# Oracle® Database

Upgrade Guide

10*g* Release 2 (10.2)

**B14238-02**

January 2008

ORACLE®

Oracle Database Upgrade Guide, 10*g* Release 2 (10.2)

B14238-02



Primary Author: Kathy Rich and Viv Schupmann

Contributor: Valarie Moore (graphic artist)

Contributors: Rae Burns, Naresh Pamnani

# Contents

## 6 Upgrading Your Applications

## 7 Downgrading a Database Back to the Previous Oracle Database Release



## 8 Data Copying Using Export/Import



## A Initialization Parameter and Data Dictionary Changes



## B Migrating from Server Manager to SQL*Plus

# Preface

This manual guides you through the process of planning and executing database upgrades on the Oracle Database. In addition, this manual provides information about compatibility, about upgrading applications to the new Oracle Database 10*g* release, and about important changes in the new release, such as initialization parameter changes and data dictionary changes.

Oracle Database Upgrade Guide contains information that describes the features and functionality of the Oracle Database (also known as the standard edition) and the Oracle Database Enterprise Edition products. The Oracle Database and the Oracle Database Enterprise Edition have the same basic features. However, several advanced features are available only with the Enterprise Edition, and some of these are optional. For example, to use application failover, you must have the Enterprise Edition with the Real Application Clusters option.

> **See Also:** *Oracle Database New Features* for information about the differences between the Oracle Database and the Oracle Database Enterprise Edition and the features and options that are available to you.

This preface contains these topics:



## Audience

Oracle Database Upgrade Guide is intended for database administrators (DBAs), application developers, security administrators, system operators, and anyone who plans or executes Oracle Database upgrades.

To use this document, you need to be familiar with the following:

- Relational database concepts
- Your current release of the Oracle Database
- Your operating system environment

# 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. Accessibility 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 more information, visit the Oracle Accessibility Program Web site at

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

### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

### TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

# Related Documentation

---

**Note:** For late-breaking updates and best practices about preupgrade, post-upgrade, compatibility, and interoperability discussions, see Note 466181.1 on Oracle*Metalink* (https://metalink.oracle.com/) that links to "The Upgrade Companion" web site.

---

For more information, see these Oracle resources:

- *Oracle Database Concepts* for a comprehensive introduction to the concepts and terminology used in this manual
- *Oracle Database Administrator's Guide* for information about administering the Oracle Database
- *Oracle Database SQL Reference* for information on Oracle's SQL commands and functions
- *Oracle Database Utilities* for information about the utilities bundled with the Oracle Database, including Export, Import, and SQL*Loader
- *Oracle Database Net Services Administrator's Guide* for information about Oracle Net Services

- *Oracle Database Enterprise User Administrator's Guide* for information about Oracle Label Security

Many of the examples in this book use the sample schemas, which are installed by default when you select the Basic Installation option with an Oracle Database installation. See *Oracle Database Sample Schemas* for information about 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

The following text conventions are used in this document:

| Convention | Meaning |
|---|---|
| **boldface** | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary. |
| *italic* | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values. |
| `monospace` | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

# 1

# Introduction

This chapter provides an overview of the database upgrade process, and information about running multiple releases of the Oracle Database.

This chapter covers the following topics:



> **Note:** For late-breaking updates and best practices about preupgrade, post-upgrade, compatibility, and interoperability discussions, see Note 466181.1 on Oracle*Metalink* (`https://metalink.oracle.com/`) that links to "The Upgrade Companion" web site.

## Overview of the Database Upgrade Process

This section includes an overview of the major steps required to upgrade an existing Oracle Database to the new Oracle Database 10*g* release. These procedures transform an existing Oracle Database system (including associated applications) into an Oracle Database 10*g* system. Oracle Database 10*g* is compatible with all earlier Oracle Database releases. Therefore, databases upgraded using the procedures described in this book can work in the same manner as in earlier releases and, optionally, can leverage new Oracle Database 10*g* functionality.

Oracle Database 10*g* supports the following tools and methods for upgrading a database to the new Oracle Database 10*g* release:

- **Database Upgrade Assistant (DBUA)**

  Provides a graphical user interface (GUI) that guides you through the upgrade of a database. The DBUA can be launched during installation with the Oracle Universal Installer, or you can launch the DBUA as a standalone tool at any time in the future.

> **Note:** The DBUA is the recommended method for upgrading a database. Beginning with release 10.2, you can also use the DBUA to upgrade to a new patch release of Oracle 10*g* release 2.

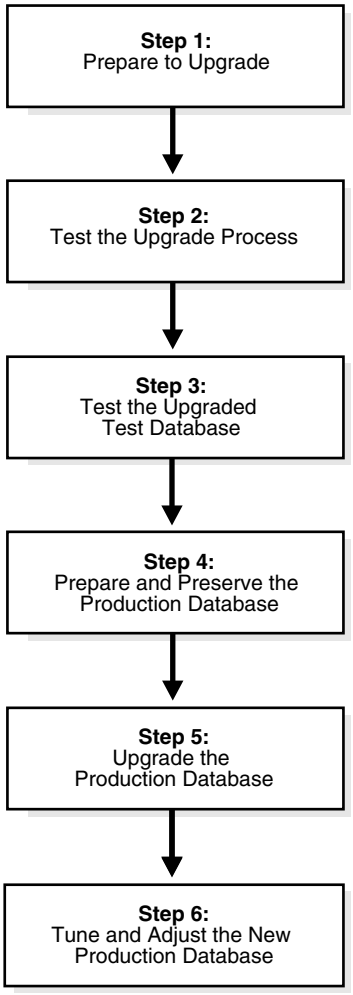
- **Manual upgrade using SQL scripts and utilities**
- **Export and Import utilities**
- **`CREATE TABLE AS` SQL statement**

  Copies data from a database into a new Oracle Database 10*g* database. Data copying can copy a subset of the data, leaving the database unchanged.

These tools and methods are described in more detail in "Choose an Upgrade Method" on page 2-2.

The major steps in the upgrade process are illustrated in Figure 1–1.

***Figure 1–1 Upgrade Steps***



**Note:** The upgrade steps apply to all operating systems, with the possible exception of a few operating system-specific details identified in your operating system-specific Oracle documentation.

The following list outlines the major steps performed during the upgrade process. Detailed instructions are provided in the appropriate chapters later in this book.

### Step 1: Prepare to Upgrade

- Become familiar with the features of the new Oracle Database 10*g* release.
- Determine the upgrade path to the new Oracle Database 10*g* release.
- Choose an upgrade method.
- Choose an Oracle home directory for the new Oracle Database 10*g* release.
- Develop a testing plan.
- Prepare a backup strategy.

### Step 2: Test the Upgrade Process

- Perform a test upgrade using a test database. The test upgrade should be conducted in an environment created for testing and should not interfere with the actual production database.

### Step 3: Test the Upgraded Test Database

- Perform the tests you planned in Step 1 on the test database and on the test database that was upgraded to the new Oracle Database 10*g* release.
- Compare results, noting anomalies between test results on the test database and on the upgraded database.
- Investigate ways to correct any anomalies you find and then implement the corrections.
- Repeat Step 1, Step 2, and the first parts of Step 3, as necessary, until the test upgrade is completely successful and works with any required applications.

Chapter 2, "Preparing to Upgrade" provides detailed information about Steps 1 through 3.

### Step 4: Prepare and Preserve the Production Database

- Prepare the current production database as appropriate to ensure the upgrade to the new Oracle Database 10*g* release will be successful.
- Schedule the downtime required for backing up and upgrading the production database.
- Back up the current production database. Perform a full or an incremental backup, as necessary, to ensure your database is protected against data loss.

### Step 5: Upgrade the Production Database

- Upgrade the production database to the new Oracle Database 10*g* release.
- After the upgrade, perform a full backup of the production database and perform other post-upgrade tasks.

Chapter 3 describes Steps 4 and 5 when using the Database Upgrade Assistant or when performing a manual upgrade. Chapter 4 describes the backup procedure after the upgrade and other post-upgrade tasks.

### Step 6: Tune and Adjust the New Production Database

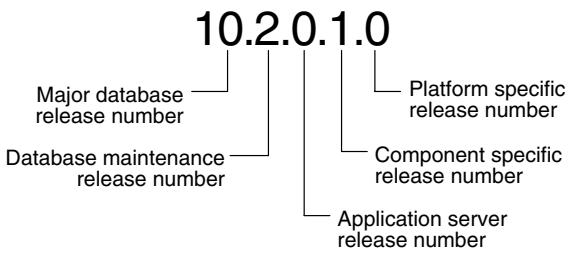
- Tune the new Oracle Database production database. The new Oracle Database production database should perform as good as, or better than, the database prior to the upgrade. Chapter 4 describes these adjustments.
- Determine which features of the new Oracle Database 10*g* release you want to use and update your applications accordingly.
- Develop new database administration procedures, as needed.
- Do not upgrade production users to the new Oracle Database until all applications have been tested and operate properly. Chapter 6 describes considerations for updating applications.

During the upgrade, consider running multiple releases of the database software so you can use the existing release as your production environment while you test the new release. See "Running Multiple Oracle Releases" on page 1-5.

## Oracle Release Numbers

This book describes moving between different **releases** of the Oracle database server. Figure 1–2 describes what each part of a release number represents.

***Figure 1–2 Example of an Oracle Release Number***




**Note:** Starting with Oracle9*i* release 9.2, maintenance releases of Oracle are denoted by a change to the second digit of a release number. In previous releases, the third digit indicated a particular maintenance release.

**See Also:** *Oracle Database Administrator's Guide* for more information about Oracle release numbers

When a statement is made in this book about a major database release number, the statement applies to all releases within that major database release. References to Oracle Database include release 10.1 and 10.2; references to Oracle9*i* include release 9.0.1 and release 9.2.

Similarly, when a statement is made in this book about a maintenance release, the statement applies to all component-specific (also referred to as patch set releases) and platform-specific releases within that maintenance release. So, a statement about

release 9.2 applies to release 9.2.0.1, release 9.2.0.2, and all other platform-specific releases within release 9.2.

## Running Multiple Oracle Releases

You can run different releases of the Oracle Database on the same computer at the same time. However, you must observe the following conditions when using multiple releases:

- An Oracle Database release must be installed in a new Oracle home that is separate from previous releases of Oracle.

  There cannot be more than one release per Oracle home. Oracle recommends that you adopt an Optimal Flexible Architecture (OFA) when creating multiple Oracle homes. See "Using Optimal Flexible Architecture (OFA)" on page 1-6 for more information.

  **Caution:** It is not possible to install release 10.2 products into an existing Oracle home of a prior major release. This functionality was only available for certain previous releases and has not been continued.

- Each database server can access only a database that is consistent with its release number.

  For example, if you have Oracle9*i* and Oracle Database 10*g* installed on the same computer, then the Oracle9*i* database server can access Oracle9*i* databases but not Oracle Database 10*g* databases, and the Oracle Database 10*g* database server can access Oracle Database 10*g* databases but not Oracle9*i* databases.

  **See Also:** Your operating system-specific Oracle documentation for more information about running multiple releases of Oracle on your operating system. Restrictions may apply on some operating systems.

The following sections provide general information about running multiple releases of the Oracle Database.

### Install Databases in Multiple Oracle Homes on the Same Computer

You can install Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* databases in multiple (separate) Oracle homes on the same computer and have Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* clients connecting to any or all of the databases.

### Install Databases in Multiple Oracle Homes on Separate Computers

You can install Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* databases in multiple (separate) Oracle homes on separate computers and have Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* clients connecting to any or all of the databases.

### Upgrade a Database to the Current Release

You can upgrade an Oracle8*i*, Oracle9*i*, or Oracle Database 10*g* database to the new Oracle Database 10*g* release and have Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* clients connecting to the upgraded database.

### Upgrade Clients to the Current Release

You can upgrade any or all of your Oracle8*i*, Oracle9*i*, or Oracle Database 10*g* clients to the new Oracle Database 10*g* release. The Oracle Database 10*g* client can be used to access your Oracle8*i*, Oracle 9*i*, and Oracle 10*g* databases.

## Using Optimal Flexible Architecture (OFA)

Oracle recommends the Optimal Flexible Architecture (OFA) standard for your Oracle Database installations. The OFA standard is a set of configuration guidelines for efficient and reliable Oracle databases that require little maintenance.

OFA provides the following benefits:

- Organizes large amounts of complicated software and data on disk to avoid device bottlenecks and poor performance
- Facilitates routine administrative tasks, such as software and data backup functions, which are often vulnerable to data corruption
- Alleviates switching among multiple Oracle databases
- Adequately manages and administers database growth
- Helps to eliminate fragmentation of free space in the data dictionary, isolates other fragmentation, and minimizes resource contention.

If you are not currently using the OFA standard, then switching to the OFA standard involves modifying your directory structure and relocating your database files.

**See Also:**

- Your operating system-specific Oracle documentation for more information about OFA
- *Oracle Database Administrator's Guide* for information about relocating database files

## Converting Databases to 64-bit Oracle Database Software

If you are installing 64-bit Oracle Database 10*g* software but were previously using a 32-bit Oracle Database installation, then the databases will automatically be converted to 64-bit during the upgrade to Oracle Database 10*g* except when upgrading from Release 1 (10.1) to Release 2 (10.2).

---

**Note:** The process is not automatic for the release 1 to release 2 upgrade, but is automatic for all other upgrades. This is because the `utlip.sql` script is not run during the release 1 to release 2 upgrade to invalid all PL/SQL objects. You must run the `utlip.sql` script as the last step in the release 10.1 environment, before upgrading to release 10.2.

---

## Rolling Upgrades

The term **rolling upgrade** refers to upgrading different databases or different instances of the same database (in a Real Application Clusters environment) one at a time, without stopping the database. Oracle Database 10*g* provides the following methods of upgrading the Oracle Database software version:

- Oracle Data Guard and logical standby databases

  Using SQL Apply and logical standby databases, you can upgrade Oracle database software and patch sets with little or no database downtime. For example, you can upgrade the Oracle Database software from patch set release 10.1.0.*n* to the next database 10.1.0.(*n*+1) patchset release, or upgrade the Oracle Database from 10.1 to 10.2.

  See "Upgrading Standby Databases" on page 3-2 for more information.

- Oracle Streams

  Using Streams source and destination databases, you can upgrade to a new version of the Oracle Database, migrate an Oracle Database to a different operating system or character set, upgrade user-created applications, and apply Oracle Database patches. These maintenance operations use the features of Oracle Streams to achieve little or no database down time.

  See *Oracle Streams Concepts and Administration* for more information.

- Real Application Clusters Rolling Upgrades with the opatch Utility

  Using the opatch command-line utility, you can perform rolling patch upgrades with Real Application Clusters (RAC), incurring little or no database downtime. Note that the opatch utility can only be used to apply individual patches, not patch set releases. See *Oracle Universal Installer and OPatch User's Guide* for more information about the opatch utility.

  The advantage of a RAC rolling upgrade is that it enables at least some instances of the RAC installation to be available during the scheduled outage required for patch upgrades. Only the RAC instance that is currently being patched needs to be brought down; the other instances can continue to remain available. This means that the effect on the application downtime required for such scheduled outages is further minimized. The Oracle opatch utility enables you to apply the patch successively to the different instances of the RAC installation.

  See *Oracle High Availability Architecture and Best Practices* for more information.

## Moving From the Standard Edition to the Enterprise Edition

If you are using the Standard Edition of the Oracle Database and want to move to the Enterprise Edition, then complete the following steps:

1. Ensure that the release number of your Standard Edition server software is the same release as the Enterprise Edition server software.

   For example, if your Standard Edition server software is release 10.2.0, then you should upgrade to release 10.2.0 of the Enterprise Edition.

2. Shut down your database.

3. If your operating system is Windows, then stop all Oracle services, including the `OracleService`*`SID`* Oracle service, where *`SID`* is the instance name.

4. Deinstall the Standard Edition server software.

5. Install the Enterprise Edition server software using the Oracle Universal Installer.

   Select the same Oracle home that was used for the de-installed Standard Edition. During the installation, be sure to select the Enterprise Edition. When prompted, choose Software Only from the Database Configuration screen.

6. Start up your database.

Your database is now upgraded to the Enterprise Edition.

---

**Note:** If you have a Standard Edition database at a release prior to Oracle Database 10*g*, you can change it to an Enterprise Edition database by first installing the Enterprise Edition and then following the normal upgrade procedures, as described in this manual.

---

# 2

# Preparing to Upgrade

This chapter describes the steps to complete before upgrading a database to the new Oracle Database 10*g* release. This chapter covers in detail Steps 1 through 3 of the upgrade process that were outlined in "Overview of the Database Upgrade Process" on page 1-1.

This chapter covers the following topics:

- Prepare to Upgrade
- Test the Upgrade Process
- Test the Upgraded Test Database

> **See Also:** *Oracle Database Net Services Administrator's Guide* for information about upgrade considerations for Oracle Net Services

> **Note:** Some aspects of upgrading are operating system-specific. See your operating system-specific Oracle documentation for additional information about preparing to upgrade.

## Prepare to Upgrade

Complete the following tasks to prepare to upgrade:

- Become Familiar with the Features of the New Oracle Database 10g Release
- Determine the Upgrade Path to the New Oracle Database 10g Release
- Choose an Upgrade Method
- Choose an Oracle Home Directory for the New Oracle Database 10g Release
- Develop a Testing Plan
- Prepare a Backup Strategy

### Become Familiar with the Features of the New Oracle Database 10*g* Release

Before you plan the upgrade process, become familiar with the features of the new Oracle Database 10*g* release. *Oracle Database New Features* is a good starting point for learning the differences between Oracle Database releases. Also, check specific books in the Oracle Database 10*g* documentation set to find information about new features for a certain component; for example, see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide* for changes in Real Application Clusters.

**Note:** Oracle Database 10*g* training classes are an excellent way to learn how to take full advantage of the functionality available with the Oracle Database. Connect to the following web page for more information:

```
http://education.oracle.com/
```

## Determine the Upgrade Path to the New Oracle Database 10*g* Release

The path that you must take to upgrade to the new Oracle Database 10*g* release depends on the release number of your current database. It may not be possible to upgrade directly from your current version of Oracle Database to the latest version. Depending on your current release, you may need to upgrade through one or more intermediate releases to upgrade to the new Oracle Database 10*g* release.

For example, if the current database is running release 8.1.6, then first upgrade to release 8.1.7 using the instructions in *Oracle8i Migration* for release 8.1.7. The release 8.1.7 database can then be upgraded to the new Oracle Database 10*g* release using the instructions in this book.

Table 2–1 contains the required upgrade path for each release of the Oracle Database. Use the upgrade path and the specified documentation to upgrade your database.

***Table 2–1 Upgrade Paths***

| Current Release | Upgrade Path |
|---|---|
| 7.3.3 and lower<br>7.3.4<br>8.0.3<br>8.0.4<br>8.0.5<br>8.0.6<br>8.1.5<br>8.1.6 | Direct upgrade is *not* supported. Upgrade to an intermediate Oracle Database release before you can upgrade to the new Oracle Database 10*g* release, as follows:<br>■ 7.3.3 (or lower) -> 7.3.4 -> 8.1.7.4 -> 10.2<br>■ 7.3.4 -> 8.1.7.4 -> 10.2<br>■ 8.0.*n* -> 8.1.7.4 -> 10.2<br>■ 8.1.*n* -> 8.1.7.4 -> 10.2<br>When upgrading to an intermediate Oracle Database release, follow the instructions in the intermediate release's documentation. Then, upgrade the intermediate release database to the new Oracle Database 10*g* release using the instructions in Chapter 3, "Upgrading to the New Oracle Database 10g Release". |
| 8.1.7.4<br>9.0.1.4<br>9.2.0.4<br>10.1.0.2 | Direct upgrade from 8.1.7.4, 9.0.1.4 or higher, 9.2.0.4 or higher, and 10.1.0.2 or higher to the newest Oracle Database 10*g* release is supported. However, you must first apply the specified minimum patch release indicated in the Current Release column.<br>To upgrade to the new Oracle Database 10*g* release, follow the instructions in Chapter 3, "Upgrading to the New Oracle Database 10g Release". |

## Choose an Upgrade Method

The following sections describe the upgrade methods you can use to upgrade your database to the new Oracle Database 10*g* release:

- Database Upgrade Assistant
- Manual Upgrade
- Export/Import
- Data Copying

### 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 DBUA automates the upgrade process by performing all of the tasks normally performed manually. The DBUA makes appropriate recommendations for configuration options such as tablespaces and redo logs. You can then act on these recommendations.

The DBUA provides support for Real Application Clusters (RAC) and Automatic Storage Management (ASM).

**Support for Real Application Clusters** In a Real Application Clusters (RAC) environment, the DBUA upgrades all the database and configuration files on all nodes in the cluster.

> **Note:** On Windows operating systems, DBUA does not support a direct upgrade of Oracle Parallel Server version 8.1.7 databases to Oracle Database 10*g* with RAC. First, manually upgrade the Oracle Parallel Server database to Oracle Real Application Clusters Oracle9*i* release 2 (9.2), and then upgrade it to Oracle Database 10*g* with Real Application Clusters (RAC) using DBUA.

**Support for Automatic Storage Management** The DBUA supports upgrades of databases that use Automatic Storage Management (ASM). If an ASM instance is detected, you have the choice of updating both the database and ASM or only the ASM instance.

> **See Also:** Upgrade the Database Using the Database Upgrade Assistant on page 3-11

### Manual Upgrade

A manual upgrade consists of running SQL scripts and utilities from a command line to upgrade a database to the new Oracle Database 10*g* release.

While a manual upgrade gives you finer control over the upgrade process, it is more susceptible to error if any of the upgrade or pre-upgrade steps are either not followed or are performed out of order.

**Before the Upgrade** When manually upgrading a database, perform the following pre-upgrade steps:

- Analyze the database using the Pre-Upgrade Information Tool. The Upgrade Information Tool is a SQL script that ships with the new Oracle Database 10*g* release, and must be run in the environment of the database being upgraded.

  The Upgrade Information Tool displays warnings about possible upgrade issues with the database. It also displays information about required initialization parameters for the new Oracle Database 10*g* release.
- Prepare the new Oracle Home.
- Perform a backup of the database.

Depending on the release of the database being upgraded, you may need to perform additional pre-upgrade steps (adjust the parameter file for the upgrade, remove obsolete initialization parameters and adjust initialization parameters that might cause upgrade problems).

**After the Upgrade** Review the upgrade spool log file and use the Post-Upgrade Status Tool. The Post-Upgrade Status Tool is a SQL script that ships with the new Oracle Database 10*g* release, and should be run in the environment of the new Oracle Database 10*g* release.

**See Also:** "Upgrade the Database Manually" on page 3-15

### Export/Import

Unlike the DBUA or a manual upgrade, the Export/Import utilities physically copy data from your current database to a new database. You can use either the Oracle Data Pump Export and Import utilities (available as of Oracle Database 10*g*) or the original Export and Import utilities to perform a full or partial export from your database, followed by a full or partial import into a new Oracle Database 10*g* database. Export/Import can copy a subset of the data in a database, leaving the database unchanged.

The current database's Export utility copies specified parts of the database into an export dump file. Then, the Import utility of the new Oracle Database 10*g* release loads the exported data into a new database. However, the new Oracle Database 10*g* database must already exist before the export dump file can be copied into it.

When importing data from an earlier release, the Oracle Database 10*g* Import utility makes appropriate changes to data definitions as it reads earlier releases' export dump files.

The following sections highlight aspects of Export/Import that may help you to decide whether to use Export/Import to upgrade your database.

**Export/Import Effects on Upgraded Databases** The Export/Import upgrade method does not change the current database, which enables the database to remain available throughout the upgrade process. However, if a consistent snapshot of the database is required (for data integrity or other purposes), then the database must run in restricted mode or must otherwise be protected from changes during the export procedure. Because the current database can remain available, you can, for example, keep an existing production database running while the new Oracle Database 10*g* database is being built at the same time by Export/Import. During the upgrade, to maintain complete database consistency, changes to the data in the database cannot be permitted without the same changes to the data in the new Oracle Database 10*g* database.

Most importantly, the Export/Import operation results in a completely new database. Although the current database ultimately contains a copy of the specified data, the upgraded database may perform differently from the original database. For example, although Export/Import creates an identical copy of the database, other factors, such as disk placement of data and unset tuning parameters, may cause unexpected performance problems.

**Export/Import Benefits** Upgrading using Export/Import offers the following benefits:

- Defragments the data - you can compress the imported data to improve performance.
- Restructures the database - you can create new tablespaces or modify existing tables, tablespaces, or partitions to be populated by imported data.
- Enables the copying of specified database objects or users - you can import only the objects, users, and other items that you wish.

- Serves as a backup archive - you can use a full database export as an archive of the current database.

**Time Requirements for Export/Import** Upgrading an entire database by using Export/Import can take a long time, especially compared to using the DBUA or performing a manual upgrade. Therefore, you may need to schedule the upgrade during non-peak hours or make provisions for propagating to the new database any changes that are made to the current database during the upgrade.

> **See Also:** Chapter 8, "Data Copying Using Export/Import"

### Data Copying

You can copy data from one Oracle Database to another using database links. For example, you can create new tables and fill the tables with data by using the `INSERT INTO` statement and the `CREATE TABLE ... AS` statement.

Copying data and Export/Import offer the same advantages for upgrading. Using either method, you can defragment data files and restructure the database by creating new tablespaces or modifying existing tables or tablespaces. In addition, you can copy only specified database objects or users.

Copying data, however, unlike Export/Import, enables the selection of specific rows of tables to be placed into the new database. Copying data is a good method for copying only part of a database table. In contrast, using Export/Import, you can copy only entire tables.

> **See Also:** `CREATE TABLE` statement in *Oracle Database SQL Reference*

## Choose an Oracle Home Directory for the New Oracle Database 10*g* Release

You must choose an Oracle home directory for the new Oracle Database 10*g* release that is separate from the Oracle home directory of your current release. You cannot install the new Oracle Database software into the same Oracle home directory as your current release, unless you are installing an Oracle 10*g* patch set release. For a patch set release, you may use the same release 10.2 Oracle home.

Using separate installation directories enables you to keep your existing software installed along with the new Oracle Database software. This method enables you to test the upgrade process on a test database before replacing your production environment entirely.

## Develop a Testing Plan

You need a series of carefully designed tests to validate all stages of the upgrade process. Executed rigorously and completed successfully, these tests ensure that the process of upgrading the production database is well understood, predictable, and successful. Perform as much testing as possible before upgrading the production database. Do not underestimate the importance of a test program.

The testing plan must include the following types of tests.

### Upgrade Testing

Upgrade testing entails planning and testing the upgrade path from your current database to the new Oracle Database, whether you use the DBUA, perform a manual upgrade, or use Export/Import or other data-copying methods.

Regardless of the upgrade method you choose, you must establish, test, and validate an upgrade plan.

### Minimal Testing

Minimal testing entails moving all or part of an application from the current database to the new Oracle Database and running the application without enabling any new database features. Minimal testing is a very limited type of testing that may not reveal potential issues that may appear in a "real-world" production environment. However, minimal testing will immediately reveal any application startup or invocation problems.

### Functional Testing

Functional testing is a set of tests in which new and existing functionality of the system are tested after the upgrade. Functional testing includes all database, networking, and application components. The objective of functional testing is to verify that each component of the system functions as it did before upgrading and to verify that new functions are working properly.

### Integration Testing

Integration testing examines the interaction of each component of the system. Consider the following factors when you plan your integration testing:

- Pro*C/C++ applications running against a new Oracle Database instance should be tested to ensure that there are no problems with the new software.
- Graphical user interfaces should be tested with other components.
- Subtle changes in the new Oracle Database, such as datatypes, data in the data dictionary (additional rows in the data dictionary, object type changes, and so on) can have an effect all the way up to the front-end application, regardless of whether or not the application is directly connected to a new Oracle Database instance.
- If the connection between two components involves Net8 or Oracle Net Services, then those connections should also be tested and stress tested.

