

---

# **Enterprise DBA Part 1A: Architecture and Administration**

**Volume 2 • Instructor Guide**

---

30049GC10

Production 1.0

August 1999

M09008

**ORACLE®**

## **Authors**

Bruce Ernst  
Hanne Rue Rasmussen  
Ulrike Schwinn  
Vijay Venkatachalam

## **Technical Contributors and Reviewers**

David Austin  
Ben van Balen  
Gerry Batista  
Doug Bridges  
Sandra Cheevers  
Ralf Durben  
Ari Fyhr  
Joel Goodman  
Scott Gossett  
Lex de Haan  
Tony Holbrook  
Heike Hundt  
Christine Jeal  
Dominique Jeunot  
Thomas Kerepes  
Steven King  
Pierre Labrousse  
Dean Margolese  
Jean-Marie Misztela  
Tigger Newman  
Howard Ostrow  
Hans Proetzi  
Gary Purcell  
Shankar Raman  
Donalyn Selinsky  
Roger Simon  
James Spiller  
Ramonito Te  
Sabine Teuber  
Jean-Francois Verrier  
Norbert Wittje

## **Publisher**

Sherry Polm

Copyright © Oracle Corporation, 1999. All rights reserved.

This documentation contains proprietary information of Oracle Corporation. It is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited. If this documentation is delivered to a U.S. Government Agency of the Department of Defense, then it is delivered with Restricted Rights and the following legend is applicable:

## **Restricted Rights Legend**

Use, duplication or disclosure by the Government is subject to restrictions for commercial computer software and shall be deemed to be Restricted Rights software under Federal law, as set forth in subparagraph (c) (1) (ii) of DFARS 252.227-7013, Rights in Technical Data and Computer Software (October 1988).

This material or any portion of it may not be copied in any form or by any means without the express prior written permission of Oracle Corporation. Any other copying is a violation of copyright law and may result in civil and/or criminal penalties.

If this documentation is delivered to a U.S. Government Agency not within the Department of Defense, then it is delivered with "Restricted Rights," as defined in FAR 52.227-14, Rights in Data-General, including Alternate III (June 1987).

The information in this document is subject to change without notice. If you find any problems in the documentation, please report them in writing to Education Products, Oracle Corporation, 500 Oracle Parkway, Box SB-6, Redwood Shores, CA 94065. Oracle Corporation does not warrant that this document is error-free.

SQL\*Loader, SQL\*Net, SQL\*Plus, Net8, Oracle Call Interface, Oracle7, Oracle8, Oracle8i, Developer/2000, Developer/2000 Forms, Designer/2000, Oracle Enterprise Manager, Oracle Parallel Server, PL/SQL, Pro\*C, Pro\*C/C++, and Trusted Oracle are trademarks or registered trademarks of Oracle Corporation.

All other products or company names are used for identification purposes only and may be trademarks of their respective owners.

## Contents

### Preface

- Profile xvii
- Related Information xix
- Typographic Conventions xx

### Introduction

- Objectives I-2
- Oracle8i Enterprise Edition I-3
- Database Administrator Tasks I-5
- Course Schedule I-6

### Lesson 1: Oracle Architectural Components

- Objectives 1-2
- Overview 1-3
- Oracle Database Files 1-5
- Other Key Files 1-6
- Oracle Instance 1-7
- Processing a SQL Statement 1-9
- Connecting to a Database 1-10
- Processing a Query 1-12
- The Shared Pool 1-14
- Database Buffer Cache 1-16
- Program Global Area 1-17
- Processing a DML Statement 1-19
- Redo Log Buffer 1-21
- Rollback Segment 1-22
- COMMIT Processing 1-23
- Log Writer 1-25
- Other Instance Processes 1-26
- Database Writer 1-27
- SMON: System Monitor 1-28
- PMON: Process Monitor 1-30
- Archiving 1-31
- Summary 1-33

## **Lesson 2: Getting Started with the Oracle Server**

- Objectives 2-2
- Overview 2-3
- Oracle Universal Installer 2-4
- Validating Privileged Users 2-8
- Oracle Enterprise Manager 2-15
- Summary 2-30

