmentation within an object. The advisor also reports on the historical growth trend of segments. You can use this information for capacity planning and for arriving at an informed decision about which segments to shrink. See "Reclaiming Wasted Space" in Chapter 6, "Managing Database Storage Structures".
Undo Advisor	The Undo Advisor helps in identifying problems in the undo tablespace and assists in correctly sizing the undo tablespace. The Undo Advisor can also be used to set the low threshold value of the undo retention period for any flashback requirements. See "Using the Undo Advisor" in Chapter 6, "Managing Database Storage Structures".

You can invoke an advisor from the Advisor Central home page that is displayed when you click **Advisor Central** under the Related Links heading on the Database Home page, or on other pages where it is listed. You can invoke advisors in other ways, often through recommendations from ADDM or alerts.

Using the SQL Tuning Advisor

Use the SQL Tuning Advisor for tuning SQL statements. Typically, you can run this advisor in response to an ADDM performance finding recommending its use.

Additionally, you can run the SQL Tuning Advisor on the most resource intensive SQL statements, referred to as top SQL, from the cursor cache or the AWR, as well as on a user-defined SQL workload.

To run the SQL Tuning Advisor do the following:

1. On the Home Page, under Related Links, click **Advisor Central**, then click **SQL Tuning Advisor**. The SQL Tuning Advisor Links page appears.

2. The advisor can be run on one of the following sources

    ■   Top SQL---The consist of recently active top SQL statements from the cursor cache (Spot SQL) or historical top SQL from the AWR (Period SQL).

    ■   SQL Tuning Sets---These consist of SQL statements you provide. An STS can be created from SQL statements captured by AWR snapshots or from any SQL workload.

3. For example, you can select **Top SQL**. The Top SQL page appears. This page has two tabs, Spot SQL and Period SQL. Spot lists recent top SQL from the cursor cache, while Period SQL lists historical top SQL captured in the AWR. You must select an interval to analyze by dragging the shaded box over the period. You then select one or more SQL statements to analyze during the selected period.

4. Click **Run SQL Tuning Advisor**. The SQL Tuning Options page appears showing the SQL statements in the interval. Give your task a name and description, select the scope for the analysis **(Comprehensive or Limited)**, and select a start time for the task. Click **OK**.

5. Navigate back to the Advisor Central page. The status of Advisor Tasks are listed under this heading in the results section. You must wait until your task status is COMPLETED. You can check the status by clicking your browser's Refresh button. Then, select your task and click **View Result**.

6. The SQL Tuning Result page appears. To view recommendations, select the SQL statement and click **View Recommendations**. The recommendation can include one or more of the following:

- Create an index to offer alternate, faster access paths to the query optimizer.

- Accept SQL profile, which contains additional SQL statistics specific to the statement that enables the query optimizer to generate a significantly better execution plan.

- Gather optimizer statistics on objects with stale or no statistics

- Advice on how to rewrite a query for better performance.

Using the SQL Access Advisor

The SQL Access Advisor helps define appropriate access structures such as indexes and materialized views to optimize SQL queries. It takes a SQL workload as an input and recommends which indexes, materialized views, or logs to create, drop, or retain for faster performance. You can select your workload from different sources including current and recent SQL activity, a SQL repository, or a user-defined workload such as from a development environment.

The recommendations that this advisor makes include possible indexes, materialized views, or materialized view logs that can improve your query performance for the given workload.

To run this advisor, navigate to the Advisor Central page, and click **SQL Access Advisor**. This begins a wizard which starts by prompting you for your workload source. You then select if you want the advisor to recommend indexes, materialized views or both. You can select to run the advisor in limited or comprehensive mode. Limited mode runs faster by concentrating on highest cost statements.

You then schedule and submit your job. Results are posted on the Advisor Central page. The SQL Access Advisor recommendations are ordered by cost benefit. For example, a recommendation might consist of a a SQL script with one or more `CREATE INDEX` statements, which you can implement by clicking **Schedule Implementation**.

Using the Memory Advisor

The Memory Advisor helps you tune the size of your memory structures. You can use this advisor only when automatic memory tuning is disabled.

The memory advisor comprises three advisors that give you recommendations on the following memory structures:

- Shared pool in SGA

- Buffer cache in SGA

- PGA
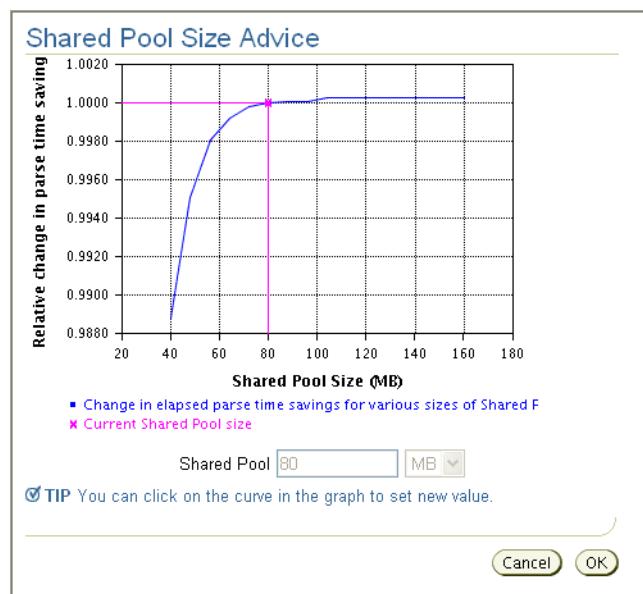
To invoke the Memory Advisors, click **Memory Advisor** on the Advisor Central page. The Memory Parameters: SGA page appears. This page gives breakdown of memory usage for the system global area (SGA). This memory area is a group of shared memory structures that contain data and control information for a single Oracle instance. The shared pool and buffer cache are part of this area. For more information on these structures, click **Help**.

The **Automatic Shared Memory Management** setting should be disabled in order to run the advisor. To run either the shared pool advisor or the buffer cache advisor, click **Advice** next to the field.

For example, to run the advisor on the shared pool, click **Advice** next to this field. The Shared Pool Advice graph appears. Refer to Figure 10–6, "Shared Pool Size Advice".

Change in parse time saving is plotted against shared pool size. A higher number for parse time saving is better for performance. In this example, the graph tells us that a shared pool size larger than 80M will not improve performance by much. Thus 80M is the recommended optimal shared pool size.

Figure 10–6   Shared Pool Size Advice

Similarly, to run the advisor on the buffer cache, click **Advice** next to this field. The Buffer Cache Size Advice graph appears, plotting relative change in physical reads versus cache size. Since a bigger cache implies less disk reads, the smaller number for change in physical reads is better for performance. Like the shared pool memory, there is usually an optimal cache size above which performance improvement is negligible.

To run the PGA advisor, click the **PGA** property page. Running this advisor is similar to running the SGA advisors. Cache hit percentage is plotted against memory size. Higher hit ratios indicate better cache performance. The optimal zone for cache hit percentage is between 75 and 100%.

Monitoring and Tuning: Oracle by Example Series

Oracle by Example (OBE) has a series on the *Oracle Database 2 Day DBA* book. This OBE steps you through the tasks in this chapter, and includes annotated screen shots.

To view the Monitoring OBE, point your browser to the following location:

```
http://otn.oracle.com/obe/2day_dba/monitoring/monitoring.htm
```

Index