### Performance Testing

Performance testing of the new Oracle Database compares the performance of various SQL statements in the new Oracle Database with the statements' performance in the current database. Before upgrading, you should understand the performance profile of the application under the current database. Specifically, you should understand the calls the application makes to the database server.

For example, if you are using Real Application Clusters, and you want to measure the performance gains realized from using cache fusion when you upgrade to the new Oracle Database 10*g* release, then make sure you record your system's statistics before upgrading. For cache fusion, record the statistics from the `V$SYSSTAT` and `V$INSTANCE_CACHE_TRANSFER` views. Doing so enables you to compare pre-cache fusion and post-cache fusion performance statistics.

For best results, run the SQL scripts `utlbstat.sql` and `utlestat.sql` to collect `V$SYSSTAT` statistics for a specific period. Use a collection time frame that most consistently reflects peak production loads with consistent transaction activity levels. To obtain data from `V$LOCK_ACTIVITY` and `V$LOCK_CLASS_PING`, use a `SELECT *` statement at the beginning and end of the statistics collection period. Repeat this process after cache fusion is running on the new Oracle Database release and evaluate

your system's performance as described in *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide*.

> **See Also:** *Oracle Database Performance Tuning Guide* for information about tuning. To thoroughly understand the application's performance profile under the source database, enable the SQL trace facility and profile with TKPROF.

### Volume and Load Stress Testing

Volume and load stress testing tests the entire upgraded database under high volume and loads. Volume describes the amount of data being manipulated. Load describes the level of concurrent demand on the system. The objective of volume and load testing is to emulate how a production system might behave under various volumes and loads.

Volume and load stress testing is crucial, but is commonly overlooked. Oracle has found that customers often do not conduct any kind of volume or load stress testing. Instead, customers often rely on benchmarks that do not characterize business applications. Benchmarks of the application should be conducted to uncover problems relating to functionality, performance, and integration, but they cannot replace volume and load stress testing.

After you upgrade the database, you should test the data to ensure that all data is accessible and that the applications function properly. You should also determine whether any database tuning is necessary. If possible, you should automate these testing procedures.

The testing plan should reflect the work performed at the site. You should test the functionality and performance of all applications on the production databases. Gather performance statistics for both normal and peak usage.

### Specific Pre-Upgrade and Post-Upgrade Tests

Include the following tests in your testing plan:

- Timing tests
- Data dictionary growth observations
- Database resource usage observations, such as undo and temporary segment usage

Collecting this information will help you compare the current database with the new Oracle Database.

Use EXPLAIN PLAN on both the previous and new databases to determine the execution plan Oracle follows to execute each SQL statement. Use the `INTO` clause to save this information in tables.

After upgrading, you can compare the execution plans of the new Oracle Database with the execution plans of the current database. If there is a difference, then execute the statement on the new Oracle Database and compare the performance with the performance of the statement executed on the current database.

> **See Also:** *Oracle Database Performance Tuning Guide* for more information about EXPLAIN PLAN.

### Prepare a Backup Strategy

The ultimate success of your upgrade depends heavily on the design and execution of an appropriate backup strategy. To develop a backup strategy, consider the following questions:

- How long can the production database remain inoperable before business consequences become intolerable?
- What backup strategy should be used to meet your availability requirements?
- Are backups archived in a safe, offsite location?
- How quickly can backups be restored (including backups in offsite storage)?
- Have recovery procedures been tested successfully?

Your backup strategy should answer all of these questions and include procedures for successfully backing up and recovering your database.

> **See Also:** *Oracle Database Backup and Recovery Basics* for information on database backups

## Test the Upgrade Process

Create a test environment that will not interfere with the current production database. Your test environment will depend on the upgrade method you have chosen:

- If you plan to use the DBUA or perform a manual upgrade, then create a test version (typically a subset) of the current production database to test the upgrade.
- If you plan to use Export/Import, then export and import small test pieces of the current production database.

Practice upgrading the database using the test environment. The best upgrade test, if possible, is performed on an exact copy of the database to be upgraded, rather than on a downsized copy or test data.

> **Caution:** Do not upgrade the actual production database until after you successfully upgrade a test subset of this database and test it with applications, as described in the next step.

Make sure you upgrade any OCI and precompiler applications that you plan to use with your new Oracle Database. Then, you can test these applications on a sample database before upgrading your current production database. See "Upgrading Precompiler and OCI Applications" on page 6-2 for more information.

## Test the Upgraded Test Database

Perform the planned tests on the current database and on the test database that you upgraded to the new Oracle Database release. Compare the results, noting anomalies. Repeat the test upgrade as many times as necessary.

Test the newly upgraded Oracle Database test database with existing applications to verify that they operate properly with a new Oracle Database. You also might test enhanced functionality by adding features that use the available Oracle Database functionality. However, first make sure that the applications operate in the same manner as they did in the current database.

**See Also:** Chapter 6, "Upgrading Your Applications" for more information on using applications with Oracle Database

# 3

# Upgrading to the New Oracle Database 10*g* Release

This chapter guides you through the process of upgrading a database to the new Oracle Database 10*g* release. This chapter covers the following topics:

- System Considerations and Requirements
- Install the Release 10.2 Oracle Software
- Install the Latest Available Patch Set Release and Any Required Patches
- Run the Pre-Upgrade Information Tool
- Run the Oracle Net Configuration Assistant
- Upgrade the Database Using the Database Upgrade Assistant
- Upgrade the Database Manually

> **See Also:** Some aspects of upgrading are operating system-specific. See your operating system-specific Oracle documentation for additional instructions about upgrading on your operating system.

## System Considerations and Requirements

The following sections discuss system considerations and requirements:

- Upgrading a Cluster Database
- Upgrading With Read-Only and Offline Tablespaces
- Upgrading Standby Databases
- Upgrading Your Operating System
- Migrating Data to a Different Operating System

### Upgrading a Cluster Database

If you are upgrading a cluster database, then most of the actions described in this chapter should be performed on only one node of the system. Actions that need to be performed on more than one node will be indicated in that particular step.

### Upgrading With Read-Only and Offline Tablespaces

The Oracle database can read file headers created prior to Oracle 10*g*, so you do not need to do anything to them during the upgrade. The only exception to this is if you want to transport tablespaces created prior to Oracle 10*g*, to another platform. In this

case, the file headers must be made read-write at some point before the transport. However, there are no special actions required on them during the upgrade.

The file headers of offline datafiles are updated later when they are brought online, and the file headers of read-only tablespaces are updated if and when they are made read-write sometime after the upgrade. In any other circumstance, read-only tablespaces never have to be made read-write.

> **See Also:** *Oracle Database Administrator's Guide* for more information about read-only tablespaces and transporting tablespaces between databases

## Upgrading Standby Databases

The methods by which you can perform an upgrade in an Oracle Data Guard configuration are described in the following sections:

- Upgrading Databases in an Oracle Data Guard Configuration
- Upgrading Databases in a Broker Configuration
- Using SQL Apply to Perform a Rolling Upgrade of Oracle Databases

### Upgrading Databases in an Oracle Data Guard Configuration

To upgrade the Oracle Database software when physical or logical standby databases are present in an Oracle Data Guard configuration, see the *Oracle Data Guard Concepts and Administration* documentation for the following topics:

- Upgrading to Oracle 10*g* with a Physical Standby Database in Place
- Upgrading to Oracle 10*g* with a Logical Standby Database in Place

### Upgrading Databases in a Broker Configuration

To upgrade or downgrade Oracle databases and Oracle Enterprise Manager in an Oracle Data Guard broker configuration, see *Oracle Data Guard Broker* for the following release scenarios:

- Upgrading from release 9.0.*n* to release 10.*n*
- Upgrading from release 9.2 to release 10.*n*
- Upgrading from release 10.1 to release 10.2
- Downgrading from release 10.2

### Using SQL Apply to Perform a Rolling Upgrade of Oracle Databases

Starting with Oracle Database 10*g* release 1 (10.1.0.3), you can use SQL Apply on a logical standby database to perform a *rolling upgrade* of the Oracle Database 10*g* software. During a rolling upgrade, you can run different releases of the Oracle database on the primary and logical standby databases while you upgrade them, one at a time, incurring minimal downtime on the primary database. See the *Oracle Data Guard Concepts and Administration* documentation for complete information.

## Upgrading Your Operating System

If required, upgrade the operating system before upgrading the Oracle database.

**See Also:**

- Your platform-specific Oracle Database installation guide to determine if you need to upgrade your operating system
- Your operating system-specific Oracle documentation for information on how to perform an operating system upgrade

### Migrating Data to a Different Operating System

When using the Database Upgrade Assistant or when performing a manual upgrade, you *cannot* migrate data in a database on one operating system to a database on another operating system. For example, you cannot migrate data in an Oracle9*i* database on Solaris to an Oracle Database 10*g* database on Windows 2000 using the Database Upgrade Assistant.

If you need to migrate Oracle Database to a different operating system, the best practice is to follow these steps:

1. Upgrade to the newest Oracle Database 10*g* release on the current operating system platform following the instructions in this book.
2. Use the Oracle Database 10*g* cross-platform transportable tablespace feature or the Oracle Data Pump Export and Import utilities to migrate the upgraded database to the different operating system.

## Install the Release 10.2 Oracle Software

Installation of the Release 10.2 Oracle software involves three basic steps: upgrading Cluster Ready Services (CRS) if necessary, installing Oracle Database, and installing Companion products if necessary.

1. If you are upgrading a Real Application Clusters (RAC) database, you must first install Oracle Clusterware (known as Cluster Ready Services prior to Release 10.2) from the product media. See the *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide* for your operating system for further instructions.
2. After installing Oracle Clusterware (assuming it was necessary), follow the instructions in your Oracle operating system-specific documentation to prepare for installation of Oracle Database and start the Oracle Universal Installer.

   When installation is complete, one or more assistants may be started. If you chose to run the Database Upgrade Assistant (DBUA) during installation, then you are ready to proceed with the upgrade when the Database Upgrade Assistant is started. However, it is recommended that you run the Pre-Upgrade Information Tool before you upgrade using DBUA, so that you can preview the types of items the DBUA will be checking. (See "Using the Pre-Upgrade Information Tool" on page 3-4.) You can then run the DBUA independently after the installation is complete.

   Note also that you must run the Oracle Net Configuration Assistant before running the Database Upgrade Assistant.

   When installation of Oracle Database has completed successfully, click the Exit button to close the Oracle Universal Installer.

**Note:** If you use Oracle Label Security, you must perform a custom install because it is not part of the standard installation.

3. After the installation of Oracle Database is complete, install the companion Oracle Database 10*g* Products if you have JServer, interMedia, Spatial, or Ultra Search in your existing databases.

**See Also:**

- "Run the Oracle Net Configuration Assistant" on page 3-11
- "Upgrade the Database Using the Database Upgrade Assistant" on page 3-11

# Install the Latest Available Patch Set Release and Any Required Patches

Once you have installed the major Oracle Database 10*g* software, check to see if there is a patch set release and/or critical patch update to be installed:

- Patch sets

  Patch sets are a software-release mechanism for delivering tested and integrated product fixes on a regular basis. Patch sets provide bug fixes only; they do not include new functionality, and do not require certification on the target system.

- Critical Patch Updates

  Critical patch updates contain security patch updates and security fixes. As of 2005, Oracle began providing Critical Patch Updates for all product offerings on a quarterly schedule. The comprehensive patches address significant security vulnerabilities and include fixes that are likely to be applied, or that are prerequisites for the security fixes.

The latest patch set release and critical patch update for Oracle Database 10*g* Release 2 should be installed prior to upgrading your databases. Refer to the specific patch release and critical patch update documentation for installation information.

# Run the Pre-Upgrade Information Tool

After you have installed the Oracle Database 10.2 software and any required patches, you should analyze your database before upgrading to the new Oracle Database 10*g* release. This is done by running the Pre-Upgrade Information Tool. This is a necessary step if you are upgrading manually. It is also recommended if you are upgrading with the Database Upgrade Assistant (DBUA), so that you can preview the types of items the DBUA will be checking.

## Using the Pre-Upgrade Information Tool

The Pre-Upgrade Information Tool is a SQL script that ships with the new Oracle Database 10*g* release, and must be copied to and run from the environment of the database being upgraded. Complete the following steps to run the Pre-Upgrade Information Tool:

1. Log in to the system as the owner of the Oracle home directory of the new Oracle Database 10*g* release.

2. Copy the following file from the *ORACLE_HOME*/rdbms/admin directory of the new Oracle Database 10*g* release to a directory outside of the Oracle home, such as the temporary directory on your system:

   - utlu102i.sql

   Make a note of the new location of this file.

3. Log in to the system as the owner of the Oracle home directory of the database to be upgraded.
4. Change to the directory outside of the Oracle home directory that you copied files to in Step 2.
5. Start SQL*Plus.
6. Connect to the database instance as a user with SYSDBA privileges.
7. Set the system to spool results to a log file for later analysis:

   ```
   SQL> SPOOL info.log
   ```

8. Run utlu102i.sql:

   ```
   SQL> @utlu102i.sql
   ```

9. Turn off the spooling of script results to the log file:

   ```
   SQL> SPOOL OFF
   ```

   Then, check the spool file and examine the output of the upgrade information tool. You named the spool file in Step 7; the suggested name was info.log.

The following is an example of the output generated by the Pre-Upgrade Information Tool:

```
Oracle Database 10.2 Upgrade Information Utility    05-25-2005 05:19:08
.
**********************************************************************
Database:
**********************************************************************
--> name:       RBX0
--> version:    9.2.0.6.0
--> compatible: 9.2.0
.
**********************************************************************
Logfiles: [make adjustments in the current environment]
**********************************************************************
--> The existing log files are adequate. No changes are required.
.
**********************************************************************
Tablespaces: [make adjustments in the current environment]
**********************************************************************
--> SYSTEM tablespace is adequate for the upgrade.
.... minimum required size: 583 MB
.... AUTOEXTEND additional space required: 163 MB
--> TEMP tablespace is adequate for the upgrade.
.... minimum required size: 58 MB
.... AUTOEXTEND additional space required: 38 MB
--> CWMLITE tablespace is adequate for the upgrade.
.... minimum required size: 16 MB
--> DRSYS tablespace is adequate for the upgrade.
.... minimum required size: 27 MB
.... AUTOEXTEND additional space required: 7 MB
```

```
--> EXAMPLE tablespace is adequate for the upgrade.
.... minimum required size: 150 MB
.... AUTOEXTEND additional space required: 1 MB
--> ODM tablespace is adequate for the upgrade.
.... minimum required size: 10 MB
--> XDB tablespace is adequate for the upgrade.
.... minimum required size: 48 MB
.... AUTOEXTEND additional space required: 3 MB
--> SYSAUX tablespace is adequate for the upgrade.
.... minimum required size: 61 MB
.... AUTOEXTEND additional space required: 11 MB
.
**************************************************************************
Update Parameters: [Update Oracle Database 10.2 init.ora or spfile]
**************************************************************************
WARNING: --> "shared_pool_size" needs to be increased to at least 178499994
WARNING: --> "java_pool_size" needs to be increased to at least 67108864
WARNING: --> "streams_pool_size" is not currently defined and needs a value of at
least 50331648
WARNING: --> "session_max_open_files" needs to be increased to at least 20
.
**************************************************************************
Deprecated Parameters: [Update Oracle Database 10.2 init.ora or spfile]
**************************************************************************
-- No deprecated parameters found. No changes are required.
.
**************************************************************************
Obsolete Parameters: [Update Oracle Database 10.2 init.ora or spfile]
**************************************************************************
--> "hash_join_enabled"
--> "max_enabled_roles"
.
**************************************************************************
Components: [The following database components will be upgraded or installed]
**************************************************************************
--> Oracle Catalog Views         [upgrade]  VALID
--> Oracle Packages and Types    [upgrade]  VALID
--> JServer JAVA Virtual Machine [upgrade]  VALID
...The 'JServer JAVA Virtual Machine' JAccelerator (NCOMP)
...is required to be installed from the 10g Companion CD.
--> Oracle XDK for Java          [upgrade]  VALID
--> Oracle Java Packages         [upgrade]  VALID
--> Oracle Text                  [upgrade]  VALID
--> Oracle XML Database          [upgrade]  VALID
--> Oracle Workspace Manager     [upgrade]  VALID
--> Oracle Data Mining           [upgrade]  VALID
--> Messaging Gateway            [upgrade]  VALID
--> OLAP Analytic Workspace      [upgrade]  UPGRADED
--> OLAP Catalog                 [upgrade]  VALID
--> Oracle OLAP API              [upgrade]  UPGRADED
--> Oracle interMedia            [upgrade]  VALID
...The 'Oracle interMedia Image Accelerator' is
...required to be installed from the 10g Companion CD.
--> Spatial                      [upgrade]  VALID
--> Oracle Ultra Search          [upgrade]  VALID
... To successfully upgrade Ultra Search, install it from
... the 10g Companion CD.
--> Oracle Label Security        [upgrade]  VALID
... To successfully upgrade Oracle Label Security, perform
... a Custom install and select the OLS option.
```

```
.
**************************************************************************
Miscellaneous Warnings
**************************************************************************
WARNING: --> Workspace Manager replication is in use.
.... Drop OWM replication support before upgrading:
.... EXECUTE dbms_wm.DropReplicationSupport;
WARNING: --> Passwords exist in some database links.
.... Passwords will be encrypted during the upgrade.
.... Downgrade of database links with passwords is not supported.
WARNING: --> Deprecated CONNECT role granted to some user/roles.
.... CONNECT role after upgrade has only CREATE SESSION privilege.
WARNING: --> Database contains stale optimizer statistics.
.... Refer to the 10g Upgrade Guide for instructions to update
.... statistics prior to upgrading the database.
.... Component Schemas with stale statistics:
....   SYS
....   XDB
....   OLAPSYS
....   MDSYS
WARNING: --> Database contains INVALID objects prior to upgrade.
.... USER SYS has 1 INVALID objects.
WARNING: --> Database contains globally authenticated users.
.... Refer to the 10g Upgrade Guide to upgrade SSL users.
WARNING: --> OLS requires post-upgrade action to update policy triggers.
.... Run rdbms/admin/olstrig.sql after the upgrade.
.
**************************************************************************
SYSAUX Tablespace:
[Create tablespace in the Oracle Database 10.2 environment]
**************************************************************************
--> New "SYSAUX" tablespace
.... minimum required size for database upgrade: 500 MB
.
```

The following sections describe the output of the Pre-Upgrade Information Tool.

### Database

This section displays global database information about the current database, such as the database name, release number, and compatibility level. A warning is displayed if the `COMPATIBLE` initialization parameter needs to be adjusted before the database is upgraded.

### Logfiles

This section displays a list of redo log files in the current database whose size is less than 4 MB. For each log file, the file name, group number, and recommended size is displayed. New files of at least 4 MB (preferably 10 MB) need to be created in the current database. Any redo log files less than 4 MB must be dropped before the database is upgraded.

### Tablespaces

This section displays a list of tablespaces in the current database. For each tablespace, the tablespace name and minimum required size is displayed. In addition, a message is displayed if the tablespace is adequate for the upgrade. If the tablespace does not have enough free space, then space must be added to the tablespace in the current database. Tablespace adjustments need to be made before the database is upgraded.

### Update Parameters

This section displays a list of initialization parameters in the parameter file of the current database that must be adjusted before the database is upgraded. The adjustments need to be made to the parameter file after it is copied to the new Oracle Database 10*g* release.

> **See Also:** Appendix A, "Initialization Parameter and Data Dictionary Changes" for more information about changes to initialization parameters in the new Oracle Database 10*g* release

### Deprecated Parameters

This section displays a list of initialization parameters in the parameter file of the current database that are deprecated in the new Oracle Database 10*g* release.

> **See Also:** "Deprecated Initialization Parameters" on page A-1 for a list of initialization parameters that are deprecated in the new Oracle Database 10*g* release

### Obsolete Parameters

This section displays a list of initialization parameters in the parameter file of the current database that are obsolete in the new Oracle Database 10*g* release. Obsolete initialization parameters need to be removed from the parameter file before the database is upgraded.

> **See Also:** "Obsolete Initialization Parameters" on page A-2 for a list of initialization parameters that are obsolete in the new Oracle Database 10*g* release

### Components

This section displays a list of database components in the new Oracle Database 10*g* release that will be upgraded or installed when the current database is upgraded.

### Miscellaneous Warnings

This section provides warnings about specific situations that may require attention before and/or after the upgrade.

### SYSAUX Tablespace

This section displays the minimum required size for the `SYSAUX` tablespace, which is required in Oracle Database 10*g*. The `SYSAUX` tablespace must be created after the new Oracle Database 10*g* release is started and BEFORE the upgrade scripts are invoked.

## Issues Requiring Further Analysis Prior to Upgrading

If the Pre-Upgrade Utility displays a warning about any of the following issues, then further analysis of the database is recommended prior to upgrading to Oracle Database 10*g*:

- Deprecated CONNECT Role
- Database Links With Passwords
- TIMESTAMP WITH TIMEZONE Datatype
- Release 8.1.7 National Character Set
- Optimizer Statistics

### Deprecated CONNECT Role

After upgrading to Oracle Database 10g, the CONNECT role will only have the CREATE SESSION privilege; the other privileges granted to the CONNECT role in earlier releases will be revoked during the upgrade. To identify which users and roles in your database are granted the CONNECT role, use the following query:

```
SELECT grantee FROM dba_role_privs
     WHERE granted_role = 'CONNECT' and
                     grantee NOT IN (
                                'SYS', 'OUTLN', 'SYSTEM', 'CTXSYS', 'DBSNMP',
                                'LOGSTDBY_ADMINISTRATOR', 'ORDSYS',
                                'ORDPLUGINS',  'OEM_MONITOR', 'WKSYS', 'WKPROXY',
                                'WK_TEST', 'WKUSER', 'MDSYS', 'LBACSYS', 'DMSYS',
                                'WMSYS',  'OLAPDBA', 'OLAPSVR', 'OLAP_USER',
                                  'OLAPSYS', 'EXFSYS', 'SYSMAN', 'MDDATA',
                                 'SI_INFORMTN_SCHEMA', 'XDB', 'ODM');
```

If users or roles require privileges other than CREATE SESSION, then grant the specific required privileges prior to upgrading. The upgrade scripts adjust the privileges for the Oracle-supplied users.

### Database Links With Passwords

During the upgrade to Oracle Database 10*g*, any passwords in database links will be encrypted. To downgrade back to the original release, all of the database links with encrypted passwords must be dropped prior to the downgrade. Consequently, the database links will not exist in the downgraded database. If you anticipate a requirement to be able to downgrade back to your original release, then save the information about affected database links from the SYS.LINK$ table, so that you can recreate the database links after the downgrade.

### TIMESTAMP WITH TIMEZONE Datatype

The time zone files that are supplied with Oracle Database 10*g* have been updated from version 1 to version 2 to reflect changes in transition rules for some time zone regions. The changes may affect existing data of TIMESTAMP WITH TIME ZONE datatype. To preserve this TIMESTAMP data for updating according to the new time zone transition rules, you must run the utltzuv2.sql script on the database before upgrading. This script is located in the new 10.2 ORACLE_HOME/rdbms/admin directory. This script analyzes your database for TIMESTAMP WITH TIME ZONE columns that are affected by the updated time zone transition rules.

> **See Also:** *Oracle Database Globalization Support Guide* for a detailed description of the utltzuv.sql script

If the utltzuv2.sql script identifies columns with time zone data affected by a database upgrade, then back up the data in character format before you upgrade the database. After the upgrade, you must update the tables to ensure that the data is stored based on the new rules. If you export the tables before upgrading and import them after the upgrade, the conversion will happen automatically during the import. Alternatively, create tables with the time zone information in character format (for example, TO_CHAR(column, 'YYYY-MM-DD HH24.MI.SSXFF TZR'), and recreate the TIMESTAMP data from these tables after the upgrade.

### Release 8.1.7 National Character Set

Starting in Oracle9*i*, the SQL NCHAR datatypes (NCHAR, NVARCHAR2, and NCLOB) are limited to the Unicode character set encoding (UTF8 and AL16UTF16) only. Any other

version 8 character sets that were available under the NCHAR datatype, including Asian character sets (such as JA16SJISFIXED), are no longer supported.

Before migrating your 8.1.7 SQL NCHAR data to the new Unicode NCHAR, Oracle Corporation recommends that you analyze your SQL NCHAR data, using the Character Set Scanner for the identification of possible invalid character set conversion or data truncation.

> **See Also:** *Oracle Database Globalization Support Guide* for more information about the Character Set Scanner

When you upgrade to Oracle Database 10*g*, the value of the National Character Set of the upgraded database is set based on the value of the National Character Set of the Oracle8*i* database being upgraded.

If the old National Character Set is UTF8, then the new National Character Set will be UTF8. Otherwise, the National Character Set is changed to AL16UTF16.

During the upgrade, the existing NCHAR columns in the data dictionary are changed to use the new format and, if the National Character Set has been changed to AL16UTF16, the dictionary NCHAR columns will be converted to the AL16UTF16 character set.

---

**Note:** NCHAR columns in user tables are not changed during the upgrade. For information about changing NCHAR columns in user tables, see "Upgrade User NCHAR Columns" on page 4-8.

---

---

**Note:** Be aware that there may be additional character set considerations if you are upgrading from a release 8.1 database. For example, if you use XDK for Java and have escape characters in your XML data, you should change your database character set to AL32UTF8 before you upgrade to Oracle Database 10*g*. See *Oracle XML DB Developer's Guide* for further information. For information on changing your database character set, see *Oracle Database Globalization Support Guide*.

---

### Optimizer Statistics

When upgrading to Oracle Database 10*g*, optimizer statistics are collected for dictionary tables that lack statistics. This statistics collection can be time consuming for databases with a large number of dictionary tables, but statistics gathering only occurs for those tables that lack statistics or are significantly changed during the upgrade.

To decrease the amount of downtime incurred when collecting statistics, you can collect statistics prior to performing the actual database upgrade. As of Oracle Database 10*g* Release 10.1, Oracle recommends that you use the DBMS_STATS.GATHER_DICTIONARY_STATS procedure to gather these statistics. For example, you can enter the following:

```
EXEC DBMS_STATS.GATHER_DICTIONARY_STATS:
```

If you are using Release 9.0.1 or 9.2.0, then you should use the DBMS_STATS.GATHER_SCHEMA_STATS procedure to gather statistics. To do this, you can run the scripts provided in Appendix C.

Table 3–1 lists the system components and schemas.

***Table 3–1 Statistics Collection for System Components and Schemas***

| Component Name | Schema |
|---|---|
| JServer JAVA Virtual Machine | SYS |
| OLAP Analytic Workspace | SYS |
| OLAP Catalog | OLAPSYS |
| Oracle Data Mining | DMSYS |
| Oracle Database Catalog Views | SYS |
| Oracle Database JAVA Packages | SYS |
| Oracle Database Packages and Types | SYS, DBSNMP, OUTLN, SYSTEM |
| Oracle Enterprise Manager | SYSMAN |
| Oracle Expression Filter | EXFSYS |
| Oracle Intermedia | ORDSYS, ORDPLUGINS, SI_INFORMTN_ SCHEMA |
| Oracle Label Security | LBACSYS |
| Oracle OLAP API | SYS |
| Oracle Spatial | MDSYS, MDDATA |
| Oracle Text | CTXSYS |
| Oracle Ultra Search | WKSYS, WKPROXY,WK_TEST |
| Oracle Workspace Manager | WMSYS |
| Oracle XDK | SYS |
| Oracle XML Database | XDB |

## Run the Oracle Net Configuration Assistant

If you are upgrading from Oracle9*i* and a listener was not configured in the Oracle9*i* repository, run Oracle Net Configuration Assistant to configure the listening protocol address and service information for the new Oracle Database 10*g* database, including a `listener.ora` file. A version 10 listener is required for an Oracle Database 10*g* database. Previous versions of the listener are not supported for use with an Oracle Database 10*g* database. However, it is possible to use a version 10 listener with previous versions of the Oracle database.