## **Lesson 3: Managing an Oracle Instance**

- Objectives 3-2
- Overview 3-3
- Stages in Startup and Shutdown 3-10
- Starting Up the Instance 3-13
- Changing Database Availability 3-16
- Opening a Database in Read-Only Mode 3-17
- Shutting Down 3-18
- Getting and Setting Parameter Values 3-22
- Managing Sessions 3-29
- Summary 3-36

## **Lesson 4: Creating a Database**

- Objectives 4-2
- Overview 4-3
- Preparing the Operating System 4-4
- Creating a Database 4-10
- Using the Database Configuration Assistant 4-11
- Creating a Database Manually 4-15
- Summary 4-29

## **Lesson 5: Creating Data Dictionary Views and Standard Packages**

- Objectives 5-2
- Overview 5-3
- Data Dictionary Overview 5-4
- Data Dictionary Contents 5-5

- Base Tables and Data Dictionary Views 5-6
- How the Data Dictionary Is Used 5-7
- Data Dictionary View Categories 5-8
- Data Dictionary Examples 5-10
- Dynamic Performance Views 5-11
- Stored Program Units 5-12
- Stored PL/SQL Program Units 5-14
- Packages 5-16
- Executing a PL/SQL Program Unit 5-17
- Package Specification and Body 5-18
- Oracle-Supplied Packages 5-19
- Obtaining Information 5-20
- Troubleshooting 5-22
- Constructing the Data Dictionary 5-23
- Administrative Scripts 5-25
- Built-in Package Example 5-27
- Triggers 5-29
- Parts of a Trigger 5-31
- Trigger Example 5-33
- Summary 5-34

## **Lesson 6: Maintaining the Control File**

- Objectives 6-2
- The Use of the Control File 6-3
- Control File Contents 6-4
- Multiplexing the Control File 6-6
- Guidelines for Control Files 6-7
- Obtaining Information About the Control File 6-9
- Summary 6-11

## **Lesson 7: Maintaining Redo Log Files**

- Objectives 7-2
- Overview 7-3
- Using Online Redo Files 7-4

- LGWR, Log Switches, and Checkpoints 7-6
- Archiving Redo Log Files 7-8
- Obtaining Log and Archive Information 7-11
- Controlling Log Switches and Checkpoints 7-18
- Multiplexing and Maintaining Members and Groups 7-21
- Relocating or Renaming Online Redo Log Files 7-24
- Dropping Online Redo Log Groups and Members 7-26
- Planning Online Redo Logs 7-30
- Troubleshooting 7-32
- Using LogMiner 7-33
- Summary 7-40

## **Lesson 8: Managing Tablespaces and Data Files**

- Objectives 8-2
- Overview 8-3
- Database Storage Hierarchy 8-4
- SYSTEM and Non-SYSTEM Tablespaces 8-7
- Creating Tablespaces 8-8
- Space Management in Tablespaces 8-12
- Locally Managed Tablespaces 8-13
- Temporary Tablespace 8-15
- Changing the Storage Settings 8-18
- Taking Tablespaces Offline or Online 8-20
- Read-Only Tablespaces 8-23
- Dropping Tablespaces 8-26
- Resizing a Tablespace 8-29
- Enabling Automatic Resizing of Data Files 8-30
- Manually Resizing Data Files 8-33
- Adding Data Files to a Tablespace 8-34
- Moving Data Files 8-36
- Data Dictionary Information 8-39
- Guidelines 8-40
- Summary 8-42

## **Lesson 9: Storage Structure and Relationships**

- Objectives 9-2
- Overview 9-3
- Types of Segments 9-4
- Storage Clause Precedence 9-8
- Extent Allocation and Deallocation 9-9
- Used and Free Extents 9-10
- Using Block Space Utilization Parameters 9-11
- Obtaining Information About Storage Structures 9-16
- Querying DBA\_SEGMENTS 9-17
- Querying DBA\_EXTENTS 9-18
- Querying DBA\_FREE\_SPACE 9-19
- Planning the Location of Segments 9-20
- Summary 9-22

## **Lesson 10: Managing Rollback Segments**

- Objectives 10-2
- Overview 10-3
- Rollback Segments 10-4
- Using Rollback Segments with Transactions 10-8
- Creating Rollback Segments 10-12
- Maintaining Rollback Segments 10-18
- Obtaining Rollback Segment Information 10-26
- Planning Rollback Segments 10-32
- Troubleshooting Rollback Segment Problems 10-34
- Summary 10-40

## **Lesson 11: Managing Tables**

- Objectives 11-2
- Overview 11-3
- Oracle Data Types 11-7
- Creating a Table 11-17
- Controlling Space Used by Tables 11-25
- Retrieving Table Information 11-41
- Summary 11-47

## **Lesson 12: Managing Indexes**

- Objectives 12-2
- Overview 12-3
- Creating Indexes 12-12
- Reorganizing Indexes 12-22
- Dropping Indexes 12-29
- Obtaining Index Information 12-31
- Summary 12-33

## **Lesson 13: Maintaining Data Integrity**

- Objectives 13-2
- Overview 13-3
- Integrity Constraints 13-5
- Implementing Constraints 13-14
- Maintaining Constraints 13-19
- Getting Constraint Information 13-26
- Summary 13-29

## **Lesson 14: Loading Data**

- Objectives 14-2
- Overview 14-3
- Loading Data Using Direct-Load Insert 14-5
- Loading Data Using SQL\*Loader 14-8
- Direct Path Loading 14-28
- Summary 14-29

## **Lesson 15: Reorganizing Data**

- Objectives 15-2
- Overview 15-3
- Transportable Tablespaces 15-23
- Transporting a Tablespace 15-25
- Exporting and Importing Metadata 15-26
- Transporting a Tablespace 15-28
- Transportable Tablespace Uses 15-29



Transportable Tablespaces and Schema Objects 15-30

Checking the Transport Set 15-31

Summary 15-32

## **Lesson 16: Managing Password Security and Resources**

Objectives 16-2

Overview 16-3

Administering Passwords 16-5

Altering and Dropping a Profile 16-20

Controlling Usage of Resources 16-24

Viewing Password and Resource Limits Information 16-31

Summary 16-33

## **Lesson 17: Managing Users**

Objectives 17-2

Overview 17-3

Creating New Database Users 17-6

Altering and Dropping Database Users 17-14

Dropping Users 17-17

Monitoring Information About Users 17-18

Summary 17-20

## **Lesson 18: Managing Privileges**

Objectives 18-2

Overview 18-3

System Privileges 18-4

Granting System Privileges 18-6

Password File Authentication 18-9

Displaying System Privileges 18-11

Revoking System Privileges 18-14

Object Privileges 18-17

Granting Object Privileges 18-18

Displaying Object Privileges 18-20

Revoking Object Privileges 18-21

- Auditing Guidelines 18-25
- Using Database Auditing 18-29
- Viewing Auditing Results 18-36
- Summary 18-37

## **Lesson 19: Managing Roles**

- Objectives 19-2
- Overview 19-3
- Creating and Modifying Roles 19-5
- Assigning Roles 19-11
- Controlling Availability of Roles 19-13
- Displaying Role Information 19-22
- Using Fine-Grained Access Control 19-23
- Summary 19-25

## **Lesson 20: Using National Language Support**

- Objectives 20-2
- Overview 20-3
- Choosing a Database and a National Character Set 20-5
- Specifying Language-Dependent Behavior 20-11
- NLS Parameters and SQL Functions 20-19
- NLS Parameters in SQL Functions 20-22
- Linguistic Index Support 20-26
- Importing and Loading Data Using NLS 20-27
- Obtaining Information About NLS Settings 20-28
- Summary 20-33

## **Appendix A: Practices**

- Environment A-2
- Practice 1: Oracle Architectural Components A-3
- Practice 2: Getting Started With Oracle A-5
- Practice 3: Managing an Oracle Instance A-6
- Practice 4: Creating a Database A-8
- Practice 5: Creating Data Dictionary Views and Standard Packages A-9