---

**Note:** It is important to run Oracle Net Configuration Assistant *before* running the Database Upgrade Assistant (DBUA).

---

**See Also:** *Oracle Database Net Services Administrator's Guide* for complete information about using Oracle Net Configuration Assistant

## Upgrade the Database Using the Database Upgrade Assistant

The following sections guide you through the process of upgrading a database using the Database Upgrade Assistant (DBUA). (Note also that you must run the Oracle Net Configuration Assistant before running the Database Upgrade Assistant.)

The DBUA provides a graphical user interface (GUI) to guide you through the upgrade of a database, or you can invoke it in silent mode, which does not present a user interface:

- Using the Database Upgrade Assistant Graphical User Interface
- Using the Database Upgrade Assistant in Silent Mode

---

**Note:** If the database instance is not running, the DBUA will try to start the instance with the default initialization parameter file. If that fails, you will be prompted to provide the name of the correct initialization parameter file or to start the instance. If the instance is already up and running, the DBUA connects to it.

---

---

**Note:** If you abort the upgrade, but do not restore the database, then you should not restart the DBUA until you start up the existing database in `UPGRADE` mode using the 10.2 server. You cannot go back to the original server unless you restore your database.

If you restore your database manually (not using the DBUA), then remove the following file before starting the DBUA: `$10.2OracleHome/cfgtoollogs/dbua/logs/Welcome_<SID>.txt`. The presence of this file indicates to the DBUA that this is a re-run operation.

---

## Using the Database Upgrade Assistant Graphical User Interface

If you installed the new Oracle Database 10*g* release and specified that you are upgrading an existing database, then the Database Upgrade Assistant is started automatically. However, if you did not specify that you are upgrading an existing database, then you can start the Database Upgrade Assistant independently after installation is complete.

In the environment of the new Oracle Database 10*g* release, start the Database Upgrade Assistant as follows:

- **On UNIX platforms**, enter the following command at a system prompt:

  ```
  dbua
  ```

---

**Note:** The `dbua` executable is usually located in *`ORACLE_HOME`*`/bin`.

---

- **On Windows operating systems**, choose:

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

When the Database Upgrade Assistant starts, its Welcome screen appears. Figure 3–1 shows the Welcome screen of the Database Upgrade Assistant. Before the upgrade, the DBUA performs the following steps:

- Check for any invalid user accounts or roles
- Check for any invalid datatypes or invalid objects

- Check for any desupported character sets
- Check for adequate resources, including rollback segments, tablespaces, and free disk space
- Check for any missing SQL scripts needed for the upgrade
- Optionally, DBUA backs up all necessary files

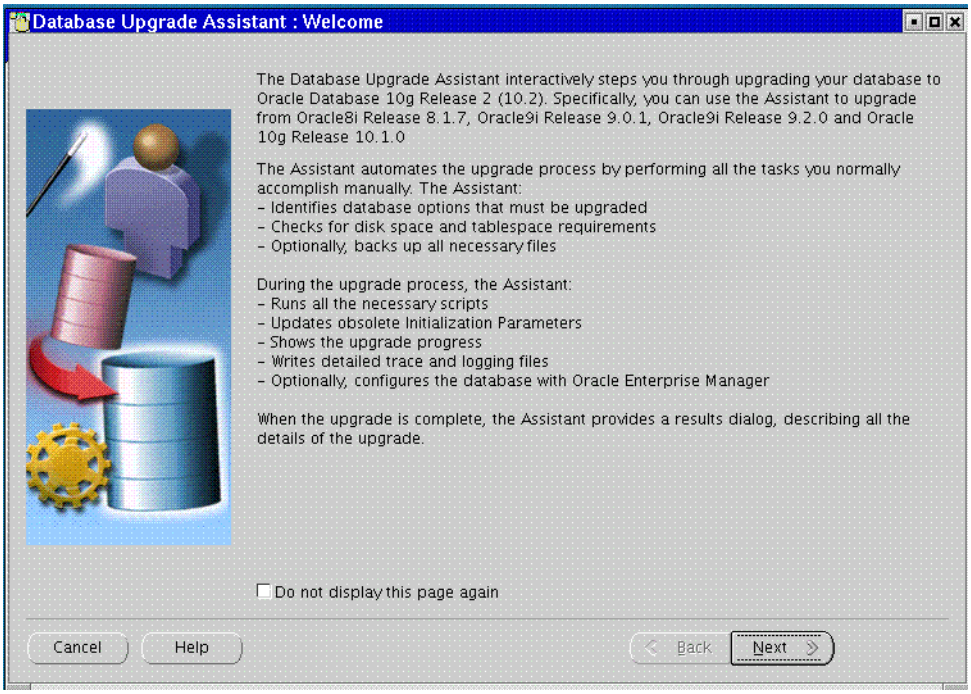
The DBUA does not begin the upgrade until all of these pre-upgrade steps are completed.

During the upgrade, the DBUA automatically modifies or creates new required tablespaces and invokes the appropriate upgrade scripts.

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

If you need detailed instructions on using the DBUA graphical user interface, see Appendix D, "Using the Database Upgrade Assistant".

***Figure 3–1 Welcome Screen of the Database Upgrade Assistant***




## Using the Database Upgrade Assistant in Silent Mode

When invoked with the `-silent` command line option, the Database Upgrade Assistant operates in silent mode. In silent mode, the Database Upgrade Assistant does not present a user interface. It also writes any messages (including information, errors, and warnings) to a log file.

For example, the following command upgrades a database named `ORCL` in silent mode:

```
dbua -silent -dbName ORCL &
```

### Database Upgrade Assistant Command Line Options

The Database Upgrade Assistant supports several command line options. You can specify all valid options from the command line using the following syntax:

```
dbua [ -silent ] [ -dbName SID ]
 [ -disableUpgradeScriptLogging ] [ -backupLocation directory ]
 [ -postUpgradeScripts script [, script ] ... ]
 [ -initParam parameter=value [, parameter=value ] ... ]
 [ -emConfiguration { LOCAL | CENTRAL | NOBACKUP | NOEMAIL | NONE }
 -dbsnmpPassword password -sysmanPassword password
 [ -hostUserName hostname -hostUserPassword password -backupSchedule hh:mm ]
 ]
 [ -smtpServer server_name -emailAddress address ]
 -centralAgent location
 ]
 [ -recoveryAreaDestination directory ] [ -h | -help ]
```

Table 3–2 describes the various options and their parameters that are supported by the Database Upgrade Assistant.

***Table 3–2 Database Upgrade Assistant Command Line options***

| Option | Description |
|---|---|
| `-silent` | Specifies that the Database Upgrade Assistant should operate in silent mode. See "Using the Database Upgrade Assistant in Silent Mode" on page 3-13. |
| `-dbName` *SID* | Specifies the system identifier (SID) of the database to upgrade |
| `-disableUpgradeScriptLoggin g` | This option disables the detailed log generation for running SQL scripts during the upgrade process. This is enabled by default. To enable the log generation, do not specify this option. |
| `-backupLocation` *directory* | Specifies a directory to back up your database before the upgrade starts |
| `-postUpgradeScripts` *script* `[,` *script* `] ...` | Specifies a comma-separated list of SQL scripts. Specify complete path names. The scripts will be executed at the end of the upgrade. |
| `-initParam` *parameter=value* `[,` *parameter=value* `] ...` | Specifies a comma-separated list of initialization parameter values of the form *name=value* |
| `-emConfiguration { LOCAL \| CENTRAL \| NOBACKUP \| NOEMAIL \| NONE }` | Specifies Enterprise Manager management options:<br>■ `LOCAL` - Database is locally managed by Enterprise Manager<br>■ `CENTRAL` - Database is centrally managed by Enterprise Manager<br>■ `NOBACKUP` - Automatic daily backups of the database are not enabled<br>■ `NOEMAIL` - E-mail notifications are not enabled<br>■ `NONE` - Database is not managed by Enterprise Manager |
| `-dbsnmpPassword` *password* | Specifies the `DBSNMP` user password |
| `-sysmanPassword` *password* | Specifies the `SYSMAN` user password |
| `-hostUserName` *hostname* | Specifies the host user name for the Enterprise Manager backup job |

*Table 3–2 (Cont.) Database Upgrade Assistant Command Line options*

| Option | Description |
|---|---|
| `-hostUserPassword` *password* | Specifies the host user password for the Enterprise Manager backup job |
| `-backupSchedule` *hh*:*mm* | Specifies the daily backup schedule in the form *hh*:*mm* (hours and minutes) |
| `-smtpServer` *server_name* | Specifies the outgoing mail (SMTP) server for E-mail notifications |
| `-emailAddress` *address* | Specifies the E-mail address for E-mail notifications |
| `-centralAgent` *location* | Specifies the Enterprise Manager central agent location |
| `-recoveryAreaDestination` *directory* | Specifies the destination directory for all recovery files |
| `-h \| -help` | Displays usage help for the Database Upgrade Assistant |

# Upgrade the Database Manually

The following sections guide you through the process of performing a manual upgrade. They assume that you have already run the Pre-Upgrade Information Tool as described in "Using the Pre-Upgrade Information Tool" on page 3-4.



## Back Up the Database

After cleanly shutting down the instance following the analysis of the database, you should perform a full backup of the database. Complete the following steps:

1. Sign on to RMAN:

```
rman "target / nocatalog"
```

2. Issue the following RMAN commands:

```
RUN
{
    ALLOCATE CHANNEL chan_name TYPE DISK;
    BACKUP DATABASE FORMAT 'some_backup_directory%U' TAG before_upgrade;
    BACKUP CURRENT CONTROLFILE TO 'save_controlfile_location';
}
```

**Caution:** If you encounter problems with the upgrade and wish to abandon the upgrade completely, then you will need to restore the database from this backup. Therefore, make sure you back up your database now as a precaution.

**See Also:** *Oracle Database Backup and Recovery Basics* for more information about backing up a database

## Prepare the New Oracle Home

After analyzing the database to be upgraded, complete the following steps to prepare the new Oracle home:

1. Copy configuration files from the Oracle home of the database being upgraded to the new Oracle Database 10*g* Oracle home:

   a. If your parameter file resides within the old environment's Oracle home, then copy it to the new Oracle home. By default, Oracle looks for the parameter file in *ORACLE_HOME*/dbs on UNIX platforms and in *ORACLE_HOME*\database on Windows operating systems. The parameter file can reside anywhere you wish, but it should not reside in the old environment's Oracle home after you upgrade to Oracle Database 10*g*.

      **Tip:** It may be necessary to create a text initialization parameter file (pfile) from the server parameter file (spfile) so that you can edit the initialization parameters.

   b. If your parameter file is a text-based initialization parameter file with either an IFILE (include file) or a SPFILE (server parameter file) entry, and the file specified in the IFILE or SPFILE entry resides within the old environment's Oracle home, then copy the file specified by the IFILE or SPFILE entry to the new Oracle home. The file specified in the IFILE or SPFILE entry contains additional initialization parameters.

   c. If you have a password file that resides within the old environment's Oracle home, then move or copy the password file to the new Oracle Database 10*g* Oracle home.

      The name and location of the password file are operating system-specific. On UNIX platforms, the default password file is *ORACLE_HOME*/dbs/orapw*sid*. On Windows operating systems, the default password file is *ORACLE_HOME*\database\pwd*sid*.ora. In both cases, *sid* is your Oracle instance ID.

   d. If you are upgrading a cluster database and your init*db_name*.ora file resides within the old environment's Oracle home, then move or copy the init*db_name*.ora file to the new Oracle home.

      **Note:** If you are upgrading a cluster database, then perform this step on all nodes in which this cluster database has instances configured.

2. Adjust your parameter file in the new Oracle Database 10*g* release by completing the following steps:

   a. Remove obsolete initialization parameters and adjust deprecated initialization parameters. Certain parameters are obsolete in the new Oracle Database 10*g* release, while other parameters have become deprecated. Remove all obsolete parameters from any parameter file that will start a release 10.2 instance. Obsolete parameters may cause errors in the new Oracle Database 10*g* release. Also, alter any parameter whose syntax has changed in the new Oracle Database 10*g* release.

The Pre-Upgrade Information Tool displays any deprecated parameters and obsolete parameters it finds in the **Deprecated Parameters** and **Obsolete Parameters** sections, respectively.

**See Also:** Appendix A, "Initialization Parameter and Data Dictionary Changes" for a list of initialization parameters that have been deprecated or have become obsolete

**b.** Make sure the `COMPATIBLE` initialization parameter is properly set for the new Oracle Database 10*g* release. The Upgrade Information Tool displays a warning in the **Database** section if `COMPATIBLE` is not properly set.

**c.** Adjust the values of the initialization parameters to at least the minimum value indicated the Pre-Upgrade Information utility.

**d.** Make sure all path names in the parameter file are fully specified. You should not have relative path names in the parameter file.

**e.** If the parameter file contains an `IFILE` entry, then change the `IFILE` entry in the parameter file to point to the new location of the include file that you specified in Step 1. b. Then, edit the file specified in the `IFILE` entry in the same way that you edited the parameter file in Step a through Step d.

**f.** If you are upgrading a cluster database, then modify the `init`*`db_name`*`.ora` file in the same way that you modified the parameter file.

Make sure you save all of the files you modified after making these adjustments.

---

**Note:** If you are upgrading a cluster database, then perform this step on all nodes in which this cluster database has instances configured.

---

**3.** If you are upgrading a cluster database, then set the `CLUSTER_DATABASE` initialization parameter to `false`. After the upgrade, you must set this initialization parameter back to `true`.

## Upgrade the Database

After preparing the new Oracle home, you are ready to proceed with the manual upgrade. Complete the following steps to upgrade the database:

**1.** Shut down the instance:

```
SQL> SHUTDOWN IMMEDIATE
```

If your operating system is Windows, then complete the following steps:

**a.** Stop the `OracleService`*`SID`* Oracle service of the database you are upgrading, where *`SID`* is the instance name. For example, if your *`SID`* is `ORCL`, then enter the following at a command prompt:

```
C:\> NET STOP OracleServiceORCL
```

**b.** Delete the Oracle service at a command prompt using ORADIM.

If your *`SID`* is `ORCL`, then enter the following command:

```
C:\> ORADIM -DELETE -SID ORCL
```

**c.** Create the new Oracle Database 10*g* service at a command prompt using the ORADIM command of the new Oracle Database release:

```
C:\> ORADIM -NEW -SID SID -INTPWD PASSWORD -MAXUSERS USERS
     -STARTMODE AUTO -PFILE ORACLE_HOME\DATABASE\INITSID.ORA
```

This syntax includes the following variables:

| Variable | Description |
|---|---|
| *SID* | The same SID name as the SID of the database you are upgrading. |
| *PASSWORD* | The password for the new release 10.2 database instance. This is the password for the user connected with SYSDBA privileges. The -INTPWD option is not required. If you do not specify it, then operating system authentication is used, and no password is required. |
| *USERS* | The maximum number of users who can be granted SYSDBA and SYSOPER privileges. |
| *ORACLE_HOME* | The release 10.2 Oracle home directory. Ensure that you specify the full path name with the -PFILE option, including drive letter of the Oracle home directory. |

For example, if your *SID* is ORCL, your *PASSWORD* is TWxy579, the maximum number of *USERS* is 10, and the *ORACLE_HOME* directory is C:\ORA92, then enter the following command:

```
C:\> ORADIM -NEW -SID ORCL -INTPWD TWxy579 -MAXUSERS 10
     -STARTMODE AUTO -PFILE C:\ORA92\DATABASE\INITORCL.ORA
```

2. If your operating system is UNIX, then make sure that your ORACLE SID is set correctly and that the following environment variables point to the new release 10.2 directories:
   - ORACLE_HOME
   - PATH
   - ORA_NLS10
   - LD_LIBRARY_PATH

   **Note:** If you are upgrading a cluster database, then perform this step on all nodes in which this cluster database has instances configured.

   **See Also:** Your operating system-specific Oracle Database installation documents for information about setting other important environment variables on your operating system.

3. Log in to the system as the owner of the Oracle home directory of the new Oracle Database 10*g* release.
4. At a system prompt, change to the *ORACLE_HOME*/rdbms/admin directory.
5. Start SQL*Plus.
6. Connect to the database instance as a user with SYSDBA privileges.
7. Start up the instance by issuing the following command:

```
SQL> STARTUP UPGRADE
```

> **Note:** The UPGRADE keyword allows you to open a pre-10.2 database. It also restricts logons to AS SYSDBA sessions, disables system triggers, and performs additional operations that prepare the environment for the upgrade.

You may need to use the PFILE option to specify the location of your initialization parameter file.

The following are common errors that may occur when attempting to start the new Oracle Database 10*g* release. If you receive any of these errors, issue the SHUTDOWN ABORT command to shut down the database and correct the problem.

- If the COMPATIBLE initialization parameter is set to a value less than "9.2.0":

  ```
  ORA-00401: the value for parameter compatible is not supported by this
  release
  ```

- If the CLUSTER_DATABASE initialization parameter is set to `true` instead of `false`:

  ```
  ORA-39701: database must be mounted EXCLUSIVE for UPGRADE or DOWNGRADE
  ```

- If the STARTUP command was issued without the UPGRADE keyword:

  ```
  ORA-39700: database must be opened with UPGRADE option
  ```

- If a redo log's size is less than 4 MB:

  ```
  ORA-00336: log file size xxxx blocks is less than minimum 8192 blocks
  ```

If errors appear listing obsolete initialization parameters, then make a note of the obsolete initialization parameters and continue with the upgrade. Then, remove the obsolete initialization parameters the next time you shut down the database.

8. If you are upgrading from release 10.1, then skip to step 9. Otherwise, if you are upgrading from release 8.1.7, 9.0.1, or 9.2.0, then create a SYSAUX tablespace. In Oracle Database 10*g*, the SYSAUX tablespace is used to consolidate data from a number of tablespaces that were separate in previous releases.

   The SYSAUX tablespace must be created with the following mandatory attributes:

   - ONLINE
   - PERMANENT
   - READ WRITE
   - EXTENT MANAGEMENT LOCAL
   - SEGMENT SPACE MANAGEMENT AUTO

   The Pre-Upgrade Information Tool provides an estimate of the minimum required size for the SYSAUX tablespace in the **SYSAUX Tablespace** section. Table 3–3 can be used to determine an optimal size for the SYSAUX tablespace.

***Table 3–3 Guidelines for Sizing the SYSAUX Tablespace***

| Factor | Small | Medium | Large |
|---|---|---|---|
| Number of CPUs | 2 | 8 | 32 |
| Number of concurrently active sessions | 5 | 20 | 100 |

*Table 3–3 (Cont.) Guidelines for Sizing the SYSAUX Tablespace*

| Factor | Small | Medium | Large |
|---|---|---|---|
| Number of user objects (tables and indexes) | 500 | 5,000 | 50,000 |
| Estimated SYSAUX size at steady state with default config | 500 MB | 2 GB | 5 GB |

The following SQL statement would create a 500 MB `SYSAUX` tablespace for the database:

```
SQL> CREATE TABLESPACE sysaux DATAFILE 'sysaux01.dbf'
         SIZE 500M REUSE
         EXTENT MANAGEMENT LOCAL
         SEGMENT SPACE MANAGEMENT AUTO
         ONLINE;
```

**See Also:** *Oracle Database Administrator's Guide* for more information about the `SYSAUX` tablespace

9. Set the system to spool results to a log file for later verification of success:

```
SQL> SPOOL upgrade.log
```

10. Run `catupgrd.sql`:

```
SQL> @catupgrd.sql
```

The `catupgrd.sql` script determines which upgrade scripts need to be run and then runs each necessary script. You must run the script in the new release 10.2 environment.

The upgrade script creates and alters certain data dictionary tables. It also upgrades or installs the following database components in the new release 10.2 database:

- Oracle Database Catalog Views
- Oracle Database Packages and Types
- JServer JAVA Virtual Machine
- Oracle Database Java Packages
- Oracle XDK
- Oracle Real Application Clusters
- Oracle Workspace Manager
- Oracle interMedia
- Oracle XML Database
- OLAP Analytic Workspace
- Oracle OLAP API
- OLAP Catalog
- Oracle Text
- Spatial

- Oracle Data Mining
- Oracle Label Security
- Messaging Gateway
- Expression Filter
- Oracle Enterprise Manager Repository

**11.** Run `utlu102s.sql` to display the results of the upgrade:

```
SQL> @utlu102s.sql
```

The Post-upgrade Status Tool displays the status of the database components in the upgraded database and the time required to complete each component upgrade. The Upgrade Status Tool displays output similar to the following:

```
Oracle Database 10.2 Upgrade Status Utility           04-20-2005 05:18:40
.
Component                               Status         Version  HH:MM:SS
Oracle Database Server                   VALID      10.2.0.1.0  00:11:37
JServer JAVA Virtual Machine             VALID      10.2.0.1.0  00:02:47
Oracle XDK                               VALID      10.2.0.1.0  00:02:15
Oracle Database Java Packages            VALID      10.2.0.1.0  00:00:48
Oracle Text                              VALID      10.2.0.1.0  00:00:28
Oracle XML Database                      VALID      10.2.0.1.0  00:01:27
Oracle Workspace Manager                 VALID      10.2.0.1.0  00:00:35
Oracle Data Mining                       VALID      10.2.0.1.0  00:15:56
Messaging Gateway                        VALID      10.2.0.1.0  00:00:11
OLAP Analytic Workspace                  VALID      10.2.0.1.0  00:00:28
OLAP Catalog                             VALID      10.2.0.1.0  00:00:59
Oracle OLAP API                          VALID      10.2.0.1.0  00:00:53
Oracle interMedia                        VALID      10.2.0.1.0  00:08:03
Spatial                                  VALID      10.2.0.1.0  00:05:37
Oracle Ultra Search                      VALID      10.2.0.1.0  00:00:46
Oracle Label Security                    VALID      10.2.0.1.0  00:00:14
Oracle Expression Filter                 VALID      10.2.0.1.0  00:00:16
Oracle Enterprise Manager                VALID      10.2.0.1.0  00:00:58
.
Total Upgrade Time: 00:56:09
```

**12.** Turn off the spooling of script results to the log file:

```
SQL> SPOOL OFF
```

Then, check the spool file and verify that the packages and procedures compiled successfully. You named the spool file in Step 9; the suggested name was `upgrade.log`. Correct any problems you find in this file (see "Troubleshoot the Upgrade" on page 3-22) and rerun the `catupgrd.sql` script, if necessary. You can rerun any of the scripts described in this chapter as many times as necessary. (If you have JAVAVM in your database, but you did not install the companion Oracle Database 10*g* Products, you will most likely receive error message ORA-29558.)

**13.** Shut down and restart the instance to reinitialize the system parameters for normal operation.

```
SQL> SHUTDOWN IMMEDIATE
SQL> STARTUP
```

Cleanly shutting down and restarting the instance flushes all caches, clears buffers, and performs other housekeeping activities. These measures are an important final

step to ensure the integrity and consistency of the newly upgraded Oracle Database.

Also, if you encountered a message listing obsolete initialization parameters when you started the database in Step 7, then remove the obsolete initialization parameters from the parameter file before restarting. If necessary, convert the spfile to a pfile so you can edit the file to delete parameters.

14. Run `olstrig.sql` to re-create DML triggers on tables with Oracle Label Security policies. (See *Oracle Database Enterprise User Administrator's Guide* for more information.) This step is only necessary if Oracle Label Security is in your database.

```
SQL> @olstrig.sql
```

15. Run `utlrp.sql` to recompile any remaining stored PL/SQL and Java code.

```
SQL> @utlrp.sql
```

Verify that all expected packages and classes are valid:

```
SQL> SELECT count(*) FROM dba_objects WHERE status='INVALID';
SQL> SELECT distinct object_name FROM dba_objects WHERE status='INVALID';
```

16. Exit SQL*Plus.

Your database is now upgraded to the new Oracle Database 10*g* release. Complete the procedures described in Chapter 4, "After Upgrading a Database".

**WARNING: If you retain the old Oracle software, then never start the upgraded database with the old software. Only start the database with the executables in the new Oracle Database installation. Also, before you remove the old Oracle environment, make sure you relocate any data files in that environment to the new Oracle Database environment. See the *Oracle Database Administrator's Guide* for information about relocating data files.**

## Troubleshoot the Upgrade

There are three resources that generally require increases for a new Oracle Database release:

- `SYSTEM` tablespace
- Shared memory
- Rollback segments/Undo Tablespace

If you run out of one of these resources during the upgrade, then increase the resource allocation. After increasing the resource allocation, you should perform a `SHUTDOWN ABORT` and restart the instance (in `UPGRADE` mode) before rerunning the `catupgrd.sql` script or restarting the Database Upgrade Assistant.

### SYSTEM and SYSAUX Tablespace

Typically you will receive one of the following messages during the upgrade if your `SYSTEM` tablespace size is insufficient:

```
ORA-01650: unable to extend rollback segment string by string in tablespace string
ORA-01651: unable to extend save undo segment by string for tablespace string
ORA-01652: unable to extend temp segment by string in tablespace string
```

```
ORA-01653: unable to extend table string.string by string in tablespace string
ORA-01654: unable to extend index string.string by string in tablespace string
ORA-01655: unable to extend cluster string.string by string in tablespace string
```

To avoid these errors, set `AUTO EXTEND ON MAXSIZE UNLIMITED` for the `SYSTEM` and `SYSAUX` tablespaces.

### Shared Memory

You may require larger shared memory pool sizes in some cases. The error message will indicate which shared memory initialization parameter needs to be increased.

```
ORA-04031: unable to allocate string bytes of shared memory
("string","string","string","string")
```

Refer to *Oracle Database Reference* for information about shared memory initialization parameters.

### Public Rollback Segment

If you are using rollback segments, then you need to have a single large (100 MB) `PUBLIC` rollback segment online while the upgrade scripts are being run. Smaller public rollback segments should be taken offline during the upgrade. Typically you will get the following error if your rollback segment size is insufficient:

```
ORA-01562: failed to extend rollback segment number string
```

## Abandon the Upgrade

If you completed the steps in "Back Up the Database" on page 3-15 to back up your database, then the easiest way to abandon the upgrade is to restore that backup. Complete the following steps:

1. Log in to the system as the owner of the Oracle home directory of the previous release.
2. Sign on to RMAN:

   ```
   rman "target / nocatalog"
   ```

3. Issue the following RMAN commands:

   ```
   STARTUP NOMOUNT
   RUN
   {
       REPLICATE CONTROLFILE FROM 'save_controlfile_location';
       ALTER DATABASE MOUNT;
       RESTORE DATABASE FROM TAG before_upgrade
       ALTER DATABASE OPEN RESETLOGS;
   }
   ```

# 4

# After Upgrading a Database

This chapter guides you through the procedures to perform after you have completed an upgrade of your database. The following topics are discussed:



## Tasks to Complete After All Upgrades

Complete the following tasks after you have upgraded your database, regardless of whether you performed the upgrade manually or by using the Database Upgrade Assistant (DBUA):



### Back Up the Database

Make sure you perform a full backup of the production database.

**See Also:** *Oracle Database Backup and Recovery Basics* for details about backing up a database

## Update Environment Variables After the Upgrade (UNIX Systems Only)

If your operating system is UNIX, then make sure that the following environment variables point to the new release 10.2 directories:

- ORACLE_HOME
- PATH
- ORA_NLS10

  (Note that the ORA_NLS10 environment variable replaces the ORA_NLS33 environment variable, so you may need to unset ORA_NLS33 and set ORA_NLS10.)
- LD_LIBRARY_PATH

**Note:** If you are upgrading a cluster database, then perform this step on all nodes in which this cluster database has instances configured.

**See Also:** Your operating system-specific Oracle Database installation documents for information about setting other important environment variables on your operating system.

## Set Threshold Values for Tablespace Alerts

An upgraded Oracle Database 10*g* database has the Tablespace Alerts disabled (the thresholds are set to null). Tablespaces in the database that are candidates for monitoring need to be identified and the appropriate threshold values set.

The default threshold values (for a newly created Oracle Database 10*g* database) are:

- 85% full warning
- 97% full critical

## Migrate Tables from the LONG Datatype to the LOB Datatype

LOB datatypes (BFILE, BLOB, CLOB, and NCLOB) can provide many advantages over LONG datatypes. See *Oracle Database Concepts* for information about the differences between LONG and LOB datatypes.

In Oracle9*i* release 9.0.1 and later, the ALTER TABLE statement can be used to change the datatype of a LONG column to CLOB and that of a LONG RAW column to BLOB.

In the following example, the LONG column named long_col in table long_tab is changed to datatype CLOB:

```
SQL> ALTER TABLE Long_tab MODIFY ( long_col CLOB );
```

After using this method to change LONG columns to LOBs, all the existing constraints and triggers on the table will still be usable. However, all the indexes, including Domain indexes and Functional indexes, on all columns of the table will become unusable and will have to be rebuilt using an ALTER INDEX ... REBUILD statement. Also, the Domain indexes on the LONG column will have to be dropped before changing the LONG column to a LOB.

**See Also:** *Oracle Database Application Developer's Guide - Large Objects* for information about modifying applications to use LOB data

## Upgrade the TIMESTAMP Data

If your database has `TIMESTAMP WITH TIMEZONE` data, you must update the data so that it is converted and stored based on the new time zone rules that come with the upgrade. (See "TIMESTAMP WITH TIMEZONE Datatype" on page 3-9).

If you used the export utility to export a copy of the affected tables, you should now use the import utility to import your data from these tables back into your database. The import utility will update the timestamp data as it imports.

If you used the manual script method, you will need to update the affected timestamp data based on your backed up table. For example, if you previously backed up your table, you need to run an update statement similar to the one below to update your timestamp data.

```
UPDATE tztab t SET t.y =
  (SELECT to_timestamp_tz(t1.y,'YYYY-MM-DD HH24.MI.SSXFF TZR')
FROM tztab_back t1
WHERE t.x=t1.x);
```

Although the transition rule changes for some time zone regions may affect data of `TIMESTAMP WITH LOCAL TIME ZONE` datatype, there is no way to upgrade the data. The data cannot be upgraded because this type does not preserve the original time zone/region associated with the data.

Time zone regions in Brazil and Israel may have frequent transition rules changes, perhaps as often as every year. Use the time zone offset instead of the time zone region name to avoid storing inconsistent data.

## Use the Latest Time Zone File for Clients

Customers using time zone regions that have been updated in version 2 of the time zone files are required to update all Oracle9*i* Database clients and databases that will communicate with an Oracle Database 10*g* server. This ensures that all environments will have the same version of the time zone file. Upgrading to the latest time zone file is not a requirement for customers that do not use the `TIMESTAMP WITH TIME ZONE` type or manipulate data from regions with frequent changes to their time zone transition rules. However Oracle recommends this action to avoid future issues. Users who need to update their time zone files to version 2 can find the following information on Oracle MetaLink (`http://metalink.oracle.com`):

- `readme.txt` contains the list of time zone regions that have changed from version 1 to version 2
- Actual time zone files for version 2 for the Oracle9*i* Database release

Oracle Database 10*g* clients that communicate with Oracle Database 10*g* servers automatically get version 2 of the time zone file, so there is no need to download the new time zone file.

**See Also:** `$ORACLE_HOME/oracore/zoneinfo/readme.txt` for detailed information about time zone file updates

## Upgrade the Recovery Catalog

For information about upgrading the recovery catalog, see *Oracle Database Backup and Recovery Advanced User's Guide*.

## Upgrade Statistics Tables Created by the DBMS_STATS Package

If you created statistics tables using the DBMS_STATS.CREATE_STAT_TABLE procedure, then upgrade these tables by executing the following procedure:

```
EXECUTE DBMS_STATS.UPGRADE_STAT_TABLE('scott', 'stat_table');
```

where SCOTT is the owner of the statistics table and STAT_TABLE is the name of the statistics table. Execute this procedure for each statistics table.

## Upgrade Externally Authenticated SSL Users

If you are using externally authenticated SSL users, you must run the following command to upgrade those users:

```
$ORACLE_HOME/rdbms/bin/extusrupgrade --dbconnectstring
<hostname:port_no:sid> --dbuser <db admin> --dbuserpassword
<password> -a
```

> **See Also:** *Oracle Database Enterprise User Administrator's Guide* for more information on the extusrupgrade script

## Install Supplied Knowledge Bases

The Supplied Knowledge Bases have been moved to be part of the companion Oracle Database 10*g* Products and are not immediately available after an upgrade to Oracle Database 10*g*. Any Text features dependent on the Supplied Knowledge Bases which were available before the upgrade will not function after the upgrade. To re-enable such features, you must install the Supplied Knowledge Bases from the installation media.

After an upgrade, all user-extensions to the Supplied Knowledge Bases must be regenerated. These changes affect all databases installed in the given ORACLE_HOME.

> **See Also:**
>
> - *Oracle Text Application Developer's Guide* for information about Supplied Knowledge Bases
> - The post-installation tasks section of your platform-specific Oracle Database installation guide for companion products for your platform

## Upgrade Change Data Capture

Beginning with Oracle 10*g* Release 10.2, Asynchronous Change Data Capture (CDC) no longer requires the same operating system for source and target databases. This feature enables a heterogeneous CDC setup with different operating systems and Oracle versions, enabling asynchronous CDC to leverage any existing Oracle9*i* Release 2 (9.2) system as a source.

See the *Oracle Database Data Warehousing Guide* for complete information about how to upgrade a release 9.2 or 10.1 Oracle Database to release 10.2 with Change Data Capture. The discussion describes the supported configurations for the Distributed HotLog mode of Change Data Capture as well as the restrictions.

## Configure Secure HTTP

If HTTPS access to Oracle XML DB is required, then you must provide correct configuration information, as described in this section.

When a database is upgraded to release 10.2, the XML schema for the XDB configuration file is automatically upgraded so that the XDB configuration file (located at `/xdbconfig.xml` in the repository) can have two additional elements, *`http2-port`* and *`http2-protocol`*. These elements are not added to the XDB configuration file by default during an upgrade. If you wish to have support for HTTPS, you must edit the configuration file to add these two new elements (see the XML schema for their exact locations), and to set the value of *`http2-protocol`* to `tcps`. The value of *`http2-port`* should be different from the value of *`http-port`*.

In addition to specifying the parameters *`http2-port`* and *`http2-protocol`* in the XDB configuration file, you must configure the database and the listener to enable Oracle XML DB to use HTTPS. Additionally, if the following steps were not taken before the upgrade, then you must perform them after the upgrade:

1. Enable the HTTP listener and the database to use SSL
2. Enable launching of a TCPS dispatcher

For more information on how to do this, see *Oracle XML DB Developer's Guide*.

## Provide Anonymous Access to XML DB Repository Data via HTTP

If anonymous access to XML DB repository data via HTTP is not required, then you do not have to perform this step. If anonymous access to XML DB repository data via HTTP is required, then you must provide correct configuration information, as described in this section. The administrator must carefully consider whether anonymous access is to be allowed, given the inherent security risks.

When a database is upgraded to Oracle 10g Release 2, the XML schema for the XML DB configuration file (located at `/xdbconfig.xml` in the repository) is automatically upgraded so that it can have an additional element, `allow-repository-anonymous-access`. This element is of Boolean type which means it can have a value of `true` or `false`. It can be used to disallow unauthenticated access to your Oracle XML DB Repository data through HTTP even if you unlock the `ANONYMOUS` user account. It is not added to the XML DB configuration file by default during an upgrade but when this element is missing, it is interpreted as `false`.

Therefore, anonymous access to XML DB repository data via HTTP is disabled when you upgrade to Oracle 10g Release 2. If you wish to have anonymous access to XML DB repository data via HTTP, you must change the configuration file to set this new element to `true`, in addition to unlocking the `ANONYMOUS` user account.

---

**Caution:** There is an inherent security risk associated with allowing unauthenticated access to the repository.

---

**See Also:** *Oracle XML DB Developer's Guide* for more information about the `allow-repository-anonymous-access` element and configuring Oracle XML DB

## Update Your HTML DB Configuration

To update your HTML DB configuration, you must complete a series of post-installation steps. These steps are described in the section on post-installation tasks in your platform-specific Oracle Database installation guide for companion products for your particular platform. In particular, refer to the sections on

Post-Installation Tasks for Oracle HTML DB and Post-Installation Tasks for Oracle HTTP Server.

### Add New Features as Appropriate

*Oracle Database New Features* describes many of the new features available in the new Oracle Database release. Determine which of these new features can benefit the database and applications; then, develop a plan for using these features.

It is not necessary to make any immediate changes to begin using your new Oracle Database. You may prefer to introduce these enhancements into your database and corresponding applications gradually.

Chapter 6, "Upgrading Your Applications" describes ways to enhance your applications so that you can take advantage of new Oracle Database features. However, before you implement new Oracle Database features, test your applications and successfully run them with the upgraded database.

### Develop New Administrative Procedures as Needed

After familiarizing yourself with new Oracle Database features, review your database administration scripts and procedures to determine whether any changes are necessary.

Coordinate your changes to the database with the changes that are necessary for each application. For example, by enabling integrity constraints in the database, you may be able to remove some data checking from your applications.

## Tasks to Complete Only After Manual Upgrades

If you are performing a manual upgrade rather than using the Database Upgrade Assistant (DBUA), then you must perform the following tasks after your database is upgraded:

- Change Passwords for Oracle-Supplied Accounts
- Migrate Your Initialization Parameter File to a Server Parameter File
- Upgrade Oracle Text
- Upgrade the Oracle Cluster Registry (OCR) Configuration
- Adjust the Initialization Parameter File for the New Release
- Install and Configure Enterprise Manager Database Control

### Change Passwords for Oracle-Supplied Accounts

Depending on the release from which you upgraded, there may be new Oracle-supplied accounts. Oracle recommends that you lock all Oracle-supplied accounts except for `SYS` and `SYSTEM`, and expire their passwords, thus requiring new passwords to be specified when the accounts are unlocked.

You can view the status of all accounts by issuing the following SQL statement:

```
SQL> SELECT username, account_status
         FROM dba_users
         ORDER BY username;
```

To lock and expire passwords, issue the following SQL statement:

```
SQL> ALTER USER username PASSWORD EXPIRE ACCOUNT LOCK;
```

## Migrate Your Initialization Parameter File to a Server Parameter File

If you are currently using a traditional initialization parameter file, perform the following steps to migrate to a server parameter file:

1. If the initialization parameter file is located on a client machine, transfer the file from the client machine to the server machine.

   **Note:** If you are using Real Application Clusters, then you must combine all of your instance-specific initialization parameter files into a single initialization parameter file. Instructions for doing this, and other actions unique to using a server parameter file for cluster databases, are discussed in:

   - *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide*
   - The Real Application Cluster installation guide for your operating system

2. Create a server parameter file using the `CREATE SPFILE` statement. This statement reads the initialization parameter file to create a server parameter file. The database does not have to be started to issue a `CREATE SPFILE` statement.
3. Start up the instance using the newly-created server parameter file.

**See Also:**

- *Oracle Database Administrator's Guide* for more information about creating server parameter files
- *Oracle Database SQL Reference* for information about the `CREATE SPFILE` statement

## Upgrade Oracle Text

After an upgrade to Oracle Database 10*g*, copy the following files from the previous `ORACLE_HOME` to the new `ORACLE_HOME`:

- Stemming user-dictionary files
- User-modified `KOREAN_MORPH_LEXER` dictionary files
- `USER_FILTER` executables

These files affect all databases installed in the given `ORACLE_HOME`.

**See Also:**

- *Oracle Text Reference* for more information about these files
- *Oracle Text Application Developer's Guide* for information about upgrading your applications from previous releases of Oracle Text

## Upgrade the Oracle Cluster Registry (OCR) Configuration

If you are using Oracle Cluster Services, then you must upgrade the Oracle Cluster Registry (OCR) keys for the database.

Use one of the following options to upgrade the OCR configuration to 10*g*:

- Use `srvconfig` from the 10*g* *ORACLE_HOME*. For example:

```
% srvconfig -upgrade -dbname db_name -orahome pre-10g_Oracle_home
```

- Run `srvctl. For example:`

```
pre-10g_Oracle_home/bin/srvctl remove database -d db_name
10g_Oracle_home/bin/srvctl add database -d db_name -o 10g_Oracle_home
10g_Oracle_home/bin/srvctl add instance -d db_name -i instance -n node
```

## Adjust the Initialization Parameter File for the New Release

Each release of the Oracle Database introduces new initialization parameters, deprecates some initialization parameters, and makes some initialization parameters obsolete. You should adjust the parameter file to account for these changes and to take advantage of new initialization parameters that may be beneficial to your system.

**See Also:**

- *Oracle Database Reference* for a list of the new initialization parameters in release 10.2, and for information about each parameter
- Appendix A, "Initialization Parameter and Data Dictionary Changes" for lists of obsolete and deprecated initialization parameters in release 10.2

The `COMPATIBLE` initialization parameter controls the compatibility level of your database. When you are certain that you no longer need the ability to downgrade your database back to its original version, set the `COMPATIBLE` initialization parameter based on the compatibility level you want for your new database.

**See Also:** "Setting the COMPATIBLE Initialization Parameter" on page 5-3 for information

## Install and Configure Enterprise Manager Database Control

If you want to use Enterprise Manager Database Control with your database, then you must install and configure it. For information on how to do this, see the section on Configuring the Database Control with EMCA in *Oracle Enterprise Manager Advanced Configuration*.

# Tasks to Complete Only After Upgrading a Release 8.1.7 Database

Complete the following additional tasks only if you upgraded your database from release 8.1.7.

- Upgrade User NCHAR Columns
- Migrate Your Server Manager Line Mode Scripts to SQL*Plus

## Upgrade User NCHAR Columns

If you upgraded from a version 8 release and your database contains user tables with `NCHAR` columns, you must upgrade the `NCHAR` columns before they can be used in the Oracle Database.

The following steps convert your NCHAR columns from the old format and character set to the new Oracle Database format. In addition, if your old National Character Set was UTF8, it will remain UTF8 in the Oracle Database. However, your National Character Set will be converted to AL16UTF16 if it was not UTF8 in the old release.

You can override the default upgrade selection of the National Character Set. That is, a version 8 UTF8 National Character Set can be converted to an Oracle Database AL16UTF16 National Character Set or a version 8 non-UTF8 National Character Set can be converted to an Oracle Database UTF8 National Character Set.

You will encounter the following error when attempting to use the NCHAR columns in the Oracle Database until you perform the steps in this section:

```
ORA-12714: invalid national character set specified
```

To upgrade user tables with NCHAR columns, perform the following steps:

1. Log in to the system as the owner of the Oracle home directory.
2. At a system prompt, change to the *ORACLE_HOME*/rdbms/admin directory.
3. Start SQL*Plus.
4. Connect to the database instance as a user with SYSDBA privileges.
5. If the instance is running, shut it down using SHUTDOWN IMMEDIATE:

   ```
   SQL> SHUTDOWN IMMEDIATE
   ```

6. Start up the instance in RESTRICT mode:

   ```
   SQL> STARTUP RESTRICT
   ```

   You may need to use the PFILE option to specify the location of your initialization parameter file.

7. Run utlnchar.sql:

   ```
   SQL> @utlnchar.sql
   ```

   Alternatively, to override the default upgrade selection, run n_switch.sql:

   ```
   SQL> @n_switch.sql
   ```

8. Shut down the instance:

   ```
   SQL> SHUTDOWN IMMEDIATE
   ```

9. Exit SQL*Plus.

## Migrate Your Server Manager Line Mode Scripts to SQL*Plus

The Oracle Database no longer supports the use of Server Manager. If you run SQL scripts using Server Manager line mode, you must modify these scripts so that they are compatible with SQL*Plus. Appendix B, "Migrating from Server Manager to SQL*Plus" contains instructions for modifying your Server Manager line mode scripts to work with SQL*Plus.

# 5

# Compatibility and Interoperability

This chapter describes compatibility and interoperability issues that may arise because of differences between Oracle Database releases. These differences may affect general database administration and existing applications.

This chapter covers the following topics:



## What Is Compatibility?

When you upgrade to a new release of the Oracle Database, certain new features may make your database incompatible with your previous release. Your upgraded Oracle database becomes incompatible with your previous release under the following conditions:

- A new feature stores any data on disk (including data dictionary changes) that cannot be processed with your previous release.
- An existing feature behaves differently in the new environment as compared to the old environment. This type of incompatibility is classified as a **language incompatibility**.

### The COMPATIBLE Initialization Parameter

The Oracle Database enables you to control the compatibility of your database with the `COMPATIBLE` initialization parameter. By default, when the `COMPATIBLE` initialization parameter is not set in your parameter file, it defaults to `10.2.0` for Oracle Database 10*g* release 10.2. You cannot use new Oracle Database 10*g* features that would make your upgraded database incompatible until the `COMPATIBLE` initialization parameter is set to this value.

Table 5–1 illustrates the default value and the range of values of the `COMPATIBLE` initialization parameter in the new Oracle Database 10*g* release and in each release supported for upgrading to the new Oracle Database 10*g* release.

*Table 5–1 The COMPATIBLE Initialization Parameter*

| Oracle Database Release | Default Value | Minimum Value | Maximum Value |
|---|---|---|---|
| Oracle8*i* release 8.1.7 | `8.0.0` | `8.0.0.0.0` | `8.1.7.x.x` |
| Oracle9*i* release 9.0.1 | `8.1.0` | `8.1.0.0.0` | `9.0.1.x.x` |
| Oracle9*i* release 9.2 | `8.1.0` | `8.1.0.0.0` | `9.2.0.x.x` |
| Oracle Database 10*g* release 10.1 | `10.0.0` | `9.2.0.0.0` | `10.1.0.x.x` |
| Oracle Database 10*g* release 10.2 | `10.2.0` | `9.2.0.0.0` | `10.2.0.x.x` |

### Downgrading and Compatibility

Before upgrading to the new Oracle Database 10*g* release, the `COMPATIBLE` initialization parameter must be set to at least `9.2.0`, which is the lowest possible setting for Oracle Database 10*g* release 10.2. Only a subset of Oracle Database 10*g* features are available while the `COMPATIBLE` initialization parameter is set to this value.

After upgrading to the new Oracle Database 10*g* release, you can set the `COMPATIBLE` initialization parameter to match the release number of the new release. Doing so enables you to use all of the features of the new release, but prevents you from downgrading back to your previous release.

If, after upgrading, you will want to downgrade, then the `COMPATIBLE` initialization parameter must be left as follows after the upgrade:

- Set to `9.2.0` if you upgraded from release 9.2
- Set to `10.1.0` or earlier if you upgraded from release 10.1

> **See Also:** Chapter 7, "Downgrading a Database Back to the Previous Oracle Database Release" for more information about downgrading

### How the COMPATIBLE Initialization Parameter Operates

The `COMPATIBLE` initialization parameter operates in the following way:

- It controls the behavior of your database. For example, if you run a release 10.2 database with the `COMPATIBLE` initialization parameter set to `9.2.0`, then the release 10.2 database generates release 9.2 compatible database structures on disk. Therefore, the `COMPATIBLE` initialization parameter enables or disables the use of features. If you try to use any new features that make the database incompatible with the `COMPATIBLE` initialization parameter, then an error is displayed. However, any new features that do not make incompatible changes on disk are enabled.
- It makes sure that the database is compatible with its setting. If the database becomes incompatible with its setting, then the database does not start and terminates with an error. If this happens, then you must set the `COMPATIBLE` initialization parameter to an appropriate value for the database.

> **See Also:** *Oracle Database Concepts* for more information about database structures

### Compatibility Level

The compatibility level of your database corresponds to the value of the `COMPATIBLE` initialization parameter. For example, if you set the `COMPATIBLE` initialization parameter to `10.2.0`, then the database runs at 10.2.0 compatibility level.

### Checking the Current Value of the COMPATIBLE Initialization Parameter

To check the current value of the COMPATIBLE initialization parameter, issue the following SQL statement:

```
SQL> SELECT name, value, description FROM v$parameter
        WHERE name = 'compatible';
```

### When to Set the COMPATIBLE Initialization Parameter

Once the upgrade is complete, you can increase the setting of the COMPATIBLE initialization parameter to the maximum level for the Oracle Database 10*g* release. However, after you do this, the database cannot subsequently be downgraded.

## Setting the COMPATIBLE Initialization Parameter

Complete the following steps to set the COMPATIBLE initialization parameter to a higher value:

1. Perform a backup of your database before you raise the COMPATIBLE initialization parameter (optional).

   Raising the COMPATIBLE initialization parameter may cause your database to become incompatible with earlier releases of the Oracle Database, and a backup ensures that you can return to the earlier release if necessary.

   **See Also:** *Oracle Database Backup and Recovery Basics* for more information about performing a backup

2. If you are using a server parameter file, then complete the following steps:

   a. Update the server parameter file to set or change the value of the COMPATIBLE initialization parameter.

      For example, to set the COMPATIBLE initialization parameter to 10.2.0, issue the following statement:

      ```
      SQL> ALTER SYSTEM SET COMPATIBLE = '10.2.0' SCOPE=SPFILE;
      ```

   b. Shut down and restart the instance.

3. If you are using an initialization parameter file, then complete the following steps:

   a. Shut down the instance if it is running:

      ```
      SQL> SHUTDOWN IMMEDIATE
      ```

   b. Edit the initialization parameter file to set or change the value of the COMPATIBLE initialization parameter.

      For example, to set the COMPATIBLE initialization parameter to 10.2.0, enter the following in the initialization parameter file:

      ```
      COMPATIBLE = 10.2.0
      ```

   c. Start the instance using STARTUP.

# What Is Interoperability?

Interoperability is the ability of different releases of the Oracle Database to communicate and work together in a distributed environment. A distributed database system can have different releases of the Oracle Database, and all supported releases of

the Oracle Database can participate in a distributed database system. However, the applications that work with a distributed database must understand the functionality that is available at each node in the system.

---

**Note:** Since this book documents upgrading and downgrading between different releases of the Oracle Database, this definition of interoperability is appropriate. However, other Oracle Database documentation may use a broader definition of the term **interoperability**; for example, in some cases, interoperability may describe communication between different hardware platforms and operating systems.

---

# Compatibility and Interoperability Issues Introduced in Oracle Database 10*g* Release 10.2

The following sections describe compatibility and interoperability issues introduced in Oracle Database 10*g* Release 2 (10.2). The following sections discuss actions you can take to prevent problems resulting from these issues.

## SQL

The behavior of date formats has changed when used with XML functions. The XML Schema standard specifies that dates and timestamps in XML data be in standard formats. Prior to release 10.2, dates and timestamps in XML data did not follow this standard; rather, the format of dates and timestamps in generated XML was determined by the database format.

As of release 10.2, the XML generation functions in Oracle XML DB produce dates and timestamps according to the XML schema standard.

**See Also:** *Oracle XML DB Developer's Guide* for more information

## CONNECT Role

After upgrading to Oracle Database 10g, the `CONNECT` role will only have the `CREATE SESSION` privilege; the other privileges granted to the `CONNECT` role in earlier releases will be revoked during the upgrade. For further information about this, see "Deprecated CONNECT Role" on page 3-9.

## Timezone Files

The time zone files that are supplied with Oracle Database 10*g* have been updated from version 1 to version 2 to reflect changes in transition rules for some time zone regions. The changes may affect existing data of `TIMESTAMP WITH TIME ZONE` datatype. For further information about this, see "TIMESTAMP WITH TIMEZONE Datatype" on page 3-9.

# Compatibility and Interoperability Issues Introduced in Oracle Database 10*g* Release 10.1

The following sections describe compatibility and interoperability issues introduced in Oracle Database 10*g* release 1 (10.1). If you are upgrading to the new Oracle Database 10*g* release from a release prior to 10.1, see the following sections for information about actions you can take to prevent problems resulting from these issues:

## SQL Optimizer

This section describes compatibility and interoperability issues relating to the SQL Optimizer in Oracle Database 10*g*.

### Rule-Based Optimizer Desupported

Starting with Oracle Database 10*g* release 10.1, the cost-based optimizer (CBO) is now enabled by default. The rule-based optimizer is no longer supported in Oracle Database 10*g*. As a result, `rule` and `choose` are no longer supported as `OPTIMIZER_MODE` initialization parameter values and a warning is displayed in the alert log if `OPTIMIZER_MODE` is set to either of these values.

> **See Also:** *Oracle Database Performance Tuning Guide* for more information about the cost-based optimizer

### Optimizer Statistics

Collection of optimizer statistics is now automatically performed by default for all schemas (including `SYS`), for pre-existing databases upgraded to Oracle Database 10*g*, and for newly created Oracle Database 10*g* databases. Gathering optimizer statistics on stale objects is scheduled by default to occur daily during the maintenance window.

> **See Also:** *Oracle Database Performance Tuning Guide* for more information about optimizer statistics

### COMPUTE STATISTICS Clause of CREATE INDEX

In earlier releases, the `COMPUTE STATISTICS` clause of `CREATE INDEX` could be used to start or stop the collection of statistics on an index. This clause has been deprecated. Oracle Database 10*g* now automatically collects statistics during index creation and rebuild. This clause is supported for backward compatibility and will not cause errors.

### SKIP_UNUSABLE_INDEXES

In earlier releases, SKIP_UNUSABLE_INDEXES was a session parameter only. In Oracle Database 10*g* release 10.1 and later, it is now an initialization parameter and defaults to true. The true setting disables error reporting of indexes and index partitions marked UNUSABLE. This setting allows all operations (inserts, deletes, updates, and selects) on tables with unusable indexes or index partitions.

**See Also:** SKIP_UNUSABLE_INDEXES in *Oracle Database Reference*

## SQL

Starting with Oracle Database 10*g* release 10.1, CLOB <-> NCLOB implicit conversion in SQL and PL/SQL is allowed.

Starting with release 10.1, name resolution for synonyms has changed. If the base object of a synonym does not exist, the SQL compiler now tries looking up PUBLIC.base_object.

Starting with release 10.1, VPD policies are attached to synonyms rather than the base objects.

## Invalid Synonyms After an Upgrade

Starting with Oracle Database 10*g* release 10.1, if a synonym (public or private) is pointing to an object that does not exist or is invalid, then the synonym will be invalid after the upgrade.

## Manageability

Database performance statistics are now automatically collected by the Automatic Workload Repository (AWR) database component for databases upgraded to Oracle Database 10*g* and for newly created Oracle Database 10*g* databases. This data is stored in the SYSAUX tablespace, and is used by the database for automatic generation of performance recommendations.

**See Also:** *Oracle Database Performance Tuning Guide*

If you currently use Statspack for performance data gathering, then refer to the Statspack README (*ORACLE_HOME*/rdbms/admin/spdoc.txt) for directions on using Statspack in Oracle Database 10*g* to avoid conflict with the AWR.

## Transaction and Space

Starting with Oracle Database 10*g* release 10.1, dropped objects are now moved to the recycle bin where the space is only reused when it is needed. This allows an object to be undropped using the FLASHBACK DROP feature.

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

Starting with release 10.1, automatic tuning of undo retention is enabled by default. The UNDO_SUPPRESS_ERRORS initialization parameter has been deprecated. Errors generated when executing rollback segment operations while in automatic undo management mode will always be suppressed.

Starting with release 10.1, the default AUTOEXTEND NEXT size is larger for Oracle-managed files (OMF).