Practice 6: Maintaining the Control File	A-10
Practice 7: Maintaining Redo Log Files	A-11
Practice 8: Managing Tablespaces and Data Files	A-12
Practice 9: Storage Structure and Relationships	A-13
Practice 10: Managing Rollback Segments	A-14
Practice 11: Managing Tables	A-15
Practice 12: Managing Indexes	A-17
Practice 13: Maintaining Data Integrity	A-19
Practice 14: Loading Data	A-20
Practice 15: Reorganizing Data	A-21
Practice 16: Managing Password Security	A-22
Practice 17: Managing Users	A-23
Practice 18: Managing Privileges	A-24
Practice 19: Managing Roles	A-25
Practice 20: Using National Language Support	A-26

## **Appendix B: Hints**

Practice 1: Oracle Architectural Components	B-2
Practice 2: Getting Started With Oracle	B-3
Practice 3: Managing an Oracle Instance	B-4
Practice 4: Creating a Database	B-7
Practice 5: Creating Data Dictionary Views and Standard Packages	B-8
Practice 6: Maintaining the Control File	B-9
Practice 7: Maintaining Redo Log Files	B-10
Practice 8: Managing Tablespaces and Data Files	B-12
Practice 9: Storage Structure and Relationships	B-14
Practice 10: Managing Rollback Segments	B-16
Practice 11: Managing Tables	B-18
Practice 12: Managing Indexes	B-20
Practice 13: Maintaining Data Integrity	B-22
Practice 14: Loading Data	B-23
Practice 15: Reorganizing Data	B-25
Practice 16: Managing Password Security	B-26

Practice 17: Managing Users	B-27
Practice 18: Managing Privileges	B-28
Practice 19: Managing Roles	B-29
Practice 20: Using National Language Support	B-30

## **Appendix C: Practice Solutions for SQL\*Plus**

Practice 1 Solutions	C-2
Practice 2 Solutions	C-4
Practice 3 Solutions	C-7
Practice 4 Solutions	C-16
Practice 5 Solutions	C-20
Practice 6 Solutions	C-24
Practice 7 Solutions	C-27
Practice 8 Solutions	C-33
Practice 9 Solutions	C-40
Practice 10 Solutions	C-46
Practice 11 Solutions	C-54
Practice 12 Solutions	C-59
Practice 13 Solution	C-63
Practice 14 Solutions	C-69
Practice 15 Solutions	C-75
Practice 16 Solutions	C-80
Practice 17 Solutions	C-84
Practice 18 Solutions	C-87
Practice 19 Solutions	C-92
Practice 20 Solutions	C-95

## **Appendix D: Practice Solutions for Oracle Enterprise Manager**

Practice 1 Solutions	D-2
Practice 2 Solutions	D-3
Practice 3 Solutions	D-4
Practice 4 Solutions	D-8
Practice 5 Solutions	D-9
Practice 6 Solutions	D-10

Practice 7 Solutions D-12
Practice 8 Solutions D-19
Practice 9 Solutions D-25
Practice 10 Solutions D-26
Practice 11 Solutions D-37
Practice 12 Solutions D-42
Practice 13 Solutions D-47
Practice 14 Solutions D-49
Practice 15 Solutions D-52
Practice 16 Solutions D-59
Practice 17 Solutions D-64
Practice 18 Solutions D-71
Practice 19 Solutions D-76
Practice 20 Solutions D-80

## **Appendix E: Certification Test: Sample Questions**

Oracle Certified Professional (OCP) Program: Oracle Certified Database Administrator Track E-2
Oracle Database Administration: Sample Test E-3
Oracle Backup and Recovery Sample Test E-5
Answers E-8
Registering for an OCP Test E-9