**See Also:** *Oracle Database SQL Reference*

## Recovery and Data Guard

Starting with Oracle Database 10*g* release 10.1, the LOG_ARCHIVE_START initialization parameter has been deprecated. Archiving is now automatically started when the database is placed in ARCHIVELOG mode.

Starting with release 10.1, the LOG_PARALLELISM initialization parameter has been deprecated. Log file parallelism is now automatically enabled.

Starting with release 10.1, the default value for the RECOVERY_PARALLELISM initialization parameter now defaults to allow parallel recovery.

Starting with release 10.1, the default value for the parallel clause in the ALTER DATABASE RECOVER DATABASE statement has changed to PARALLEL.

**See Also:** *Oracle Database SQL Reference*

Starting with release 10.1, the default buffer size for the ASYNC attribute of the LOG_ARCHIVE_DEST_*n* initialization parameter has increased from 2,048 blocks to 61,440 blocks.

Starting with release 10.1, the default values of the parameters MAX_SGA and MAX_SERVERS as set by the DBMS_LOGSTDBY.APPLY_SET() procedure have changed.

**See Also:** *Oracle Database PL/SQL Packages and Types Reference*

Starting with release 10.1, the default values for the Data Guard broker properties ApplyParallel, AsyncBlocks, and LogXptMode have changed.

**See Also:** *Oracle Data Guard Concepts and Administration*

Starting with release 10.1, the default behavior of the STARTUP SQL*Plus command and the ALTER DATABASE MOUNT and ALTER DATABASE OPEN SQL statements have changed for physical standby databases. The commands now automatically detect that the database is a physical standby and thus the STANDBY DATABASE and READ ONLY options are made default.

**See Also:** *Oracle Database SQL Reference*

## RMAN

Starting with Oracle Database 10*g* release 10.1, RMAN now creates an empty file when restoring a file from backup and no backup of the file exists. RMAN backup of archived logs now automatically backs up logs that were created before the last resetlogs. Such logs were previously ignored.

Starting with release 10.1, RMAN now continues to run the remaining portions of a backup or restore job when it encounters an error. RMAN now tries to restore from an alternate backup if it finds the targeted backup is corrupt.

## CREATE DATABASE

In Oracle Database 10*g*, a SYSAUX tablespace is always created at database creation time or whenever a database is upgraded. The SYSAUX tablespace serves as an auxiliary tablespace to the SYSTEM tablespace. Because SYSAUX is the default

tablespace for many Oracle features and products that previously required their own tablespaces, it reduces the number of tablespaces that a DBA must maintain.

> **See Also:** *Oracle Database Administrator's Guide* for more information about the SYSAUX tablespace

Starting with release 10.1, the minimum and default log file sizes have increased. The minimum size is now 4 MB. The default size is 50 MB, unless using Oracle-managed files (OMF) in which case the default is 100 MB.

## Real Application Clusters

In Oracle Database 10*g*, there is now an automated high availability (HA) framework for Real Application Clusters. The framework provides detection, recovery, restart, and notification services.

> **See Also:** *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide* for more information

## Materialized Views

Starting with Oracle Database 10*g* release 10.1, some privilege name changes have been made. The new names appear in all data dictionary views, but both the old and new names are accepted by the GRANT and REVOKE SQL statements.

- CREATE SNAPSHOT changed to CREATE MATERIALIZED VIEW
- CREATE ANY SNAPSHOT changed to CREATE ANY MATERIALIZED VIEW
- ALTER ANY SNAPSHOT changed to ALTER ANY MATERIALIZED VIEW
- DROP ANY SNAPSHOT changed to DROP ANY MATERIALIZED VIEW

## Change Data Capture

Starting with Oracle Database 10*g* release 10.1, the interfaces in DBMS_CDC_SUBSCRIBE and DBMS_CDC_PUBLISH now take a subscription name parameter instead of a subscription handle.

> **See Also:** *Oracle Database PL/SQL Packages and Types Reference*

Starting with release 10.1, subscriber views are now managed automatically. There is no longer any need to call the DBMS_CDC_SUBSCRIBE and DBMS_CDC_PUBLISH interfaces PREPARE_SUBSCRIBER_VIEW() and DROP_SUBSCRIBER_VIEW().

Starting with release 10.1, the computation of synchronous Change Data Capture's RSID$ column has been changed to facilitate joining a subscriber view to itself in order to show both old and new values in the same row. The RSID$ values for the UO and UN rows associated with the same update operation are now the same. To revert to the Oracle9*i* behavior where UN RSID$ value is UO RSID$ value + 1 for the same update operation, set event 10983 to level 4.

## Change in the Default Archival Processing to Remote Archive Destinations

Starting with Oracle Database 10*g* release 10.1, the default archival processing to remote destinations has changed so that archiver processes on the primary database will completely and successfully archive the local online redo log files before

transmitting the redo data to remote standby destinations. This default behavior is equivalent to setting the LOG_ARCHIVE_LOCAL_FIRST initialization parameter to true, which is also new in release 10.1. Note that this new default archival processing is relevant only when log transport services are defined to use archiver processes (ARC*n*), not the log writer process (LGWR), when the archiver processes are writing to remote destinations, and when the remote standby destination is not a mandatory destination.

Prior to release 10.1, the default behavior was to transmit redo data to the standby destination at the same time the online redo log file was being archived to the local online redo log files. You can achieve this behavior by setting the LOG_ARCHIVE_LOCAL_FIRST initialization parameter to false. This archival processing is also relevant only when log transport services are defined to use archiver processes (ARC*n*), not the log writer process (LGWR), when the archiver processes are writing to remote destinations, and when the remote standby destination is not a mandatory destination.

The benefit of the new default behavior is that local archiving, and hence, processing on the primary database, are not affected by archival to non-mandatory, remote destinations. Because local archiving is now disassociated with remote archiving, sites that may have policies to delete archived redo log files on the primary database immediately after backing them up must make sure that the standby destinations have received the corresponding redo data before deleting the archived redo log files on the primary database. You can query the V$ARCHIVED_LOG view to verify that the redo data has been received on standby destinations.

---

**Note:** Any value specified for the LOG_ARCHIVE_LOCAL_FIRST initialization parameter is ignored for mandatory destinations (configured with the MANDATORY attribute of the LOG_ARCHIVE_DEST_*n* initialization parameters).

---

**See Also:** *Oracle Data Guard Concepts and Administration* for complete information about setting up archival to remote destinations

## Limitations on NCHAR Datatypes

In Oracle Database 10*g*, the NCHAR datatypes such as NCHAR, NVARCHAR2, and NCLOB, are limited to the Unicode character set encoding, UTF8 and AL16UTF16.

## PL/SQL Native Compilation

Starting with Oracle Database 10g release 10.1, the configuration of initialization parameters and the command setup for native compilation has been simplified. The only required parameter is PLSQL_NATIVE_LIBRARY_DIR. The parameters related to the compiler, linker, and make utility have been obsoleted. Native compilation is turned on and off by a separate initialization parameter, PLSQL_CODE_TYPE, rather than being one of several options in the PLSQL_COMPILER_FLAGS parameter, which is now deprecated. The $ORACLE_HOME/plsql/spnc_commands file contains the commands and options for compiling and linking, rather than a makefile.

**See Also:**

- *Oracle Database PL/SQL User's Guide and Reference* for further information about compiling PL/SQL code for native execution
- "PL/SQL Native Compilation (NCOMP) In Oracle Database 10g Rel 1" on the Oracle Technology Network (OTN):

  `http://www.oracle.com/technology/tech/pl_sql/htdocs/ncomp_faq.html`

# Compatibility and Interoperability Issues Introduced in Oracle9*i* Release 9.2

The following sections describe compatibility and interoperability issues introduced in Oracle9*i* release 9.2. If you are upgrading to the new Oracle Database 10*g* release from a release earlier than release 9.2, then the sections which follow discuss actions you can take to prevent problems resulting from these issues.



## Locally Managed SYSTEM Tablespace

Starting with Oracle9*i* release 9.2, the `SYSTEM` tablespace can be locally managed. The `SYSTEM` tablespace can be migrated from dictionary managed format to locally managed format using the `DBMS_SPACE_ADMIN.TABLESPACE_MIGRATE_TO_LOCAL` procedure.

Before the `SYSTEM` tablespace can be migrated to locally managed format, you should ensure the following:

- The database has a default temporary tablespace which is not `SYSTEM`
- There are not any rollback segments in dictionary managed tablespaces
- There is at least one online rollback segment in a locally managed tablespace, or an undo tablespace (if using automatic undo management mode) should be online.
- All tablespaces other than the tablespace containing the undo space (undo tablespace or the tablespace containing the rollback segment) and the default temporary tablespace are in read-only mode.
- There is a complete backup of the system.
- The system is in restricted mode.

The following query determines whether the SYSTEM tablespace is locally managed:

```
SQL> SELECT ts# FROM ts$
        WHERE ts# = 0 AND bitmapped <> 0;
```

If 0 rows are returned, then the SYSTEM tablespace is dictionary managed. Otherwise, the SYSTEM tablespace is locally managed.

## New AnyData Datatypes

Starting with Oracle9*i* release 9.2, persistent storage of AnyData values of the following datatypes is allowed:

- TIMESTAMP
- TIMESTAMP WITH TIME ZONE
- TIMESTAMP WITH LOCAL TIME ZONE
- INTERVAL YEAR TO MONTH
- INTERVAL DAY TO SECOND
- NCHAR
- NVARCHAR2
- NCLOB

## Dictionary Managed Tablespaces

Starting with Oracle9*i* release 9.2, dictionary managed tablespaces are deprecated. Once the SYSTEM tablespace has been migrated from dictionary managed format to locally managed format, existing dictionary managed tablespaces are read-only. That is, they cannot be made read-write once the SYSTEM tablespace is locally managed.

Once the SYSTEM tablespace is locally managed (either due to a new installation or SYSTEM tablespace migration), new dictionary managed tablespaces cannot be created.

## Change in Compatibility for Automatic Segment-Space Managed Tablespaces

Starting with Oracle9*i* release 9.0.1.3.0, the compatibility requirement for automatic segment-space managed tablespaces has been changed from 9.0.0 when first introduced in Oracle9*i* release 9.0.1 to 9.0.1.3. If you are upgrading from an Oracle9*i* release earlier than release 9.0.1.3.0 and the database contains any automatic segment-space managed tablespaces, then the COMPATIBLE initialization parameter will need to be set to 9.0.1.3 or higher in order to open the database. The existing tablespaces need not be dropped.

## Compatibility and Object Types

Starting with Oracle9*i* release 9.2, object types support user-defined constructors using the CONSTRUCTOR keyword that cannot be referred to from PL/SQL programs in previous releases of the Oracle Database. Specifically, such programs will fail to compile with an error.

## Oracle Managed Files

Starting with Oracle9*i* release 9.0.1.2.0, the naming scheme used by the Oracle Database to keep track of Oracle Managed Files has changed. As a result, existing

Oracle Managed Files created in Oracle9*i* releases earlier than release 9.0.1.2.0 will appear to the Oracle Database to be regular operating system files.

## Oracle OLAP

The OLAP API client provided with Oracle9*i* release 9.0.1 is not compatible with newer Oracle Database releases; similarly, the OLAP API client provided with Oracle9*i* release 9.2 is not compatible with earlier Oracle Database releases.

The procedure that an application uses to make a connection through the OLAP API has changed in release 9.2. Connections in previous releases relied on CORBA software, but in release 9.2 and later, connections are made through Java Database Connectivity (JDBC). Consequently, programs created using the OLAP API client provided with release 9.0.1 will not execute in later releases, and programs created using the OLAP API client provided with release 9.2 will not execute in earlier Oracle releases.

To upgrade OLAP API applications designed to run in release 9.0.1, application developers must use the OLAP API client provided with release 9.2 and revise the code for making a connection and for creating a MetadataProvider.

For information about using the OLAP API in release 9.2 to perform these actions, see the *Oracle OLAP Developer's Guide to the OLAP API* and the online Oracle OLAP API Reference help.

## Log Format Change with Parallel Redo

Starting with Oracle9*i* release 9.2, the parallel redo feature generates redo logs using a new format. Previous releases of the Oracle Database cannot apply parallel redo generated logs. However, when previous Oracle Database releases detect that release 9.2 parallel redo is being applied, the following error is displayed:

```
ORA-00303: cannot process Parallel Redo
```

Release 9.2 can process Oracle9*i* release 9.0.1 and earlier format logs as well as release 9.2 parallel redo format logs.

## Oracle Dynamic Services

Starting with Oracle9*i* Database release 9.2, Oracle Dynamic Services has been Deprecated. Oracle Dynamic Services, an XML-based broker for the creation, aggregation, and deployment of services from various content sources, was released with Oracle9*i* Database release 9.0.1.

Starting with Oracle9*i*AS release 2 (9.0.2), Oracle is delivering an integrated, J2EE-compliant Web Services platform. Oracle Dynamic Services has been integrated with Oracle9*i*AS Web Services as the XML/HTML Stream Processing Tool.

Oracle9*i*AS release 2 (9.0.2) provides a standards-based, fully integrated J2EE and Web services deployment platform. The current Dynamic Services functionality has been integrated into the Oracle9*i*AS platform, and the Dynamic Services terminal release is being delivered with Oracle9*i* Database release 9.2.

## Oracle Syndication Server

Starting with Oracle9*i* Database release 9.2, Oracle Syndication Server has been Deprecated. Oracle Syndication Server, designed to deliver file system and database content to Information and Content Exchange (ICE)-compliant subscribers, was released with Oracle9*i* Database release 9.0.1.

Starting with Oracle9*i*AS release 2 (9.0.2), Oracle Syndication Server has become a feature of Oracle9*i*AS. The current Syndication Server functionality has been integrated into this platform, and the Syndication Server terminal release is being delivered with Oracle9*i* Database release 9.2.

Oracle9*i*AS Syndication Server is automatically installed with the Oracle9*i*AS Portal install.

# Compatibility and Interoperability Issues Introduced in Oracle9*i* Release 9.0.1

The following sections describe compatibility and interoperability issues introduced in Oracle9*i* release 9.0.1. If you are upgrading to the new Oracle Database 10*g* release from a release earlier than release 9.0.1, then the sections which follow discuss actions you can take to prevent problems resulting from these issues.

## The STARTUP Command

This section describes compatibility and interoperability issues related to the SQL*Plus `STARTUP` command.

### Change in Default Parameter File Selection

When the `STARTUP` command is issued without the `PFILE` option, the Oracle Database attempts to start up the instance using a default parameter file. Starting with Oracle9*i* release 9.0.1, the search criteria for selecting the default parameter file has changed to facilitate the use of a server parameter file.

In previous releases of the Oracle Database, the `STARTUP` command looked for an initialization parameter file with the name *`ORACLE_HOME`*`/dbs/init`*`SID`*`.ora`, where *`SID`* is the instance name.

In release 9.0.1 and later, the process of selecting a default parameter file is as follows:

- The `STARTUP` command first looks for a server parameter file with the name *`ORACLE_HOME`*`/dbs/spfile`*`SID`*`.ora`, where *`SID`* is the instance name.
- The `STARTUP` command next looks for a server parameter file with the name *`ORACLE_HOME`*`/dbs/spfile.ora`.
- If the `STARTUP` command cannot find a server parameter file, it defaults to the behavior of the `STARTUP` command in previous releases, and looks for an initialization parameter file with the name *`ORACLE_HOME`*`/dbs/init`*`SID`*`.ora`.

> **See Also:** *Oracle Database Administrator's Guide* for more information about server parameter files

## Tablespaces and Datafiles

This section describes compatibility and interoperability issues related to tablespaces and datafiles.

### CREATE TABLESPACE: New Behavior

In Oracle8*i*, the default type of tablespace that is created is dictionary managed if the `EXTENT MANAGEMENT` clause is not specified in the `CREATE TABLESPACE` statement.

Starting with Oracle9*i* release 9.0.1, the default for the EXTENT MANAGEMENT clause is locally managed. The default storage clause is parsed to determine whether to use AUTOALLOCATE or UNIFORM allocation policy for this tablespace.

In addition, there was another change made to disallow assigning permanent locally managed tablespaces as a user's temporary tablespace. In Oracle8*i*, an error would be signalled only when a temporary segment had to be created in the tablespace.

### Default Temporary Tablespaces

Oracle strongly recommends using a default temporary tablespace for the database. The default temporary tablespace should be created using the CREATE TEMPORARY TABLESPACE statement.

### Undo Tablespaces

Oracle Database instances can run in one of two undo space management modes:

- Automatic undo management mode
- Manual undo management mode

All instances of the same database must run in the same undo space management mode.

The instance is started in manual undo management mode if the UNDO_MANAGEMENT initialization parameter is not specified.

In the manual undo management mode, CREATE, ALTER, and DROP operations on undo tablespaces are allowed. Rollback segments can coexist with undo tablespaces. That is, rollback segments can exist while running in automatic undo management mode and undo tablespaces can exist while running in manual undo management mode. Undo tablespaces cannot be brought online unless the instance is running in automatic undo management mode.

In automatic undo management mode, DROP ROLLBACK SEGMENT operations are allowed. Rollback segments cannot be brought online.

> **See Also:** *Oracle Database Administrator's Guide* for more information about managing undo space.

## Datatypes

This section describes compatibility and interoperability issues relating to datatypes.

### Datetime and Interval Datatypes

When a database is upgraded to Oracle9*i* release 9.0.1 or later, the database time zone is set to the time zone of the environment variable ORA_SDTZ. If ORA_SDTZ is not set, the database time zone is set to the time zone of the operating system clock. If the time zone of the operating system clock is not set or is not valid, the database time zone defaults to UTC.

old Oracle Database DATE data with time portion can be migrated to either TIMESTAMP to support fractional seconds or TIMESTAMP WITH LOCAL TIME ZONE to support time zone adjustments in addition to fractional seconds without having legacy data rewritten. An ALTER TABLE statement must be explicitly issued to modify a DATE column to a TIMESTAMP column or a TIMESTAMP WITH LOCAL TIME ZONE column.

### Database Character Sets

In Oracle8*i* and earlier releases, the SQL NCHAR datatypes (NCHAR, NVARCHAR2, and NCLOB) will be limited to the Unicode character set encoding (UTF8 and AL16UTF16) only. Any other character sets that were available under the NCHAR datatype, including Asian character sets (such as JA16SJISFIXED), will no longer be supported.

Before migrating your SQL NCHAR data to the new Unicode NCHAR, Oracle recommends that you analyze your SQL NCHAR data, using the Character Set Scanner for the identification of possible invalid character set conversion or data truncation.

> **See Also:** *Oracle Database Globalization Support Guide* for more information about the Character Set Scanner

When you upgrade to Oracle Database 10g, the value of the National Character Set of the upgraded database is set based on the value of the National Character Set of the database being upgraded.

If the old National Character Set is UTF8, then the new National Character Set will be UTF8. Otherwise, the National Character Set is changed to AL16UTF16.

During the upgrade, the existing NCHAR columns in the data dictionary are changed to use the new Oracle Database format and, if the National Character Set has been changed to AL16UTF16, the dictionary NCHAR columns will be converted to the AL16UTF16 character set.

---

**Note:** NCHAR columns in user tables are not changed during the upgrade. To change NCHAR columns in user tables, see "Upgrade User NCHAR Columns" on page 4-8.

---

### AL24UTFFSS Character Set Desupported

The AL24UTFFSS Unicode character set has been desupported in Oracle9*i* release 9.0.1 and later. AL24UTFFSS was introduced in Oracle7 as the Unicode character set supporting the UTF-8 encoding scheme based on the Unicode 1.1 standard, which is now obsolete. In release 9.0.1 and later, The Unicode database character sets AL32UTF8 and UTF8, include the Unicode enhancements based on the Unicode 3.1 standard.

The migration path for existing AL24UTFFSS databases is to upgrade your database character set to UTF8 prior to upgrading your Oracle Database. As with all migrations to a new database character set, Oracle recommends you use the Character Set Scanner for data analysis before attempting to migrate your existing database character set to UTF8.

> **See Also:** *Oracle Database Globalization Support Guide* for more information about the Character Set Scanner

## User-Defined Datatypes

This section describes compatibility and interoperability issues relating to user-defined datatypes.

### Subtypes and Non-Final Types

Types created in Oracle8*i* release 8.1 and earlier are considered to be `FINAL` types. Thus, they cannot be used as supertypes in Oracle9*i* release 9.0.1 and later. However, an `ALTER` statement can be explicitly used to change the type to be `NOT FINAL`.

**Release 8.1 Clients Accessing a Release 9.0.1 or Higher Server** Any transfer involving data of non-final types will return an error. Release 8.1 clients cannot access a release 9.0.1 or higher server if the type has been altered to non-final on the server.

**Release 9.0.1 and Higher Clients Accessing a Release 8.1 Server** Since the release 8.1 server can have only non-final types, no errors occur.

## Oracle Replication

If you plan to use `CHAR` column length semantics in Oracle Database 10g, or if your replication database contains tables with `NCHAR` or `NVARCHAR2` columns, then this section contains considerations for upgrading a replication environment to Oracle Database 10*g*.

### CHAR Column Length Semantics

If you plan to use `CHAR` column length semantics in a replication database after you upgrade it to Oracle Database 10g, then all of the databases participating with that database in the replication environment must also use `CHAR` column length semantics. In this case, Oracle recommends that you upgrade all of the databases participating in the replication environment at the same time. This applies to both master sites and materialized view sites in your replication environment.

If you cannot upgrade all of the databases in your replication environment at the same time, then you can only use `CHAR` column length semantics in your Oracle Database if all of the databases prior to Oracle9*i* are using a single-byte character set. Otherwise, do not switch to `CHAR` column length semantics in the Oracle Database until all of the other databases in the replication environment are upgraded to Oracle Database 10g.

### NCHAR or NVARCHAR2 Columns

If your replication database contains tables with `NCHAR` or `NVARCHAR2` columns, then Oracle recommends that you upgrade all of the databases participating in the replication environment at the same time. This applies to both master sites and materialized view sites in your replication environment. In Oracle Database 10g, all columns specified as `NCHAR` or `NVARCHAR2` datatype are stored in Unicode format.

If you cannot upgrade all of the databases in your replication environment at the same time, then interoperability is only supported if all of the databases prior to Oracle9*i* are using a fixed width national character set. If any of the databases prior to Oracle9*i* are using a variable width character set, then you must convert these databases to fixed width character sets before you upgrade any of the other databases in the replication environment to Oracle Database 10*g*.

**See Also:**

- *Oracle Database Advanced Replication* for more information about replication support for column length semantics and Unicode
- *Oracle Database Globalization Support Guide* for general information about column length semantics and Unicode

# 6

# Upgrading Your Applications

This chapter describes upgrading your current applications and covers the following topics:



## Overview of Upgrading Applications

You do not need to modify existing applications that do not use features available in the new Oracle Database 10*g* release. Existing applications running against a new Oracle Database function the same as they did in prior releases and achieve the same, or enhanced, performance.

Many new features and enhancements are available after upgrading to the new Oracle Database 10*g* release. Some of these features provide added functionality, while others provide improved performance. Before you upgrade your applications, you should review these new features to decide which ones you want to use.

> **See Also:** *Oracle Database New Features* for information about the features available in the new Oracle Database 10*g* release

### Compatibility Issues for Applications

There may be compatibility issues between different releases of the Oracle Database that could affect your applications. These compatibility issues result from differences in the Oracle Database in various releases. Also, in each new release of the Oracle Database, new Oracle reserved words may be added, changes may be made to initialization parameters, and changes may be made to the data dictionary.

When you upgrade your Oracle Database to a new release, make sure that your applications do not use any Oracle reserved words, that your applications are compatible with the initialization parameters of the database, and that your applications are compatible with the data dictionary of the database. Finally, a new release of the Oracle Database software may require certain operating system releases or the application of certain patch sets.

**See Also:**

- Appendix A, "Initialization Parameter and Data Dictionary Changes" for information about initialization parameter changes and data dictionary changes
- *Oracle Database SQL Reference* for a complete list of Oracle reserved words
- Your operating system-specific Oracle documentation for information about operating system requirements

Net8 and Oracle Net Services work with various Oracle Database releases. Thus, Oracle8*i*, Oracle9*i*, and Oracle Database 10*g* can communicate by using Net8 and Oracle Net Services.

# Upgrading Precompiler and OCI Applications

The upgrade path is very similar for precompiler and OCI applications. This section guides you through your upgrade options for these applications and notes differences between precompiler and OCI applications whenever necessary.

Create a test environment before you upgrade your production environment. Your test environment should include your upgraded application and the new Oracle Database. Also, your test environment should provide a realistic test of your application.

**See Also:** *Pro*C/C++ Programmer's Guide*, *Pro*COBOL Programmer's Guide*, and *Oracle Call Interface Programmer's Guide* for more information about using these programming environments.

## Understanding Software Upgrades and Your Client/Server Configuration

To understand your options for upgrading precompiler and OCI applications, you first need to understand the type of software upgrade you are performing and your client/server configuration.

### Types of Software Upgrades

Two types of upgrades are possible for Oracle Database client and server software.

**Major Database Release Upgrade** The upgrade changes the first digit of the release number.For example, upgrading from Oracle9*i* to Oracle Database 10*g* is a major database release upgrade.

**Database Maintenance Release Upgrade** The upgrade changes the second digit of the release number. For example, upgrading from release 9.0.1 to release 9.2 is a database maintenance release upgrade.

---

**Note:** Starting with release 9.2, maintenance releases of the Oracle Database are denoted by a change to the second digit of a release number. In previous releases, the third digit indicated a particular maintenance release.

---

### Possible Client/Server Configurations

Your precompiler and OCI applications run on the client in a client/server environment, where the Oracle Database server is the server. You may use one or more of the following client/server configurations in your environment.

**Different Computers** The client software and the server software are on different computers, and they are connected through a network. The client and server environments are separate.

**Different Oracle Home Directories on the Same Computer** The client software and the server software are on the same computer, but they are installed in different Oracle home directories. Again, the client and server environments are separate.

**Same Oracle Home** The client software and server software are installed in the same Oracle home on the same computer. In this case, any upgrade of the server software is also an upgrade of the client software.

> **See Also:** *Oracle Database Concepts* and *Oracle Database Heterogeneous Connectivity Administrator's Guide* for more information about client/server environments

## Compatibility Rules for Applications When Upgrading Client/Server Software

This section covers compatibility rules that apply when you upgrade Oracle Database client or server software. The rules are based on the type of software upgrade you are performing and on your client/server configuration.

The following sections contain compatibility rules for the following type of upgrades:

- Upgrading the Oracle Database Server Software
- Upgrading the Oracle Database Client Software

> **Note:** This section uses the terms introduced in "Understanding Software Upgrades and Your Client/Server Configuration" on page 6-2.

### Upgrading the Oracle Database Server Software

The following rules apply when you upgrade the Oracle Database server software.

**If You Do Not Change the Client Environment, Then You Do Not Need to Relink.** If your client and server are on different computers or are in different Oracle home directories on the same computer, and you upgrade the Oracle Database server software without changing the client software, then you do not need to precompile, compile, or relink your applications. In these cases, the client software is separate from the server software and will continue to function against the server.

However, if your applications are using the same Oracle home as the Oracle Database server, then your server upgrade also upgrades your client software, and you must follow the rules in "Upgrading the Oracle Database Client Software" on page 6-4.

**Note:** It is possible to upgrade the Oracle Database server software but not install the new precompiler or OCI client software when you are using the same Oracle home for both. In this case, the client software is not upgraded. However, such a configuration is not recommended.

**Applications Can Run Against Newer or Older Oracle Database Server Releases** When you run a precompiler or OCI application against a database server, Oracle recommends that the release of the database server software be equal to or higher than the client software release, but this configuration is not strictly required. For example, if your Oracle Database client software is release 8.1.7, then it is recommended that your Oracle Database server software be release 8.1.7 or higher to run a precompiler application on the client against the server.