## Reorganizing Data

## Objectives

### Objectives

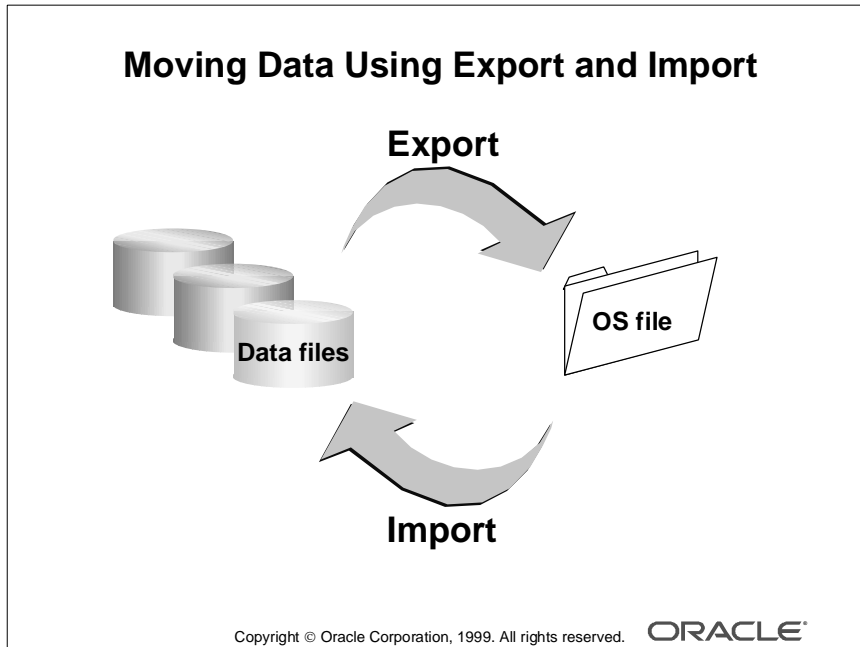
**After completing this lesson, you should be able to do the following:**

- **Reorganize data using the Export and Import utilities**
- **Move data using transportable tablespaces**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**



## Overview



### The Export and Import Utilities

Export and Import utilities enable the administrator to move data between Oracle databases, and within an Oracle database, to different tablespaces or users, or to reorganize data for efficient storage and performance.

#### Export Utility

The Export utility can be used to make a logical copy of object definitions and data to an operating system binary file. Export can write to a file on disk or tape. The Export utility extracts a consistent view of data within each table.

#### Import Utility

The Import utility can read the operating system files created by the Export utility and copy the object definitions and data into an Oracle database. The Import utility cannot read text files or files created in any other format.

### Uses of Export and Import

- **Reorganize tables**
- **Move data owned by one user to another user**
- **Move data between databases:**
  - **Development to production**
  - **OLTP system to a data warehouse**
- **Migrate the database to a different:**
  - **OS platform**
  - **Release of the Oracle database**
- **Repeat test runs during development or upgrade**
- **Perform a logical backup**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Using Export and Import

Export and Import can be used in the following cases:

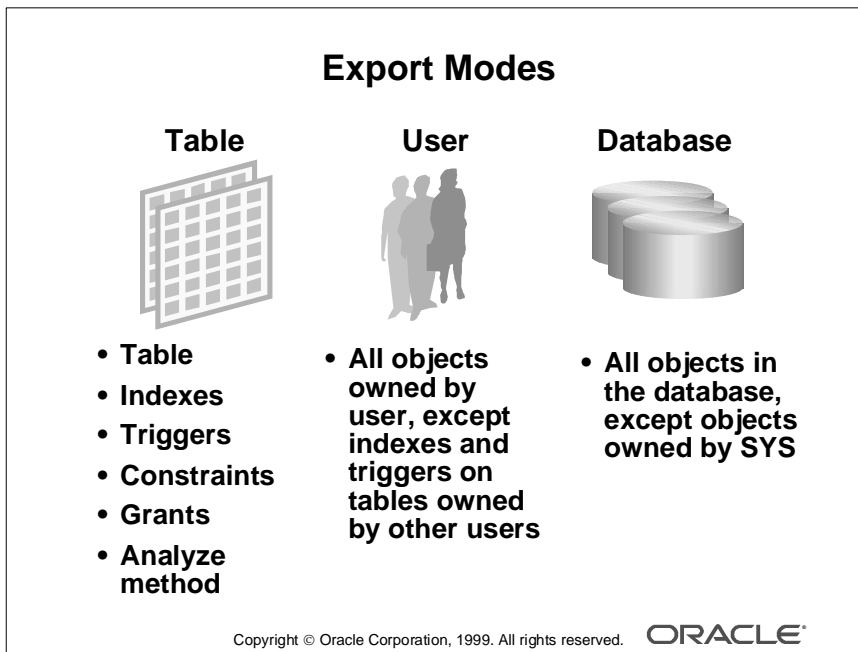
- **Reorganize tables:** There are many instances where tables need reorganization:
  - Data in one tablespace may need to be moved from one tablespace to another to minimize contention, reduce free-space fragmentation, or to facilitate backup.
  - A table may contain many migrated rows.
  - A table may have many blocks with a large amount of freespace.
  - A table may have many empty blocks below the high-water mark.
- **Move data owned by one user to another user:** This may be necessary when a schema needs to be removed from the database or to redistribute object ownership. Data that was exported from one user can be imported into a different user's schema.
- **Move data between databases:** Object definitions can be moved from development to production by extracting only definitions and disregarding data. Export and Import can also be used to extract data from an OLTP application into a data warehouse.
- **Migrate to a different operating system platform or release of Oracle:** Data that is exported on one machine can be imported into a database on a different machine, possibly using a different character set.

**Using Export and Import (continued)**

- **Migrate to a different release of the Oracle database:** When you are upgrading to a new release of Oracle server, data can be exported from the older release and imported into the new release. Note that it may not be possible to use this method for moving data from a later release to an earlier release.
- **Repeat test runs during development or upgrade:** In a development or test database, an application may require several test runs before it is fully debugged and accepted. Test data can be exported to an external file and imported before each run to ensure that tests are performed on the same set of data. This method is also useful to test a new version of the Oracle server before a production database is upgraded.
- **Perform a logical backup:** All or some objects in a database can be exported, and the export file can be used as a logical backup.

In this lesson, the use of Export and Import for data reorganization and moving data between users is presented.

**Note:** The use of Export and Import for backup and recovery is discussed in detail in the course *Enterprise DBA Part 1B: Backup and Recovery*.



### Three Export Modes

The Export utility provides three modes of export:

- Table
- User
- Database

#### Table Mode

All users can use the table mode to export their own tables. Privileged users can export tables owned by any user. The use of the table mode exports:

- The table definition
- Data in the table, if required
- All indexes on the table if the export is performed by a privileged user (Otherwise, only those indexes on the table owned by the user are exported.)
- All triggers on the table only if the utility is run by a privileged user (Otherwise, only the triggers on the table owned by the user are exported.)
- Constraints on the table
- All grants made on the table
- Definition of the analyze method to use on import

## User Mode

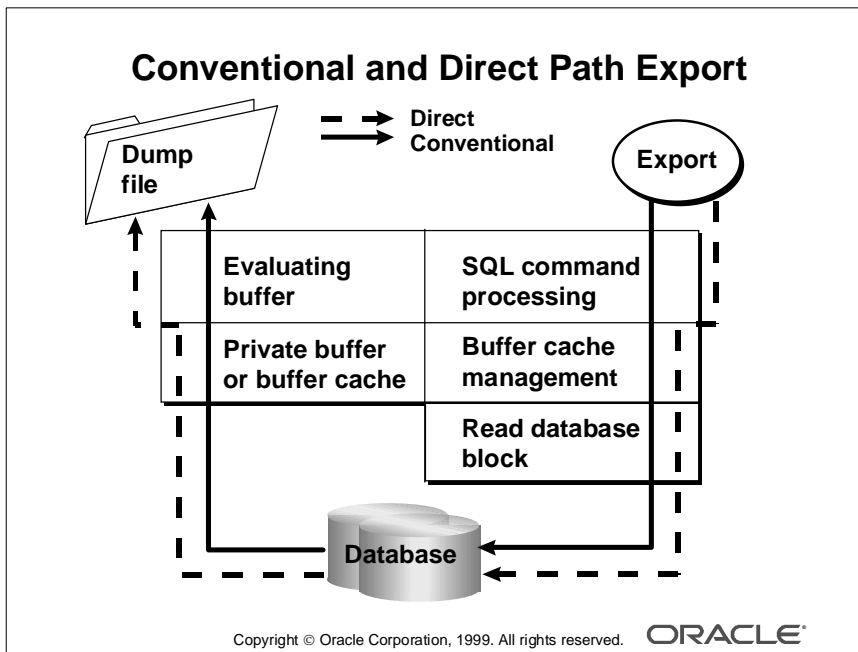
User mode export works differently depending on whether the user running the export has special privileges.