### Upgrading the Oracle Database Client Software

Oracle recommends that you upgrade your client software to match the current server software. For example, if you upgrade your Oracle Database server to release 10.2, then Oracle recommends upgrading the client software to release 10.2 as well. Keeping the server and client software at the same release number ensures the maximum stability for your applications. In addition, the latest Oracle Database client software may provide added functionality and performance enhancements that were not available with previous releases.

The following rules apply when you upgrade the Oracle Database client software.

**Applications Can Be Linked with Newer Libraries** The code generated by precompiler applications can be linked with a release of the client library that is equal to or higher than the server release.

OCI applications can be linked with a version of the OCI runtime library that is equal to or higher than the version of the OCI library with which the application was developed.

**Statically-Linked Applications Always Need to be Relinked** For statically-linked applications, when you perform any type of upgrade of the client software, you need to relink only your precompiler and OCI applications.

**Dynamically-Linked Applications Need To Be Relinked When Upgrading to a Major Release** When you upgrade client software to a major release, you need to relink your dynamically-linked precompiler and OCI applications.

## Upgrading Options for Your Precompiler and OCI Applications

You have the following options for upgrading your precompiler and OCI applications:

- **Option 1:** Leave the application and its environment unchanged. Do not relink, precompile, or compile the application, and do not change the application code. The application will continue to work against Oracle Database 10*g*.
- **Option 2:** Precompile and/or compile and then relink the application using the new Oracle Database 10*g* software. Application code needs to be changed if any APIs are deprecated or changed.
- **Option 3:** Change the application code to use new Oracle Database 10*g* features. Then, precompile and/or compile and then relink the code.

These options are listed in order of increasing difficulty and increasing potential benefits. That is, Option 1 is the least difficult option, but it offers the least potential benefits, while Option 3 is the most difficult option, but it offers the most potential benefits. These options are discussed in the following sections.

### Option 1: Leave the Application Unchanged

This option requires that ORACLE_HOME environment of the application is not upgraded. You can leave the application unchanged, and it will continue to work with the new Oracle Database 10g server. The major advantage to this option is that it is simple and easy. In addition, this option requires the least amount of administration, because you do not need to upgrade any of your client computers. If you have a large number of client computers, then avoiding the administrative costs of upgrading all of them can become very important.

The major disadvantage to this option is that your application cannot use the features that are available in the new Oracle Database 10*g* release. In addition, your application cannot leverage some of the possible performance benefits of the Oracle Database 10*g* release.

### Option 2: Precompile or Compile the Application Using the New Software

When upgrading from a major release to Oracle Database 10g software, you have to precompile or compile the application with the new Oracle Database 10g software, after making necessary code changes to account for APIs that are deprecated or changed. Recompiling is not, however, required if you are upgrading to a minor release within Oracle Database 10g software.

This option requires that you install the new Oracle Database client software on each client computer. However, you only need to precompile or compile, and relink your application once, regardless of the number of clients you have. The advantages, however, can be quite large.

By recompiling, you perform a syntax check of your application code. Some problems in the application code that were not detected by previous releases of the Oracle software may emerge when you precompile or compile with the new Oracle software. Therefore, precompiling and compiling with the new software often helps you detect and correct problems in the application code that may have gone unnoticed before.

Also, recompiling affords maximum stability for your application, because you are sure that it works with the new Oracle software. Further, your environment is ready for new development using the latest tools and features available. In addition, you may benefit from performance improvements that are available with the new Oracle software only after you recompile and relink.

### Option 3: Change the Application Code to Use New Oracle Database 10*g* Features

You can make code changes to your application to take advantage of new Oracle Database 10*g* features. This option is the most difficult, but it can provide the most potential benefits. You gain all of the advantages described in Option 2. In addition, you also benefit from changes to your application that may leverage performance and scalability benefits available with Oracle Database 10*g*. Further, you can add new features to your application that are available only with the new Oracle Database 10*g* release.

Become familiar with the new Oracle Database 10*g* features by reading *Oracle Database New Features*. Also, consult the Oracle documentation for your development environment so that you understand how to implement the features you want to use.

For the precompilers, see *Pro*C/C++ Programmer's Guide* and *Pro*COBOL Programmer's Guide*. For OCI, see *Oracle Call Interface Programmer's Guide*.

When you have decided on the new features you want to use, change the code of your application to use these features. Follow the appropriate instructions in the following sections based on your development environment:

- Changing Precompiler Applications
- Changing OCI Applications

**Changing Precompiler Applications** Complete the following steps to change your precompiler application to use Oracle Database 10*g* features:

1. If you want to take advantage of the new Oracle Database 10*g* features, then incorporate code for the new Oracle Database 10*g* functionality into the existing application.
2. Precompile the application using the Oracle precompiler.
3. Compile the application.
4. Relink the application with the Oracle Database 10*g* runtime library, SQLLIB, which is included with the precompiler.

**Changing OCI Applications** Complete the following steps to change your OCI application to use Oracle Database 10*g* features:

1. Incorporate the new Oracle Database 10*g* OCI calls into the existing application.
2. Compile the application.
3. Relink the application with the Oracle Database 10*g* runtime library.

## Upgrading SQL*Plus Scripts and PL/SQL

To use new Oracle Database 10*g* functionality, change existing SQL scripts to use the new Oracle Database 10*g* syntax. Existing SQL scripts run unchanged on Oracle Database 10*g*, and require no modification, if they do not use new Oracle Database 10*g* functionality.

No changes to PL/SQL packages, procedures, or functions should be required. However, improved error checking may now identify errors at compile time rather than at run time.

**See Also:**

- *SQL*Plus User's Guide and Reference* to learn about new functionality in SQL*Plus
- *Oracle Database SQL Reference* for more information about upgrading SQL scripts

## Upgrading Oracle Forms or Oracle Developer Applications

Forms applications run the same on Oracle8*i*, Oracle9*i*, and Oracle Database 10*g*. However, review the new features described in *Oracle Database New Features* to determine whether any of the new Oracle Database 10*g* features would be beneficial to your applications or might otherwise affect them. Information about the ways in which the Oracle Database 10*g* features interact with forms and developer applications is provided in the Oracle Developer documentation set. Also, the Oracle Developer

documentation for your operating system contains instructions for upgrading your forms or developer applications.

---

**Note:** New releases of Oracle Developer may introduce new reserved words that are specific to Oracle Developer. Code changes may be required if your application uses any of these new reserved words.

---

# 7

# Downgrading a Database Back to the Previous Oracle Database Release

This chapter guides you through the process of downgrading a database back to the previous Oracle Database release. This chapter covers the following topics:

- Supported Releases for Downgrading
- Check for Incompatibilities
- Perform a Full Offline Backup
- Downgrade the Database
- Perform Post-Downgrade Tasks

> **See Also:** Some aspects of downgrading are operating system-specific. See your operating system-specific Oracle documentation for additional instructions about downgrading on your operating system.

## Supported Releases for Downgrading

In Oracle Database 10*g* release 10.2, downgrading is supported back to release 10.1 and release 9.2. However, if the release number of your Oracle9*i* database is lower than release 9.2.0.6.0, then you should install the latest patch release for release 9.2; downgrading is not supported to releases prior to release 9.2.0.6.0. Likewise, if the release number of your Oracle10g database is lower than 10.1.0.4, then you should install the latest patch for release 10.1.

If you have Messaging Gateway or Workspace Manager in your database, be aware that neither of them are part of Oracle Database patch sets. Therefore, you will need to separately apply all relevant patches to release 9.2 or 10.1 before downgrading.

---

**Note:** You do not need to first upgrade your previous database to release 9.2.0.6.0 or later, but the release 9.2.0.6.0 or later software must be installed before the downgrade from release 10.2.

Likewise, you do not need to first upgrade your previous database to release 10.1.0.4 or later, but the release 10.1.0.4 or later software must be installed before the downgrade from 10.2.

---

## Check for Incompatibilities

Check the compatibility level of your database to see if your database might have incompatibilities that prevent you from downgrading. If the compatibility level of your release 10.2 database is `10.2.0` or higher, then you will not be able to downgrade. Your compatibility level is determined by the setting of the `COMPATIBLE` initialization parameter. Check your `COMPATIBLE` initialization parameter setting by issuing the following SQL statement:

```
SQL> SELECT name, value, description FROM v$parameter
        WHERE name='compatible';
```

If you are downgrading to release 10.1, the `COMPATIBLE` initialization parameter must be set to `10.1.0` or lower.

If you are downgrading to release 9.2, the `COMPATIBLE` initialization parameter must be set to `9.2.0`.

> **See Also:** "Downgrading and Compatibility" on page 5-2

## Perform a Full Offline Backup

Perform a full offline backup of your release 10.2 database before you downgrade.

> **See Also:** *Oracle Database Backup and Recovery Basics* for more information

## Downgrade the Database

Complete the following steps to downgrade your release 10.2 database to the Oracle Database release from which you originally upgraded:

1. Log in to the system as the owner of the release 10.2 Oracle home directory.
2. Note: This step is required only if the Enterprise Manager Database Control is already configured for the database.

   Stop the Database Control, as follows:

   a. Set the ORACLE_SID environment variable to the *`databaseSid`*

   b. Execute the following command: `ORACLE_HOME/bin/emctl stop dbconsole`

   If the database being downgraded is a Real Application Clusters (RAC) database, this step should be performed for all the instances.
3. At a system prompt, change to the *`ORACLE_HOME`*`/rdbms/admin` directory.

   > **Note:** If you are downgrading a cluster database, shut down the instance completely and change the `CLUSTER_DATABASE` parameter to `false`. After the downgrade, you must set this parameter back to `true`.

4. Start SQL*Plus.
5. Connect to the database instance as a user with `SYSDBA` privileges.
6. Start up the instance in `DOWNGRADE` mode:

   ```
   SQL> STARTUP DOWNGRADE
   ```

You may need to use the `PFILE` option to specify the location of your initialization parameter file.

7. Set the system to spool results to a log file for later verification of success:

```
SQL> SPOOL downgrade.log
```

8. Run `catdwgrd.sql`:

```
SQL> @catdwgrd.sql
```

The following are notes about running the script:

- You must use the version of the script included with release 10.2.
- You must run the script in the release 10.2 environment.
- The script downgrades all Oracle Database components in the database to the major release from which you originally upgraded.

If you encounter any problems when you run the script, or any of the scripts in the remaining steps, then correct the causes of the problems and rerun the script. You can rerun any of the scripts described in this chapter as many times as necessary.

If the downgrade for a component fails, then an `ORA-39709` error will be displayed and the downgrade will not complete. All components must be successfully downgraded before the Oracle Database data dictionary is downgraded.

9. Turn off the spooling of script results to the log file:

```
SQL> SPOOL OFF
```

Then, check the spool file and verify that there were no errors generated during the downgrade. You named the spool file in Step 7; the suggested name was `downgrade.log`. Correct any problems you find in this file and rerun the downgrade script if necessary.

10. Shut down the instance:

```
SQL> SHUTDOWN IMMEDIATE
```

11. Exit SQL*Plus.

12. If your operating system is UNIX, then change the following environment variables to point to the directories of the release to which you are downgrading:

- `ORACLE_HOME`
- `PATH`
- `ORA_NLS33`
- `LD_LIBRARY_PATH`

**See Also:** Your operating system-specific Oracle Database 10*g* installation documents for information about setting other important environment variables on your operating system

13. If your operating system is Windows, then complete the following steps:

a. Stop all Oracle services, including the `OracleService`*`SID`* Oracle service of the release 10.2 database, where *`SID`* is the instance name.

For example, if your *SID* is ORCL, then enter the following at a command prompt:

```
C:\> NET STOP OracleServiceORCL
```

**See Also:** Your *Administrator's Guide* for Windows for information about stopping services

**b.** Delete the Oracle service at a command prompt by issuing the ORADIM command. For example, if your *SID* is ORCL, then enter the following command:

```
C:\> ORADIM -DELETE -SID ORCL
```

**c.** Create the Oracle service of the database to which you are downgrading at a command prompt using the ORADIM command.

```
C:\> ORADIM -NEW -SID SID -INTPWD PASSWORD -MAXUSERS USERS
     -STARTMODE AUTO -PFILE ORACLE_HOME\DATABASE\INITSID.ORA
```

This syntax includes the following variables:

| Variable | Description |
|---|---|
| *SID* | is the same SID name as the SID of the database being downgraded. |
| *PASSWORD* | is the password for the database instance. This is the password for the user connected with SYSDBA privileges. The -INTPWD option is not required. If you do not specify it, then operating system authentication is used, and no password is required. |
| *USERS* | is the maximum number of users who can be granted SYSDBA and SYSOPER privileges. |
| *ORACLE_HOME* | is the Oracle home directory of the database to which you are downgrading. Ensure that you specify the full path name with the -PFILE option, including drive letter of the Oracle home directory. |

For example, if you are downgrading to release 9.2, if your *SID* is ORCL, your *PASSWORD* is TWxy579, the maximum number of *USERS* is 10, and the *ORACLE_HOME* directory is C:\ORANT, then enter the following command:

```
C:\> ORADIM -NEW -SID ORCL -INTPWD TWxy579 -MAXUSERS 10
     -STARTMODE AUTO -PFILE C:\ORANT\DATABASE\INITORCL.ORA
```

14. Restore the configuration files (for example, parameter files, password files, and so on) of the release to which you are downgrading.
15. At a system prompt, change to the *ORACLE_HOME*/rdbms/admin directory of the previous release.
16. Start SQL*Plus.
17. Connect to the database instance as a user with SYSDBA privileges.
18. Start up the instance:

```
SQL> STARTUP MIGRATE
```

19. Set the system to spool results to a log file for later verification of success:

```
SQL> SPOOL reload.log
```

20. Run `catrelod.sql`:

    ```
    SQL> @catrelod.sql
    ```

    The `catrelod.sql` script reloads the appropriate version of all of the database components in the downgraded database.

21. Turn off the spooling of script results to the log file:

    ```
    SQL> SPOOL OFF
    ```

    Then, check the spool file and verify that the packages and procedures compiled successfully. You named the spool file in Step 19; the suggested name was `reload.log`. Correct any problems you find in this file and rerun the appropriate script if necessary.

    `ORA-22308: operation not allowed on evolved type` errors in the spool file may safely be ignored.

22. Shut down and restart the instance for normal operation:

    ```
    SQL> SHUTDOWN IMMEDIATE
    SQL> STARTUP
    ```

    You may need to use the `PFILE` option to specify the location of your initialization parameter file.

23. Do this step if the database is configured for Oracle Label Security. Copy the `olstrig.sql` script from the 10.2 Oracle Home to the version to which the database will be downgraded. Run `olstrig.sql` to re-create DML triggers on tables with Oracle Label Security policies. (See *Oracle Database Enterprise User Administrator's Guide* for more information.)

    ```
    SQL> @olstrig.sql
    ```

24. Run `utlrp.sql`:

    ```
    SQL> @utlrp.sql
    ```

    The `utlrp.sql` script recompiles all existing PL/SQL modules that were previously in an `INVALID` state, such as packages, procedures, types, and so on.

25. Exit SQL*Plus.

Your database is now downgraded.

## Perform Post-Downgrade Tasks

Note: This step is required only if you are downgrading to 10.1 and some form of Enterprise Manager is configured on the host.

Execute the `emca -restore` command with the appropriate options to restore the 10.1 environment. The options that you specify depend on whether the database being downgraded is a Real Application Clusters (RAC) database or an Automatic Storage Management (ASM) database, as follows:

- **Non-RAC, Non-ASM database** - `102Home/bin/emca -restore db`
- **RAC, Non-ASM database** - `102Home/bin/emca -restore db -cluster`
- **Non-RAC, ASM instance** - `102Home/bin/emca -restore asm`
- **RAC ASM instance(s)** - `102Home/bin/emca -restore asm -cluster`

- **Non-RAC, Database and ASM instance** - `102Home/bin/emca -restore db_asm`
- **RAC, Database and ASM instance(s)** - `102Home/bin/emca -restore db_asm -cluster`

When you execute these commands, you will be prompted to enter some or all of the following information, depending on what needs to be restored:

- Oracle Home (`ORACLE_HOME`): Current Oracle Home
- Source Oracle Home (`SRC_OH`): Oracle Home from where the database will be running after downgrade
- Port (`PORT`): Database port after downgrade
- ASM Port (`ASM_PORT`): ASM instance port after downgrade
- SID (`SID`): SID for the non-RAC database
- Database unique name (`DB_UNIQUE_NAME`): Database unique name for the RAC database
- ASM Oracle Home (`ASM_OH`): Oracle Home from where the ASM instance will be running after downgrade
- ASM SID (`ASM_SID`): ASM instance SID for non-RAC ASM instance

# 8

# Data Copying Using Export/Import

This chapter guides you through the process of using the Export and Import utilities to copy data from one Oracle Database to another in which the database release numbers are different. This chapter covers the following topics:

- Export and Import Requirements
- Upgrade the Database Using Export/Import

> **See Also:** *Oracle Database Utilities* for detailed information about the Export and Import utilities

> **Note:** This chapter discusses data copying using the original Export and Import utilities. Data Pump Export and Data Pump Import cannot be used in releases prior to Oracle Database 10*g* release 10.1. Also, `dmp` files from the original Export cannot be read by the Data Pump Import, or vice versa.

## Export and Import Requirements

Dump files created by the Export utility can be imported into all future releases of the Oracle Database. For example, an Oracle8 export dump file can be imported by the oracle8*i*, Oracle9*i*, and Oracle Database 10*g* Import utilities.

Export dump files, however, are *not* downward compatible with the Import utilities of previous Oracle Database releases. That is, exported data *cannot* be imported by the Import utilities of previous Oracle Database releases. For example, an Oracle9*i* export dump file cannot be imported by an Oracle8*i* Import utility, and an Oracle Database 10*g* export dump file cannot be imported by an Oracle9*i* Import utility. The following tables provide specific examples.

Table 8–1 shows which releases to use when exporting data from release 10.2 and then importing that data into earlier releases.

***Table 8–1 Exporting Data From Release 10.2 and Importing Into Earlier Releases***

| Export From | Import To | Use Export Utility For | Use Import Utility For |
|---|---|---|---|
| Release 10.2 | Release 10.2 | Release 10.2 | Release 10.2 |
| Release 10.2 | Release 10.1 | Release 10.1 | Release 10.1 |
| Release 10.2 | Release 9.2 | Release 9.2 | Release 9.2 |

*Table 8–1 (Cont.) Exporting Data From Release 10.2 and Importing Into Earlier Releases*

| Export From | Import To | Use Export Utility For | Use Import Utility For |
|---|---|---|---|
| Release 10.2 | Release 9.0.1 | Release 9.0.1 | Release 9.0.1 |
| Release 10.2 | Release 8.1.7 | Release 8.1.7 | Release 8.1.7 |
| Release 10.2 | Release 8.0.6 | Release 8.0.6 | Release 8.0.6 |

Table 8–2 shows which releases to use when exporting data from releases earlier than 10.2 and then importing that data into release 10.2.

*Table 8–2 Exporting Data From Releases Earlier Than 10.2 and Importing Into Release 10.2*

| Export From | Import To | Use Export Utility For | Use Import Utility For |
|---|---|---|---|
| Release 10.1 | Release 10.2 | Release 10.1 | Release 10.2 |
| Release 9.2 | Release 10.2 | Release 9.2 | Release 10.2 |
| Release 8.1.7 | Release 10.2 | Release 8.1.7 | Release 10.2 |
| Release 8.0.6 | Release 10.2 | Release 8.0.6 | Release 10.2 |
| Release 7.3.4 | Release 10.2 | Release 7.3.4 | Release 10.2 |

## Export/Import Usage on Data Incompatible with a Previous Release

When you export data to a previous release, data that is incompatible with the previous release either is not exported at all or is exported with the loss of some features. For example, the new floating point datatypes in Oracle Database 10*g* will not be exported to an Oracle9*i* database.

In general, if you need to export data to a previous release, then first remove as many incompatibilities with the previous release as possible before you export the data.

# Upgrade the Database Using Export/Import

To upgrade a database using the Export/Import utilities, complete the following steps:

1. Export data from the current database using the Export utility shipped with the current database. See the current database's utilities documents for information about using the Export utility on the current database.

   To ensure a consistent export, make sure the current database is not available for updates during and after the export. If the current database will be available to users for updates after the export, then, prior to making the current database available, put procedures in place to copy the changes made in the current database to the new Oracle Database after the import is complete.

2. Install the new Oracle Database software. Installation is operating system-specific. Installation steps for Oracle Database are covered in your operating system-specific Oracle documentation.

3. If the new Oracle Database will have the same name as the current database, then shut down the current database before creating the new Oracle Database.

4. Create the new Oracle Database.

   **See Also:** *Oracle Database Administrator's Guide* for information about creating an Oracle Database

5. Start SQL*Plus in the new Oracle Database environment.
6. Connect to the database instance as a user with `SYSDBA` privileges.
7. Start an Oracle Database instance using `STARTUP`.
8. Pre-create tablespaces, users, and tables in the new database to improve space usage by changing storage parameters. When you pre-create tables using SQL*Plus, either run the database in the original database compatibility mode or make allowances for the specific data definition conversions that occur during import. You need to specify `IGNORE=Y` on Import when items have been pre-created.

   **Note:** If the new Oracle Database will be created on the same computer as the source database, and you do not want to overwrite the source database data files, then you must pre-create the tablespaces and specify `DESTROY=N` when you import.

9. Use the Import utility of the new Oracle Database to import the objects exported from the current database. Include the `LOG` parameter to save the informational and error messages from the import session to a file.

   **See Also:** *Oracle Database Utilities* for a complete description of the Import utility.

10. After the import, check the import log file for information about which imports of which objects completed successfully and, if there were failures, which failed.

    **See Also:** *Oracle Database Utilities* and the Oracle Database `README.doc` file for error handling information.

11. Use further Import scenarios (see *Oracle Database Utilities*) or SQL scripts that create the database's objects to clean up incomplete imports (or possibly to start an entirely new import).
12. If changes are made to the current database after the export, then make sure those changes are propagated to the new Oracle Database prior to making it available to users. See Step 1 for more information.
13. Complete the procedures described in Chapter 4, "After Upgrading a Database".

# A

# Initialization Parameter and Data Dictionary Changes

This appendix lists changes to initialization parameters and the data dictionary across different releases of the Oracle Database. This appendix also discusses compatibility issues with certain initialization parameters.

This appendix covers the following topics:

- Initialization Parameter Changes
- Compatibility Issues with Initialization Parameters
- Static Data Dictionary View Changes
- Dynamic Performance View Changes

> **Note:** Some of the initialization parameters listed in this appendix are operating system-specific. See your operating system-specific Oracle documentation for more information about these initialization parameters.

## Initialization Parameter Changes

The following sections list changes to initialization parameters across different releases of the Oracle Database:

- Deprecated Initialization Parameters
- Obsolete Initialization Parameters

> **See Also:** The "What's New in Oracle Database Reference" section of *Oracle Database Reference* for a list of new initialization parameters in Oracle Database 10*g*

### Deprecated Initialization Parameters

The following sections list initialization parameters that have been deprecated. To get a list of all deprecated initialization parameters, issue the following SQL statement:

```
SQL> SELECT name FROM v$parameter
         WHERE isdeprecated = 'TRUE';
```

A deprecated parameter behaves the same way as a regular parameter, except that a warning message is displayed at instance startup if a deprecated parameter is

specified in the parameter file. In addition, all deprecated parameters are logged to the alert log at instance startup:

- Initialization Parameters Deprecated in Release 10.2
- Initialization Parameters Deprecated in Release 10.1
- Initialization Parameters Deprecated in Release 9.2
- Initialization Parameters Deprecated in Release 9.0.1

### Initialization Parameters Deprecated in Release 10.2

The following initialization parameters were deprecated in release 10.2:

LOGMNR_MAX_PERSISTENT_SESSIONS
MAX_COMMIT_PROPAGATION_DELAY
REMOTE_ARCHIVE_ENABLE
SERIAL_REUSE
SQL_TRACE

### Initialization Parameters Deprecated in Release 10.1

The following initialization parameters were deprecated in release 10.1:

BUFFER_POOL_KEEP (replaced by DB_KEEP_CACHE_SIZE)
BUFFER_POOL_RECYCLE (replaced by DB_RECYCLE_CACHE_SIZE)
GLOBAL_CONTEXT_POOL_SIZE
LOCK_NAME_SPACE
LOG_ARCHIVE_START
MAX_ENABLED_ROLES
PARALLEL_AUTOMATIC_TUNING
PLSQL_COMPILER_FLAGS (replaced by PLSQL_CODE_TYPE and PLSQL_DEBUG)

### Initialization Parameters Deprecated in Release 9.2

The following initialization parameters were deprecated in release 9.2:

DRS_START (replaced by DG_BROKER_START)

### Initialization Parameters Deprecated in Release 9.0.1

The following initialization parameters were deprecated in release 9.0.1:

FAST_START_IO_TARGET (replaced by FAST_START_MTTR_TARGET)
MTS_CIRCUITS (replaced by CIRCUITS)
MTS_DISPATCHERS (replaced by DISPATCHERS)
MTS_MAX_DISPATCHERS (replaced by MAX_DISPATCHERS)
MTS_MAX_SERVERS (replaced by MAX_SHARED_SERVERS)
MTS_SERVERS (replaced by SHARED_SERVERS)
MTS_SESSIONS (replaced by SHARED_SERVER_SESSIONS)
PARALLEL_SERVER (replaced by CLUSTER_DATABASE)
PARALLEL_SERVER_INSTANCES (replaced by CLUSTER_DATABASE_INSTANCES)

## Obsolete Initialization Parameters

The following sections list initialization parameters that have been made obsolete:

- Initialization Parameters Obsolete in Release 10.2
- Initialization Parameters Obsolete in Release 10.1

> **Note:** An attempt to start a release 10.1 database using one or more of these obsolete initialization parameters will succeed, but a warning will be returned and recorded in the alert log.

### Initialization Parameters Obsolete in Release 10.2

ENQUEUE_RESOURCES

### Initialization Parameters Obsolete in Release 10.1

The following initialization parameters were made obsolete in release 10.1:

DBLINK_ENCRYPT_LOGIN
HASH_JOIN_ENABLED
LOG_PARALLELISM
MAX_ROLLBACK_SEGMENTS
MTS_CIRCUITS
MTS_DISPATCHERS
MTS_LISTENER_ADDRESS
MTS_MAX_DISPATCHERS
MTS_MAX_SERVERS
MTS_MULTIPLE_LISTENERS
MTS_SERVERS
MTS_SERVICE
MTS_SESSIONS
OPTIMIZER_MAX_PERMUTATIONS
ORACLE_TRACE_COLLECTION_NAME
ORACLE_TRACE_COLLECTION_PATH
ORACLE_TRACE_COLLECTION_SIZE
ORACLE_TRACE_ENABLE
ORACLE_TRACE_FACILITY_NAME
ORACLE_TRACE_FACILITY_PATH
PARTITION_VIEW_ENABLED
PLSQL_NATIVE_C_COMPILER
PLSQL_NATIVE_LINKER
PLSQL_NATIVE_MAKE_FILE_NAME
PLSQL_NATIVE_MAKE_UTILITY
ROW_LOCKING
SERIALIZABLE
TRANSACTION_AUDITING
UNDO_SUPPRESS_ERRORS

### Initialization Parameters Obsolete in Release 9.2

The following initialization parameters were made obsolete in release 9.2:

DISTRIBUTED_TRANSACTIONS
MAX_TRANSACTION_BRANCHES
PARALLEL_BROADCAST_ENABLED
STANDBY_PRESERVES_NAMES

### Initialization Parameters Obsolete in Release 9.0.1

The following initialization parameters were made obsolete in release 9.0.1:

```
ALWAYS_ANTI_JOIN
ALWAYS_SEMI_JOIN
DB_BLOCK_LRU_LATCHES
DB_BLOCK_MAX_DIRTY_TARGET
DB_FILE_DIRECT_IO_COUNT
GC_DEFER_TIME
GC_RELEASABLE_LOCKS
GC_ROLLBACK_LOCKS
HASH_MULTIBLOCK_IO_COUNT
INSTANCE_NODESET
JOB_QUEUE_INTERVAL
OPS_INTERCONNECTS
OPTIMIZER_PERCENT_PARALLEL
SORT_MULTIBLOCK_READ_COUNT
TEXT_ENABLE
```

# Compatibility Issues with Initialization Parameters

The lists of deprecated and obsolete initialization parameters earlier in this appendix show changes to initialization parameters across different releases of the Oracle Database. However, certain initialization parameter changes require special attention because they may raise compatibility issues for your database. These parameter changes are described in this section.

## Change in Behavior for SESSION_CACHED_CURSORS

In previous Oracle Database releases, the number of SQL cursors cached by PL/SQL was determined by the `OPEN_CURSORS` initialization parameter. Starting with Oracle Database 10*g* release 10.1, the number of cached cursors is determined by the `SESSION_CACHED_CURSORS` initialization parameter.

**See Also:** `SESSION_CACHED_CURSORS` in *Oracle Database Reference*

## New default value for DB_BLOCK_SIZE

In Oracle Database 10*g*, the default value of `DB_BLOCK_SIZE` is operating system specific, but is typically 8 KB (8192 bytes). In previous Oracle Database releases, the default value was 2 KB (2048 bytes). If `DB_BLOCK_SIZE` is not specified in the parameter file when upgrading to the new Oracle Database 10*g* release, then you will receive an error when attempting to start up your Oracle Database. Add the following to your parameter file:

```
DB_BLOCK_SIZE = 2048
```

If `DB_BLOCK_SIZE` is specified in the parameter file, then the Oracle Database will use this value instead of the default value of 8 KB.

## OPTIMIZER_MAX_PERMUTATIONS and OPTIMIZER_FEATURES_ENABLE

Starting with Oracle Database 10*g*, the `OPTIMIZER_MAX_PERMUTATIONS` initialization parameter has been made obsolete. If you are upgrading from Oracle9*i* and have `OPTIMIZER_FEATURES_ENABLE` set to `8.1.7` or lower and `OPTIMIZER_MAX_PERMUTATIONS` explicitly set to `2000` in the parameter file, then the release 8.1.7 default of `80000` will be used when you start up the release 10.1 database.

Setting `OPTIMIZER_FEATURES_ENABLE` to `9.0.0` or higher will set the default to `2000`.

## Change in Behavior for LOG_ARCHIVE_FORMAT

Starting with Oracle Database 10*g* release 10.1, if the COMPATIBLE initialization parameter is set to 10.0.0 or higher, then archive log file names must contain each of the elements %s (sequence), %t (thread), and %r (resetlogs ID) to ensure that all archive log file names are unique. If the LOG_ARCHIVE_FORMAT initialization parameter is set in the parameter file, then make sure the parameter value contains the %s, %t, and %r elements.

## New Default Value for PGA_AGGREGATE_TARGET

Starting with Oracle Database 10*g* release 10.1, Automatic PGA Memory Management is now enabled by default (unless PGA_AGGREGATE_TARGET is explicitly set to 0 or WORKAREA_SIZE_POLICY is explicitly set to MANUAL). PGA_AGGREGATE_TARGET defaults to 20% of the size of the SGA, unless explicitly set. Oracle recommends tuning the value of PGA_AGGREGATE_TARGET after upgrading.

**See Also:** *Oracle Database Performance Tuning Guide*

## Change in Behavior for SHARED_POOL_SIZE

In previous releases, the amount of shared pool memory that was allocated was equal to the value of the SHARED_POOL_SIZE initialization parameter plus the amount of internal SGA overhead computed during instance startup. Starting with Oracle Database 10*g* release 10.1, the value of SHARED_POOL_SIZE must now also accommodate this shared pool overhead.

## Shared Server Parameters

Starting with Oracle Database 10*g* release 10.1, the recommended way to turn on shared server mode is to set SHARED_SERVERS to a value greater than 0. This can be done at startup or dynamically after the instance is started. If shared server mode is turned off by setting SHARED_SERVERS to 0, then this only affects new clients (that is, no new clients can connect in shared mode; clients that are already connected in shared mode continue to be serviced by shared servers).

In previous releases, the recommended way to turn on shared server mode was to set DISPATCHERS. If SHARED_SERVERS was changed to 0 and shared server clients were still connected, client requests would hang.

Prior to release 10.1, the following shared server parameters could not be changed dynamically:

- MAX_SHARED_SERVERS
- MAX_DISPATCHERS
- SHARED_SERVER_SESSIONS
- CIRCUITS

Starting with release 10.1, these shared server parameters are dynamically modifiable.

### New Default Value for DISPATCHERS

Starting with release 10.1, the default for DISPATCHERS is '(PROTOCOL=TCP)'. DISPATCHERS is given this default value if it is not set or if it is set to '' and SHARED_SERVERS is set to 1 or higher.

In previous releases, there was no default value for DISPATCHERS.

### New Default Value for SHARED_SERVERS

Starting with release 10.1, if DISPATCHERS is set such that the total number of dispatchers is equal to 0, then SHARED_SERVERS defaults to 0. If DISPATCHERS is set such that the total number of dispatchers is greater than 0, then SHARED_SERVERS defaults to 1 as in previous releases.

In previous releases, if DISPATCHERS was set such that the number of dispatchers is equal to 0, then SHARED_SERVERS defaulted to 1.

### New Default Value for MAX_SHARED_SERVERS

Starting with release 10.1, there is no preset default for MAX_SHARED_SERVERS. The maximum number of shared servers varies depending on the number of free process slots. If MAX_SHARED_SERVERS is not set or is set to a value greater than or equal to PROCESSES, then PMON will not spawn any more shared servers if the number of free process slots is either 2 (if PROCESSES is less than 24) or is less than 1 / 8, unless the existing servers are involved in a deadlock situation. If the existing servers are involved in a deadlock situation, then no matter the transaction load, a new server will be spawned if there is a free process slot.

In previous releases, the default for MAX_SHARED_SERVERS is 20, or 2 * SHARED_SERVERS, whichever is greater, subject to the condition that MAX_SHARED_SERVERS does not exceed PROCESSES.

Starting with release 10.1, SHARED_SERVERS can be set higher than MAX_SHARED_SERVERS, in which case the number of servers will remain constant at the level set for SHARED_SERVERS. This is to allow the range SHARED_SERVERS - MAX_SHARED_SERVERS to be changed without having to change these parameters in a specific order.

In previous releases, SHARED_SERVERS cannot be set higher than MAX_SHARED_SERVERS.

### New Default Value for SHARED_SERVER_SESSIONS

Starting with release 10.1, there is no preset default for SHARED_SERVER_SESSIONS. That is, if SHARED_SERVER_SESSIONS is not specified, then shared server sessions can be created as needed and as permitted by the session limit.

In previous releases, the default for SHARED_SERVER_SESSIONS was the maximum number of virtual circuits (CIRCUITS), or the maximum number of database sessions (SESSIONS) - 5, whichever is smaller.

### New Default Value for CIRCUITS

Starting with release 10.1, there is no preset default for CIRCUITS. That is, if CIRCUITS is not specified, then circuits can be created as needed and as permitted by dispatcher constraints and system resources.

In previous releases, the default for CIRCUITS was the maximum number of database sessions (SESSIONS) if shared server mode was enabled, 0 otherwise.

### New Default Value for MAX_DISPATCHERS

Starting with release 10.1, there is no preset default for MAX_DISPATCHERS. MAX_DISPATCHERS no longer limits the number of dispatchers; the user can increase the number of dispatchers via the DISPATCHERS parameter as long as there are free process slots and system resources.

In previous releases, the default for MAX_DISPATCHERS was 5, or the total number of dispatchers specified via the DISPATCHERS parameter, whichever was greater.

### New Default Value for DB_BLOCK_CHECKSUM

Starting with Oracle9*i* release 9.0.1, the `DB_BLOCK_CHECKSUM` initialization parameter has a new default value. In previous releases, the default value was `false`, but in Oracle9*i* release 9.0.1 and higher, the default value is `true`.

**See Also:** `DB_BLOCK_CHECKSUM` in *Oracle Database Reference*

### Maximum Number of Job Queue Processes

In Oracle9*i*, the maximum number of job queue processes that can be spawned per instance is 1000. In previous releases, the maximum number was 36. The `JOB_QUEUE_PROCESSES` initialization parameter controls the number of job queue processes.

**See Also:** `JOB_QUEUE_PROCESSES` in *Oracle Database Reference*

### New Default Value for LOG_CHECKPOINT_TIMEOUT

Starting with Oracle8*i* release 8.1.5, the `LOG_CHECKPOINT_TIMEOUT` initialization parameter has a new default value. In previous releases, the default value was zero seconds, but in Oracle8*i* release 8.1.5 and higher, the default value is 1800 seconds.

**See Also:** `LOG_CHECKPOINT_TIMEOUT` in *Oracle Database Reference*

### The O7_DICTIONARY_ACCESSIBILITY Parameter

The `O7_DICTIONARY_ACCESSIBILITY` initialization parameter controls whether to continue Oracle7 data dictionary behavior. Use of this initialization parameter is only a temporary expedient. Starting with Oracle9*i* release 9.0.1, the default value of this initialization parameter is `false`.

## Static Data Dictionary View Changes

The following sections list changes to static data dictionary views across different releases of the Oracle Database:

- Deprecated Static Data Dictionary Views
- Obsolete Static Data Dictionary Views
- Static Data Dictionary Views with Renamed Columns
- Static Data Dictionary Views with Dropped Columns

**See Also:** The "What's New in Oracle Database Reference" section of *Oracle Database Reference* for a list of new static data dictionary views in Oracle Database 10*g*

### Deprecated Static Data Dictionary Views

The following sections list static data dictionary views that have been deprecated:

- Static Data Dictionary Views Deprecated in Release 10.1
- Static Data Dictionary Views Deprecated in Release 9.2
- Static Data Dictionary Views Deprecated in Release 9.0.1

#### Static Data Dictionary Views Deprecated in Release 10.1

The following static data dictionary views were deprecated in release 10.1:

ALL_STORED_SETTINGS (replaced by ALL_PLSQL_OBJECT_SETTINGS)
DBA_STORED_SETTINGS (replaced by DBA_PLSQL_OBJECT_SETTINGS)
USER_STORED_SETTINGS (replaced by USER_PLSQL_OBJECT_SETTINGS)

### Static Data Dictionary Views Deprecated in Release 9.2

The following static data dictionary views were deprecated in release 9.2:

ALL_RULESETS (replaced by ALL_RULE_SETS)
DBA_RULESETS (replaced by DBA_RULE_SETS)
USER_RULESETS (replaced by USER_RULE_SETS)

### Static Data Dictionary Views Deprecated in Release 9.0.1

The following static data dictionary views were deprecated in release 9.0.1:

ALL_REGISTERED_SNAPSHOTS (replaced by ALL_REGISTERED_MVIEWS)
ALL_SNAPSHOT_LOGS (replaced by ALL_BASE_TABLE_MVIEWS and ALL_MVIEW_LOGS)
ALL_SNAPSHOT_REFRESH_TIMES (replaced by ALL_MVIEW_REFRESH_TIMES)
DBA_REGISTERED_SNAPSHOT_GROUPS (replaced by DBA_REGISTERED_MVIEW_GROUPS)
DBA_REGISTERED_SNAPSHOTS (replaced by DBA_REGISTERED_MVIEWS)
DBA_SNAPSHOT_LOG_FILTER_COLS (replaced by DBA_MVIEW_LOG_FILTER_COLS)
DBA_SNAPSHOT_LOGS (replaced by DBA_BASE_TABLE_MVIEWS and DBA_MVIEW_LOGS)
DBA_SNAPSHOT_REFRESH_TIMES (replaced by DBA_MVIEW_REFRESH_TIMES)
USER_REGISTERED_SNAPSHOTS (replaced by USER_REGISTERED_MVIEWS)
USER_SNAPSHOT_LOGS (replaced by USER_BASE_TABLE_MVIEWS and USER_MVIEW_LOGS)
USER_SNAPSHOT_REFRESH_TIMES (replaced by USER_MVIEW_REFRESH_TIMES)

## Obsolete Static Data Dictionary Views

The following sections list static data dictionary views that have been made obsolete:

- Static Data Dictionary Views Obsolete in Release 10.1

### Static Data Dictionary Views Obsolete in Release 10.1

The following static data dictionary views were made obsolete in release 10.1:

| ALL_ View | DBA_ View | USER_ View |
|---|---|---|
| ALL_SOURCE_TAB_COLUMNS | DBA_SOURCE_TAB_COLUMNS | USER_SOURCE_TAB_COLUMNS |

## Static Data Dictionary Views with Renamed Columns

The following sections list static data dictionary views with renamed columns:

- Static Data Dictionary Views with Renamed Columns in Release 9.0.1

### Static Data Dictionary Views with Renamed Columns in Release 9.0.1

The static data dictionary view columns listed in Table A–1 were renamed in release 9.0.1:

*Table A–1 Static Data Dictionary Views with Renamed Columns in Release 9.0.1*

| Static Data Dictionary View | Pre-Release 9.0.1 Column Name | Release 9.0.1 and Higher Column Name |
|---|---|---|
| DBA_RSRC_PLAN_DIRECTIVES | MAX_ACTIVE_SESS_TARGET_P1 | ACTIVE_SESS_POOL_P1 |
| DBA_RSRC_PLANS | MAX_ACTIVE_SESS_TARGET_MTH | ACTIVE_SESS_POOL_MTH |

## Static Data Dictionary Views with Dropped Columns

The following sections list static data dictionary views with dropped columns:

- Static Data Dictionary Views with Dropped Columns in Release 10.2
- Static Data Dictionary Views with Dropped Columns in Release 9.0.1

### Static Data Dictionary Views with Dropped Columns in Release 10.2

The following static data dictionary view columns were dropped in release 10.2:

| Static Data Dictionary View | Dropped Columns |
|---|---|
| DBA_HIST_SQLBIND | CHILD_NUMBER |

### Static Data Dictionary Views with Dropped Columns in Release 9.0.1

The following static data dictionary view columns were dropped in release 9.0.1:

| Static Data Dictionary View | Dropped Columns |
|---|---|
| DBA_RSRC_PLAN_DIRECTIVES | MAX_ACTIVE_SESS_TARGET_P1 |
| DBA_RSRC_PLANS | MAX_ACTIVE_SESS_TARGET_MTH |

# Dynamic Performance View Changes

The following sections list changes to dynamic performance views (V$ views) across different releases of the Oracle Database:

- Deprecated Dynamic Performance Views
- Obsolete Dynamic Performance Views
- Dynamic Performance Views with Renamed Columns
- Dynamic Performance Views with Dropped Columns

> **See Also:** The "What's New in Oracle Database Reference" section of *Oracle Database Reference* for a list of new dynamic performance views in Oracle Database 10*g*

## Deprecated Dynamic Performance Views

The following sections list dynamic performance views that have been deprecated:

- Dynamic Performance Views Deprecated in Release 10.1
- Dynamic Performance Views Deprecated in Release 9.2
- Dynamic Performance Views Deprecated in Release 9.0.1

### Dynamic Performance Views Deprecated in Release 10.1

The following dynamic performance views were deprecated in release 10.1:

GV$CACHE
GV$CACHE_TRANSFER
GV$CLASS_CACHE_TRANSFER (replaced by GV$INSTANCE_CACHE_TRANSFER)
GV$FALSE_PING
GV$FILE_CACHE_TRANSFER (replaced by GV$INSTANCE_CACHE_TRANSFER)
GV$GC_ELEMENTS_WITH_COLLISIONS
GV$LOCK_ACTIVITY
GV$TEMP_CACHE_TRANSFER (replaced by GV$INSTANCE_CACHE_TRANSFER)
V$CACHE
V$CACHE_LOCK
V$CACHE_TRANSFER
V$CLASS_CACHE_TRANSFER (replaced by V$INSTANCE_CACHE_TRANSFER)
V$FALSE_PING
V$FILE_CACHE_TRANSFER (replaced by V$INSTANCE_CACHE_TRANSFER)
V$GC_ELEMENTS_WITH_COLLISIONS
V$LOCK_ACTIVITY
V$TEMP_CACHE_TRANSFER (replaced by V$INSTANCE_CACHE_TRANSFER)

### Dynamic Performance Views Deprecated in Release 9.2

The following dynamic performance views were deprecated in release 9.2:

GV$SORT_USAGE (replaced by GV$TEMPSEG_USAGE)
V$SORT_USAGE (replaced by V$TEMPSEG_USAGE)

### Dynamic Performance Views Deprecated in Release 9.0.1

The following dynamic performance views were deprecated in release 9.0.1:

GV$BSP (replaced by GV$CR_BLOCK_SERVER)
GV$CLASS_PING (replaced by GV$CLASS_CACHE_TRANSFER)
GV$DLM_ALL_LOCKS (replaced by GV$GES_ENQUEUE)
GV$DLM_CONVERT_LOCAL (replaced by GV$GES_CONVERT_LOCAL)
GV$DLM_CONVERT_REMOTE (replaced by GV$GES_CONVERT_REMOTE)
GV$DLM_LATCH (replaced by GV$GES_LATCH)
GV$DLM_LOCKS (replaced by GV$GES_BLOCKING_ENQUEUE)
GV$DLM_MISC (replaced by GV$GES_STATISTICS)
GV$DLM_RESS (replaced by GV$GES_RESOURCE)
GV$DLM_TRAFFIC_CONTROLLER (replaced by GV$GES_TRAFFIC_CONTROLLER)
GV$FILE_PING (replaced by GV$FILE_CACHE_TRANSFER)
GV$LOCK_ELEMENT (replaced by GV$GC_ELEMENT)
GV$LOCKS_WITH_COLLISIONS (replaced by GV$GC_ELEMENTS_WITH_COLLISIONS)
GV$MAX_ACTIVE_SESS_TARGET_MTH (replaced by GV$ACTIVE_SESS_POOL_MTH)
GV$MTS (replaced by GV$SHARED_SERVER_MONITOR)
GV$PING (replaced by GV$CACHE_TRANSFER)
GV$TEMP_PING (replaced by GV$TEMP_CACHE_TRANSFER)
V$BSP (replaced by V$CR_BLOCK_SERVER)
V$CLASS_PING (replaced by V$CLASS_CACHE_TRANSFER)
V$DLM_ALL_LOCKS (replaced by V$GES_ENQUEUE)
V$DLM_CONVERT_LOCAL (replaced by V$GES_CONVERT_LOCAL)
V$DLM_CONVERT_REMOTE (replaced by V$GES_CONVERT_REMOTE)

V$DLM_LATCH (replaced by V$GES_LATCH)
V$DLM_LOCKS (replaced by V$GES_BLOCKING_ENQUEUE)
V$DLM_MISC (replaced by V$GES_STATISTICS)
V$DLM_RESS (replaced by V$GES_RESOURCE)
V$DLM_TRAFFIC_CONTROLLER (replaced by V$GES_TRAFFIC_CONTROLLER)
V$FILE_PING (replaced by V$FILE_CACHE_TRANSFER)
V$LOCK_ELEMENT (replaced by V$GC_ELEMENT)
V$LOCKS_WITH_COLLISIONS (replaced by V$GC_ELEMENTS_WITH_COLLISIONS)
V$MAX_ACTIVE_SESS_TARGET_MTH (replaced by V$ACTIVE_SESS_POOL_MTH)
V$MTS (replaced by V$SHARED_SERVER_MONITOR)
V$PING (replaced by V$CACHE_TRANSFER)
V$TEMP_PING (replaced by V$TEMP_CACHE_TRANSFER)

## Obsolete Dynamic Performance Views

The following sections list dynamic performance views that have been made obsolete:

- Dynamic Performance Views Obsolete in Release 10.1
- Dynamic Performance Views Obsolete in Release 9.2
- Dynamic Performance Views Obsolete in Release 9.0.1

### Dynamic Performance Views Obsolete in Release 10.1

The following dynamic performance views were made obsolete in release 10.1:

| GV$ Views | V$ Views |
|---|---|
| GV$COMPATIBILITY | V$COMPATIBILITY |
| GV$COMPATSEG | V$COMPATSEG |
| GV$MLS_PARAMETERS | V$MLS_PARAMETERS |
| GV$MTS | V$MTS |

### Dynamic Performance Views Obsolete in Release 9.2

The following dynamic performance views were made obsolete in release 9.2:

| GV$ Views | V$ Views |
|---|---|
| GV$LOADCSTAT | V$LOADCSTAT |
| GV$LOADTSTAT | V$LOADTSTAT |

### Dynamic Performance Views Obsolete in Release 9.0.1

The following dynamic performance views were made obsolete in release 9.0.1:

| GV$ Views | V$ Views |
|---|---|
| GV$TARGETRBA | V$TARGETRBA |

## Dynamic Performance Views with Renamed Columns

The following sections list dynamic performance views with renamed columns:

- Dynamic Performance Views with Renamed Columns in Release 9.2

- Dynamic Performance Views with Renamed Columns in Release 9.0.1

### Dynamic Performance Views with Renamed Columns in Release 9.2

The dynamic performance view columns listed in Table A–2 were renamed in release 9.2:

*Table A–2 Dynamic Performance Views with Renamed Columns in Release 9.2*

| Dynamic Performance View | Pre-Release 9.2 Column Name | Release 9.2 and Higher Column Name |
|---|---|---|
| GV$ARCHIVE_DEST and V$ARCHIVE_DEST | MANIFEST | REGISTER |
| GV$ARCHIVE_DEST and V$ARCHIVE_DEST | REGISTER | REMOTE_TEMPLATE |
| GV$DATABASE and V$DATABASE | STANDBY_MODE | PROTECTION_MODE |
| GV$LOGMNR_CALLBACK and V$LOGMNR_CALLBACK | CALLBACK_STATE | STATE |
| GV$LOGMNR_CALLBACK and V$LOGMNR_CALLBACK | CALLBACK_TYPE | TYPE |
| GV$LOGMNR_CALLBACK and V$LOGMNR_CALLBACK | CALLBACK_CAPABILITY | CAPABILITY |
| GV$LOGMNR_REGION and V$LOGMNR_REGION | ID | MEMSTATE |
| GV$LOGMNR_REGION and V$LOGMNR_REGION | CURRENT_STATE | STATE |

### Dynamic Performance Views with Renamed Columns in Release 9.0.1

The dynamic performance view columns listed in Table A–3 were renamed in release 9.0.1:

*Table A–3 Dynamic Performance Views with Renamed Columns in Release 9.0.1*

| Dynamic Performance View | Pre-Release 9.0.1 Column Name | Release 9.0.1 and Higher Column Name |
|---|---|---|
| GV$RSRC_CONSUMER_GROUP and V$RSRC_CONSUMER_GROUP | SESSIONS_QUEUED | QUEUE_LENGTH |

## Dynamic Performance Views with Dropped Columns

The following sections list dynamic performance views with dropped columns. If an application requires one or more of these columns, then modify the application accordingly:

- Dynamic Performance Views with Dropped Columns in Release 9.2
- Dynamic Performance Views with Dropped Columns in Release 9.0.1

### Dynamic Performance Views with Dropped Columns in Release 9.2

The following dynamic performance view columns were dropped in release 9.2:

| Dynamic Performance View | Dropped Columns |
|---|---|
| GV$DATABASE and V$DATABASE | STANDBY_MODE |

| Dynamic Performance View | Dropped Columns |
|---|---|
| GV$LOGMNR_CALLBACK and V$LOGMNR_CALLBACK | FUNC_NAME |
| | CALLBACK_ID |
| | CALLBACK_RESULT_SIZE |
| | CALLBACK_STATE |
| | CALLBACK_TYPE |
| | CALLBACK_CAPABILITY |
| | NUMBER_INVOKED |
| GV$LOGMNR_REGION and V$LOGMNR_REGION | ID |
| | CURRENT_STATE |

### Dynamic Performance Views with Dropped Columns in Release 9.0.1

The following dynamic performance view columns were dropped in release 9.0.1:

| Dynamic Performance View | Dropped Columns |
|---|---|
| GV$LOGMNR_CONTENTS and V$LOGMNR_CONTENTS | PH1_NAME |
| | PH1_REDO |
| | PH1_UNDO |
| | PH2_NAME |
| | PH2_REDO |
| | PH2_UNDO |
| | PH3_NAME |
| | PH3_REDO |
| | PH3_UNDO |
| | PH4_NAME |
| | PH4_REDO |
| | PH4_UNDO |
| | PH5_NAME |
| | PH5_REDO |
| | PH5_UNDO |
| GV$RSRC_CONSUMER_GROUP and V$RSRC_CONSUMER_GROUP | SESSIONS_QUEUED |

# B

# Migrating from Server Manager to SQL*Plus

This appendix guides you through the process of modifying your Server Manager line mode scripts to work with SQL*Plus. Server Manager is not supported in Oracle9i release 9.0.1 and later. If you run SQL scripts using Server Manager line mode, then you will need to change these scripts so that they are compatible with SQL*Plus, and then run them using SQL*Plus.

This appendix covers the following topics:

- Startup Differences
- Commands
- Syntax Differences

> **See Also:** *SQL*Plus User's Guide and Reference* for detailed information about SQL*Plus

> **Note:** For brevity, Server Manager line mode is referred to as Server Manager in the rest of this appendix.

## Startup Differences

The methods for starting Server Manager and SQL*Plus are different, and your SQL scripts must be modified to properly start SQL*Plus. The following sections explain the startup differences and provide options for starting SQL*Plus.

### Starting Server Manager

To start Server Manager, enter the name of the Server Manager program at a system prompt; the name of this program is operating system-specific. After you start up Server Manager, connect using the `CONNECT` command, as in the following example:

```
CONNECT hr/hr
```

### Starting SQL*Plus

The following sections describe various ways to start SQL*Plus.

#### Starting SQL*Plus with the NOLOG Option

If you want SQL*Plus to behave in the same way as Server Manager, then use the `NOLOG` option when you start SQL*Plus, as in the following example:

```
sqlplus /nolog
```

SQL*Plus starts and you can use the CONNECT command to connect as a user.

### Starting SQL*Plus with Connect Information

Another option for starting SQL*Plus is to enter the connect information when you start the program. For example, to start SQL*Plus and connect as user hr with password hr, enter the following:

```
sqlplus hr/hr
```

SQL*Plus starts and connects as user hr.

### Starting SQL*Plus without Options or Connect Information

To start SQL*Plus without options or connect information, enter the following:

```
sqlplus
```

SQL*Plus prompts you for a user name and password. When you enter a valid user name and password, SQL*Plus starts and connects as the user you specified at the prompts. In your SQL scripts, however, you may not want to prompt the user to enter a user name and password.

# Commands

Server Manager and SQL*Plus share certain commands that behave the same in both programs. Other commands, however, behave differently in SQL*Plus than they do in Server Manager. To successfully migrate from Server Manager to SQL*Plus, you need to understand these differences and similarities. The following sections include information about modifying your SQL scripts to use commands that are interpreted correctly by SQL*Plus.

## Commands Introduced in SQL*Plus Release 8.1

Table B–1 lists Server Manager commands that are available in SQL*Plus release 8.1 and higher. You can use these commands in SQL scripts that you run with SQL*Plus.

---

**Note:** If you run SQL scripts containing any of these commands in Oracle7 or release 8.0, then you must use Server Manager to run these scripts. Versions of SQL*Plus before SQL*Plus release 8.1 will not run scripts containing these commands.

---

***Table B–1 Commands Introduced in SQL*Plus Release 8.1***

| Command | Description |
|---|---|
| ARCHIVE LOG | Starts or stops automatic archiving of online redo log files, manually (explicitly) archives specified redo log files, or displays information about archives. |
| RECOVER | Performs media recovery on one or more tablespaces, one or more datafiles, or the entire database. |
| SET AUTORECOVERY | ON causes the RECOVER command to automatically apply the default filenames of archived redo log files needed during recovery. No interaction is needed when AUTORECOVERY is set to ON, provided the necessary files are in the expected locations with the expected names. |

*Table B–1 (Cont.) Commands Introduced in SQL*Plus Release 8.1*

| Command | Description |
|---|---|
| SET INSTANCE | Changes the default instance for your session to the specified instance path. Does not connect to a database. The default instance is used for commands when no instance is specified. |
| SET LOGSOURCE | Specifies the location from which archive logs are retrieved during recovery. The default value is set by the LOG_ARCHIVE_DEST initialization parameter. Issuing the SET LOGSOURCE command without a pathname restores the default location. |
| SHOW AUTORECOVERY | Shows whether autorecovery is enabled. |
| SHOW INSTANCE | Shows the connect string for the default instance. SHOW INSTANCE returns the value LOCAL if you have not used SET INSTANCE or if you have used the LOCAL option of the SET INSTANCE command. |
| SHOW LOGSOURCE | Shows the current setting of the archive log location. Displays DEFAULT if the default setting is in effect, as specified by the LOG_ARCHIVE_DEST initialization parameter. |
| SHOW PARAMETERS | Displays the current values of one or more initialization parameters. The SHOW PARAMETERS command, without any string following the command, displays all initialization parameters. |
| SHOW SGA | Displays information about the current instance's System Global Area. |
| SHUTDOWN | Shuts down a currently running Oracle instance, optionally closing and dismounting a database.<br>**Note:** The STARTUP and SHUTDOWN commands in SQL*Plus release 8.1 are not supported against an Oracle7 server. |
| STARTUP | Starts an Oracle instance with several options, including mounting and opening a database.<br>**Note:** The STARTUP and SHUTDOWN commands in SQL*Plus release 8.1 are not supported against an Oracle7 server. |

## Commands Common to Server Manager and SQL*Plus

The commands listed in Table B–2 are available in both Server Manager and SQL*Plus, and have been available in both programs in past releases of Oracle. You do not need to alter these commands in your SQL scripts to use SQL*Plus.

**Note:** There may be minor formatting differences in the output for these commands in the two programs.

*Table B–2 Server Manager Commands Corresponding to Existing SQL*Plus Commands*

| Command | Description |
|---|---|
| CONNECT | Connects to a database using the specified user name. |
| DESCRIBE | Describes a function, package, package body, procedure, table, view, or object type. For example, for a table, displays the definitions of each column in the table. |
| REMARK | Enters a comment, typically in SQL script files. |
| SET COMPATIBILITY | Sets compatibility mode to V7, V8, or NATIVE. The compatibility mode setting affects the specification of character columns, integrity constraints, and rollback segment storage parameters. NATIVE matches the version of the database. |
| SET ECHO | Controls whether the START command lists each command in a command file as the command is executed. ON lists the commands; OFF suppresses the listing. |
| SET NUMWIDTH | Sets the default width for displaying numbers. |

*Table B–2 (Cont.) Server Manager Commands Corresponding to Existing SQL*Plus Commands*

| Command | Description |
|---|---|
| `SET SERVEROUTPUT` | Controls whether to display the output (that is, `DBMS_OUTPUT.PUT_LINE`) of stored procedures or PL/SQL blocks in SQL*Plus. `OFF` suppresses the output of `DBMS_OUTPUT.PUT_LINE`; `ON` displays the output. |
| `SET TERMOUT` | Controls the display of output generated by commands executed from a command file. `OFF` suppresses the display so that you can spool output from a command file without seeing the output on the screen. `ON` displays the output. |
| `SHOW ALL` | Lists all of the system variables set by the `SET` command in alphabetical order, except ERRORS, PARAMETERS, and SGA. |
| `SHOW ERRORS` | Shows the errors generated from the last compilation of a procedure, package, or function, if any. |
| `SPOOL` | Stores query results in an operating system file and, optionally in SQL*Plus, sends the file to a printer.<br>**Note:** The extension of spool files may differ between SQL*Plus and Server Manager. To ensure an extension, specify it when you issue the `SPOOL` command. Also, SQL*Plus may format white space in terminal output using tab characters in place of repeated spaces. Use `SET TAB OFF` in SQL*Plus to prevent this replacement. Server Manager never outputs tab characters. |

## SQL*Plus Equivalents for Server Manager Commands

Table B–3 lists the SQL*Plus commands that correspond to Server Manager commands with different names. If you are using any of these Server Manager commands in SQL scripts, then modify the scripts to use the SQL*Plus commands instead.

*Table B–3 SQL*Plus Equivalents for Server Manager Commands*

| Server Manager Commands | SQL*Plus Commands | Description |
|---|---|---|
| SET CHARWIDTH<br>SET DATEWIDTH<br>SET LONGWIDTH | COLUMN FORMAT | You can use the COLUMN FORMAT command in SQL*Plus to set the column width of character columns, date columns, and number columns. In your SQL scripts, replace the SET CHARWIDTH, SET DATEWIDTH, and SET LONGWIDTH Server Manager commands with the SQL*Plus COLUMN FORMAT command.<br>Use COLUMN FORMAT for all character columns to be changed. There is no equivalent command to change all character columns with one command.<br>For example, suppose you have the following entry in a SQL script:<br>SET CHARWIDTH 5<br>This command sets the width for all character columns to 5 in Server Manager.<br>To specify that a particular column, such as first_name, display with a width of 5 characters, enter the following SQL*Plus command:<br>COLUMN first_name FORMAT A5<br>Use COLUMN FORMAT for all character columns to be changed. There is no equivalent command to change all character columns with one command.<br>Use COLUMN FORMAT for all date columns to be changed. There is no equivalent command to change all date columns with one command.<br>Use SET LONG to specify how much of the LONG column to fetch and display. |
| SET STOPONERROR | WHENEVER SQLERROR<br>WHENEVER OSERROR | Use the WHENEVER SQLERROR and WHENEVER OSERROR commands to direct SQL*Plus to either exit or continue whenever a SQL error or operating system error occurs. Use these commands in your SQL scripts instead of the Server Manager SET STOPONERROR command.<br>For both WHENEVER SQLERROR and WHENEVER OSERROR, the EXIT clause directs SQL*Plus to exit, while the CONTINUE clause directs SQL*Plus to continue. Other terms and clauses are also available for these commands. |

## Possible Differences in the SET TIMING Command

The SET TIMING command is available in both Server Manager and SQL*Plus, but this command may function differently in the two programs on some operating systems. Check your operating system-specific Oracle documentation for more information. If the SET TIMING command functions differently in these two programs on your operating system, then modify your SQL scripts so that this command functions properly with SQL*Plus.

## Server Manager Commands Unavailable in SQL*Plus

The following Server Manager commands are unavailable in SQL*Plus release 8.1 and higher:

- SET MAXDATA
- SET RETRIES

Remove these commands from your SQL scripts.

# Syntax Differences

The following sections explain the syntax differences between Server Manager and SQL*Plus. Modify your SQL scripts to conform with SQL*Plus syntax conventions before you attempt to run your scripts using SQL*Plus.

## Comments

SQL*Plus recognizes the following types of comments:

- the SQL*Plus REMARK command (or REM)
- the SQL comment delimiters, /* ... */
- the ANSI/ISO comments, --

The *SQL*Plus User's Guide and Reference* provides detailed information about using these types of comments in SQL*Plus scripts.

Server Manager supports these types of comments, but the behavior is different for some of them. Also, certain types of comments are available in Server Manager, but not in SQL*Plus. The sections below discuss each type of comment and the syntax differences between Server Manager and SQL*Plus.

### REMARK Command (or REM)

In general, the `REMARK` command works the same in Server Manager and SQL*Plus, and you do not need to change the occurrences of the `REMARK` command in your SQL scripts. There is, however, one difference: SQL*Plus interprets a hyphen that terminates a `REMARK` command differently than Server Manager. See "Hyphens Used as Dividing Lines" on page B-8 for information about this difference.

### SQL Comment Delimiters, /* ... */

In Server Manager, the SQL comment delimiters can be placed after a semicolon (;), but in SQL*Plus, placing a SQL comment delimiter after a semicolon is not allowed. Except for this one difference, SQL comment delimiters work the same in Server Manager and SQL*Plus.

If your SQL scripts contain any SQL comment delimiters placed after a semicolon, then either move the comment to its own line, or remove the semicolon and place a slash (/) on the next line to end the SQL statement.

For example, suppose you have the following Server Manager code in one of your SQL scripts:

```
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK'; /* Includes only clerks. */
```

In SQL*Plus, replace this code with either of the following entries:

```
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK';
/* Includes only clerks. */

SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK' /* Includes only clerks. */
    /
```

### ANSI/ISO Comments, --

In Server Manager, the ANSI/ISO comments can be placed after a semicolon (;), but in SQL*Plus, placing an ANSI/ISO comment after a semicolon is not allowed. Except for this one difference, ANSI/ISO comments work the same in Server Manager and SQL*Plus.

If your SQL scripts contain any ANSI/ISO comments that are placed after a semicolon, then either move the comment to its own line, or remove the semicolon and place a slash (/) on the next line to end the SQL statement.

For example, suppose you have the following Server Manager code in one of your SQL scripts:

```
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK'; -- Includes only clerks.
```

In SQL*Plus, replace this code with either of the following entries:

```
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK';
-- Includes only clerks.

SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK' -- Includes only clerks.
    /
```

### Server Manager Pound (#) Comments

Server Manager supports the use of the pound sign (#) to indicate a comment line. If your scripts contain these comments, then change the '#' to '--' to run the scripts using SQL*Plus.

For example, suppose you have the following Server Manager code in one of your SQL scripts:

```
# This statement returns only clerks.
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK';
```

In SQL*Plus, replace this code with the following entry:

```
-- This statement returns only clerks.
SELECT * FROM hr.employees
    WHERE job_id LIKE '%CLERK';
```

## Blank Lines

Server Manager ignores blank lines within SQL statements, but when SQL*Plus encounters a blank line the default behavior is to stop recording the statement and return to the prompt.

Both products allow blank lines between distinct SQL statements. This section only applies to blank lines between clauses of SQL statements.

In SQL*Plus, the `SET SQLBLANKLINES` command alters the way blank lines are handled. When `SQLBLANKLINES` is set to `OFF`, the default setting, and there is a SQL statement containing a blank line, SQL*Plus buffers the statement at the blank line, returning to the prompt without executing the statement. This behavior allows interactive users to abort and buffer an unwanted SQL command, or to perform other SQL*Plus commands before executing or editing this buffered SQL command.

If any of your SQL scripts contain blank lines within SQL statements, then either set SQLBLANKLINES to ON, or remove the blank lines before you run these scripts using SQL*Plus.

For example, suppose you have the following SQL statement in one of your SQL scripts:

```
SELECT employee_id, first_name, last_name, salary, commission_pct

    FROM hr.employees

    WHERE job_id LIKE '%MAN';
```

Either set SQLBLANKLINES to ON, or delete the blank lines:

```
SELECT employee_id, first_name, last_name, salary, commission_pct
    FROM hr.employees
    WHERE job_id LIKE '%MAN';
```

If you do not remove the blank lines or set SQLBLANKLINES to ON, then SQL*Plus will treat each blank line of code as a command terminator.

The value of SQLBLANKLINES does not affect blank lines in PL/SQL blocks. These are always treated as part of the block and do not return to the SQL*Plus prompt.

Interactive users can terminate SQL or PL/SQL statements by entering a period on a line by itself, regardless of the value of SQLBLANKLINES.

## The Hyphen Continuation Character

SQL*Plus supports the use of a hyphen as a continuation character for long SQL statements or SQL*Plus commands. For example, you can use the continuation character in the following way:

```
SELECT employee_id, first_name, last_name FROM hr.employees -
WHERE job_id LIKE '%MAN';
```

Server Manager does not support the use of a hyphen as a continuation character, but you may use hyphens for other purposes in your SQL scripts. If you do, then SQL*Plus may interpret a hyphen as a continuation character, which can cause unexpected output.

The following sections provide scenarios in which SQL*Plus interprets the use of hyphens in SQL scripts as continuation characters, when the hyphens were meant for another purpose. Check your SQL scripts for the use of hyphens and modify them to avoid scenarios similar to those described below.

### Hyphens Used as Dividing Lines

Your SQL scripts may use a long row of hyphens following a REMARK command as a dividing line in the code. Consider the following sample lines from a SQL script:

```
Rem ------------------------------------------------------------------------
SELECT employee_id, first_name, last_name, job_id
    FROM hr.employees;
```

In this statement, SQL*Plus interprets the first line of the SELECT statement as a continuation of the previous line, which is a REMARK comment. Therefore, the FROM line is interpreted as the first line of a SQL statement, and SQL*Plus returns the following error:

```
unknown command beginning "FROM hr..." - rest of line ignored.
```

If you use hyphens as dividing lines in your SQL scripts, then remove the REM command preceding the hyphens before you run the scripts using SQL*Plus.

### Hyphens Used as Minus Signs

Because the hyphen is the same keyboard character as the minus sign, you may have a hyphen at the end of a line. Consider the following sample lines from a SQL script:

```
CREATE TABLE xx (
    a int,
    b int,
    c int);

INSERT INTO xx VALUES (10, 20, 30);

SELECT a + b -
    c FROM xx;
```

SQL*Plus interprets the 'c' as an alias because the minus symbol is interpreted as a continuation character:

```
SELECT a + b c FROM xx;
```

Therefore, SQL*Plus returns the following unexpected output:

```
         C
----------
        30
```

Server Manager, however, interprets this code as the following:

```
SELECT a + b - c FROM xx;
```

Therefore, Server Manager returns the following expected output:

```
A+B-C
----------
         0
```

Make sure you do not have a minus sign at the end of a line in your SQL scripts.

## Ampersands

SQL*Plus interprets an ampersand (&) as a substitution variable, whereas Server Manager interprets an ampersand as a normal string. If the text following the ampersand does not have a defined value, then SQL*Plus interprets it as an undefined value and prompts the user for input, even if the ampersand is enclosed in a comment. Therefore, ampersands can cause unexpected output in SQL*Plus.

If you have SQL scripts that use ampersands as normal text strings, then you have two options:

- Use the SET ESCAPE command to place an escape character before each ampersand.
- Use the SET DEFINE OFF command to disable the recognition of substitution variables.

> **Note:** Do not use the `SET DEFINE OFF` command if you have other, valid substitution variables; if you do, then the other variables will not be recognized.

For example, the following SQL statement prompts the user for input in SQL*Plus:

```
CREATE TABLE "Employees & Managers" (
    Employees varchar(16),
    Managers varchar(16));

Enter value for managers:
```

### Using the SET ESCAPE Command

To avoid the user prompt, you can use the `SET ESCAPE` command to set an escape character. Then, place the escape character before the ampersand. A backslash (\) is often used as an escape character.

To avoid the prompt in the preceding example by using the `SET ESCAPE` command, change the entry to the following:

```
SET ESCAPE \

CREATE TABLE "Employees \& Managers" (
    Employees varchar(16),
    Managers varchar(16));
```

### Using the SET DEFINE OFF Command

To avoid the prompt in the preceding example by using the `SET DEFINE OFF` command, change the entry to the following:

```
SET DEFINE OFF

CREATE TABLE "Employees & Managers" (
    Employees varchar(16),
    Managers varchar(16));
```

## CREATE TYPE and CREATE LIBRARY Commands

SQL*Plus treats the `CREATE TYPE` and `CREATE LIBRARY` commands as PL/SQL blocks. Therefore, in SQL*Plus, you must use a slash (/) on a separate line to end these commands, while Server Manager allows you to end these commands with a semicolon (;).

If you end any `CREATE TYPE` or `CREATE LIBRARY` command with a semicolon in your SQL scripts, then remove the semicolon and place a slash (/) on the next line. For example, the following SQL statements are not recognized by SQL*Plus:

```
CREATE OR REPLACE TYPE sys.dummy AS OBJECT (data CHAR(1));
CREATE OR REPLACE LIBRARY DBMS_SPACE_ADMIN_LIB TRUSTED AS STATIC;
```

Edit these statements in the following way before you run them with SQL*Plus:

```
CREATE OR REPLACE TYPE sys.aq$_dummy_t AS OBJECT (data CHAR(1))
/
CREATE OR REPLACE LIBRARY DBMS_SPACE_ADMIN_LIB TRUSTED AS STATIC
/
```

## COMMIT Command

SQL*Plus requires that the COMMIT command be terminated either with a semicolon (;) or a slash (/), but Server Manager allows the COMMIT command with no terminator. Therefore, if you use the COMMIT command in your SQL scripts without a terminator, then edit these scripts to include a terminator.

For example, suppose you have the following COMMIT command in a SQL script:

```
commit
```

Include a terminator for the command, as shown in either of the following examples:

```
commit;

commit
/
```

# C

# Gathering Optimizer Statistics

This appendix provides scripts that collect optimizer statistics for dictionary objects. By running these scripts prior to performing the actual database upgrade, you can decrease the amount of downtime incurred during the database upgrade.

This process should be tested on a test database just like any other aspect of the upgrade. Also, some schemas referenced in these scripts may not exist if some database components have not been installed.

## C.1 Collecting Statistics for System Component Schemas

If you are using Release 9.0.1 or 9.2.0, then you should use the `DBMS_STATS.GATHER_SCHEMA_STATS` procedure to gather statistics. The following example shows a sample script that uses this procedure to collect statistics for system component schemas. The statistics collection may give errors if a particular component schema does not exist in the database. This can happen if a component is not installed or if it is invalid.

To run this script, connect to the database `AS SYSDBA` using SQL*Plus.

```
spool gdict

grant analyze any to sys;

exec dbms_stats.gather_schema_stats('WMSYS',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('MDSYS',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('CTXSYS',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);

exec dbms_stats.gather_schema_stats('XDB',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('WKSYS',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('LBACSYS',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);
```

```
exec dbms_stats.gather_schema_stats('OLAPSYS',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);

exec dbms_stats.gather_schema_stats('DMSYS',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('ODM',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('ORDSYS',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);

exec dbms_stats.gather_schema_stats('ORDPLUGINS',options=>'GATHER', estimate_
percent => DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO',
cascade => TRUE);

exec dbms_stats.gather_schema_stats('SI_INFORMTN_SCHEMA',options=>'GATHER',
estimate_percent => DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS
SIZE AUTO', cascade => TRUE);

exec dbms_stats.gather_schema_stats('OUTLN',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

exec dbms_stats.gather_schema_stats('DBSNMP',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);

exec dbms_stats.gather_schema_stats('SYSTEM',options=>'GATHER', estimate_percent
=> DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade
=> TRUE);

exec dbms_stats.gather_schema_stats('SYS',options=>'GATHER', estimate_percent =>
DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt => 'FOR ALL COLUMNS SIZE AUTO', cascade =>
TRUE);

spool off
```

## C.2 Creating a Statistics Table

This script creates the table, `dictstattab`, and exports the statistics for the RDBMS component schemas into it. The export returns an error if a particular component schema does not exist in the database. This can happen if a component is not installed or if it is invalid.

This script is useful when you want to import the statistics back into the database. For example, the following PL/SQL subprograms import the statistics for the `SYS` schema after deleting the existing statistics:

```
EXEC DBMS_STATS.DELETE_SCHEMA_STATS('SYS');
EXEC DBMS_STATS.IMPORT_SCHEMA_STATS('SYS','dictstattab');
```

To run the following script, connect to the database `AS SYSDBA` using SQL*Plus.

```
spool sdict
```

```
grant analyze any to sys;

exec dbms_stats.create_stat_table('SYS','dictstattab');

exec dbms_stats.export_schema_stats('WMSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('MDSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('CTXSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('XDB','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('WKSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('LBACSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('OLAPSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('DMSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('ODM','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('ORDSYS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('ORDPLUGINS','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('SI_INFORMTN_SCHEMA','dictstattab',statown =>
'SYS');
exec dbms_stats.export_schema_stats('OUTLN','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('DBSNMP','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('SYSTEM','dictstattab',statown => 'SYS');
exec dbms_stats.export_schema_stats('SYS','dictstattab',statown => 'SYS');

spool off
```

# D

# Using the Database Upgrade Assistant

Complete the following steps to upgrade the database using the DBUA graphical user interface:

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

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

2. If an ASM instance is detected on the system, then the Upgrade Operations page is displayed. From this page you can choose to upgrade only the ASM instance or the database. If you choose to upgrade the database and if the database is using ASM, then the DBUA will prompt you about whether or not to upgrade the ASM instance along with the database. Oracle recommends that you upgrade the database and ASM in separate DBUA sessions, in separate Oracle homes.

3. At the Selecting a Database Instance screen, select the database you want to upgrade from the Available Databases table. Then, click Next.

   You can select only one database at a time. If you are running the Database Upgrade Assistant from a user account that does not have `SYSDBA` privileges, then you must enter the user name and password credentials to enable `SYSDBA` privileges for the selected database.

   The Database Upgrade Assistant analyzes the database, performing pre-upgrade checks and displaying warnings as necessary:

   - It checks for any redo log files whose size is less than 4 MB. If such files are found, then the Database Upgrade Assistant gives the option to drop/create new redo log files.
   - It checks the parameter file for any obsolete or deprecated initialization parameters

4. At the Creating the SYSAUX Tablespace screen, specify the attributes for the `SYSAUX` tablespace, which is added automatically to all new Oracle Database 10*g* databases you create. Then, click Next.

   **See Also:** *Oracle Database Administrator's Guide* for more information about the `SYSAUX` tablespace

   Many of the attributes of the `SYSAUX` tablespace are set automatically and cannot be modified. For example, the `SYSAUX` tablespace is set to use Automatic Segment-Space Management. However, you can specify the location of the data file, the default size of the `SYSAUX` tablespace, and its autoextend attributes.

**Note:** If you specify an existing datafile for the SYSAUX tablespace, then you must select Reuse Existing File Name. Otherwise, the Database Upgrade Assistant alerts you to the fact that the file already exists.

5. At the Recompiling Invalid Objects screen, decide whether you want the Database Upgrade Assistant to recompile all invalid PL/SQL modules after the upgrade is complete. Then, click Next.

   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.

   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 screen and automatically determines the number of CPUs you have available.

   If the database is in ARCHIVELOG mode, the Database Upgrade Assistant gives you the choice of changing it to NOARCHIVELOG mode.

   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.

6. At the Choosing a Database Backup Procedure screen, specify whether or not you want the Database Upgrade Assistant to back up your database for you. Then, click Next.

If you choose not to use the Database Upgrade Assistant for your backup, then Oracle assumes you have already backed up your database using your own backup procedures. Oracle strongly recommends that you back up your database before the upgrade. If errors occur during the upgrade, then you may need to restore the database from the backup.

---

**Note:** Database Upgrade Assistant does not back up ASM databases. You must manually back them up on your own.

---

If you use the Database Upgrade Assistant to back up your database, then the Database Upgrade Assistant will make a copy of all your database files in the directory you specify in the Backup Directory field. The Database Upgrade Assistant will perform this cold backup automatically after it shuts down the database and before it begins performing the upgrade procedure. The cold backup will not compress your database files and the backup directory must be a valid file system path. You cannot specify a raw device for the cold backup files.

In addition, the Database Upgrade Assistant creates a batch file in the specified directory. You can use this batch file to restore the database files:

- On Windows operating systems, the file is called *db_name*_restore.bat.
- On UNIX platforms, the file is called *db_name*_restore.sh.

7. Note: The Management Options screen described in this step is not displayed if a release 10.1 database is already being monitored with Enterprise Manager.

   At the Management Options screen, you have the option of setting up your database so it can be managed with Enterprise Manager. 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 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 cluster 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 E-mail notifications will be enabled immediately upon installation.

      Select Enable E-mail Notifications if you want the SYSMAN user (the default Super Administrator and owner of the Management Repository schema) to receive E-mail notification when a metric for a specified condition reaches a critical or warning threshold. For example, Enterprise Manager can send an

E-mail 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 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 screen of the Database Upgrade Assistant.

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

**8.** At the Database Credentials screen, 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.

**9.** At the Recovery Configuration screen, 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 screen shown previously in the Database Upgrade Assistant.

**10.** At the Network Configuration for the database screen, you have two tabs:

The Listeners tab is displayed if you have more than one listener in the release 10.2 Oracle home. Select the listeners in the release 10.2 Oracle home for which you would like to register the upgraded database.

The Directory Service tab shows up if you have directory services configured in the release 10.2 Oracle home. You can select to either register or not register the upgraded database with the directory service.

**11.** At the Summary screen, 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.

The Database Upgrade Assistant lists the initialization parameters that will be set for the database during the upgrade. The `COMPATIBLE` initialization parameter will be set to at least `9.2.0`.

> **See Also:** Chapter 5, "Compatibility and Interoperability" for information about setting the `COMPATIBLE` initialization parameter after the upgrade

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

You may encounter error messages with Ignore and Abort choices. If other errors appear, then you must address them accordingly. If an error is severe and cannot be handled during the upgrade, then you have the following choices:

- If Ignore is presented as a choice in the message, then clicking the button will ignore the error and proceed with the upgrade. The errors ignored are logged and shown later in the summary.

  This causes the Database Upgrade Assistant to display the step as skipped and 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 skipped steps.

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

  This will abort 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.

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 the OK button.

**13.** At the Checking Upgrade Results screen, you can examine the results of the upgrade, manage the passwords in the upgraded database, and, if necessary, restore the original database settings.

The upgrade results summary includes a description of the original and upgraded databases and changes made to the initialization parameters. The upgrade results also include a Step Execution Summary that describes the steps performed during the database upgrade. For each step in the process, the summary provides the step name, the log file for the step, and the status. In some cases, you can click the status to display details about the execution step. The Step Execution Summary also includes the directory where the various log files are stored after the upgrade. You can examine any of these log files to obtain more details about the upgrade process.

---

**Note:** An HTML version of the Upgrade Results is also saved in the log files directory.

---

The Password Management section of the screen allows you to unlock and set passwords for various users in the newly upgraded database. Click Configure Database Passwords to display the Password Management dialog box. The Password Management dialog box allows you to change the default password for a user after you upgrade the database. For security reasons, all users are locked except for the following users:

- `SYS`
- `SYSTEM`

If you have enabled Local Management with Enterprise Manager, then the SYSMAN and DBSNMP accounts are also unlocked. These accounts provide Enterprise Manager with access to the database so it can gather monitoring data and so you can perform administration tasks with Enterprise Manager.

If you have enabled Central Management with Enterprise Manager, then the DBSNMP account is unlocked, as well as the SYS and SYSTEM user accounts.

---

**Note:** To prevent unauthorized use of the database, Oracle recommends that you change all user passwords immediately after you upgrade your database.

---

If you are not satisfied with the upgrade results, then click Restore. Depending on the method you used to back up your database, the Restore operation performs one of two tasks:

- If you used the Database Upgrade Assistant to back up your database, then clicking Restore will restore the original database and the original database settings from the backup.
- If you used your own backup procedure to back up the database, then clicking Restore will restore only the original database settings. To restore the database itself, you must restore the backup you created with your own backup utilities.

If you are satisfied with the upgrade results, then click Exit to quit the Database Upgrade Assistant and use your newly upgraded database. The Database Upgrade Assistant removes the entry of the upgraded database from the old listener.ora file and reloads the listener of the old database.

14. At the **Changes in Default Behavior** screen, 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.
15. Complete the procedures described in Chapter 4, "After Upgrading a Database".

---

**WARNING: If you retain the old Oracle software, then never start the upgraded database with the old Oracle software. Only start the database with the executables in the new Oracle Database installation. Also, before you remove the old Oracle environment, make sure you relocate any data files in that environment to the new Oracle Database environment. See the *Oracle Database Administrator's Guide* for information about relocating data files.**

---

# Index

## Numerics



## A



## B



## C



## D

## E



## F



## I



## J



## L



## M

## N



## O



## P



## R



## S

## T



## U

## W