- A privileged user can export objects owned by any user. In this case, the objects exported are:
  - All objects owned by the user, except indexes and triggers that are owned by the user but are on tables owned by other users
  - Triggers and indexes created by other users on the user's tables
- Nonprivileged users can export only objects owned by them, and this mode will not include any indexes or triggers that are created by other users on the tables owned by this user.

## Full Database Mode

All objects in the database, except those owned by the user SYS, are exported when using this mode. This mode requires special privileges and cannot be used by all the users.

**Note:** In all the three modes of export, privileged users are users with the EXP\_FULL\_DATABASE role. This role is discussed in the lesson “Managing Roles.”



## Export Paths

The slide shows the difference between conventional path and direct path exports.

### Conventional Path

This term refers to the default method of formatting data from a database and writing it out to an export file. Conventional path export uses the SQL `SELECT` statement to extract data from tables. Data is read from disk into a buffer cache, and rows are transferred to the evaluation buffer. The data, after passing expression evaluation, is transferred to the export client, which then writes the data into the export file.

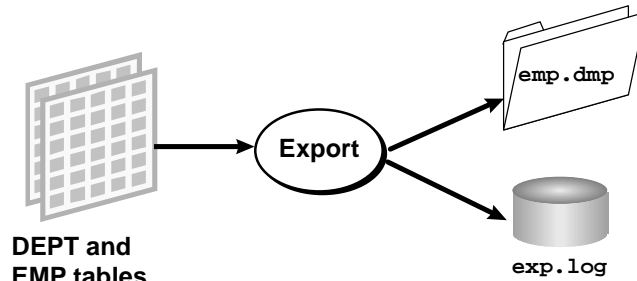
### Direct Path

Direct path export extracts data much faster than a conventional path export by reading data directly and bypassing the SQL Command Processing layer. In a direct path export, data is read from disk into the buffer cache and rows are transferred directly to the export process. The evaluating buffer is bypassed—that is, data in the blocks is not reorganized to bring row pieces together. The data is already in the format that Export expects, thus avoiding unnecessary data conversion. The data is transferred to the export process, which then writes the data into the export file.

The Import utility is capable of using an export file created by any of the paths. The time taken to perform the import is not affected significantly by the export path used.

## Using Export

```
$exp scott/tiger tables=(dept,emp) \
> file=emp.dmp log=exp.log \
> compress=n direct=y
```



Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE

## Using Export

Export can be invoked using:

- Command line
- Interactive mode
- Graphical interface, where available

The interactive mode is primarily provided for backward compatibility and does not offer the full range of options that the command line provides. As a result, use of command line mode is recommended.

## Command Line

Use the following command on UNIX or Windows NT to perform an export:

```
$exp [keyword=]{value|(value, value ...)}
[ [ [,] keyword=]{value|(value, value ...)} ] ...
```

where:	keyword	is one of the keywords discussed in the next section
	value	is the value assigned to the keyword

## Command Line (continued)

### Note

- If keywords are not specified, the values must be specified in the correct order. Although this option is available, it is generally advisable to use the keywords.
- As shown in the slide, it is possible to specify the first few values without keywords and then specify other values with keywords.
- Some operating systems, such as UNIX, use escape characters before special characters, such as a parenthesis, so that the character is not treated as a special character.

## Windows NT: Command Line

Use the same command on Windows NT to perform an export:

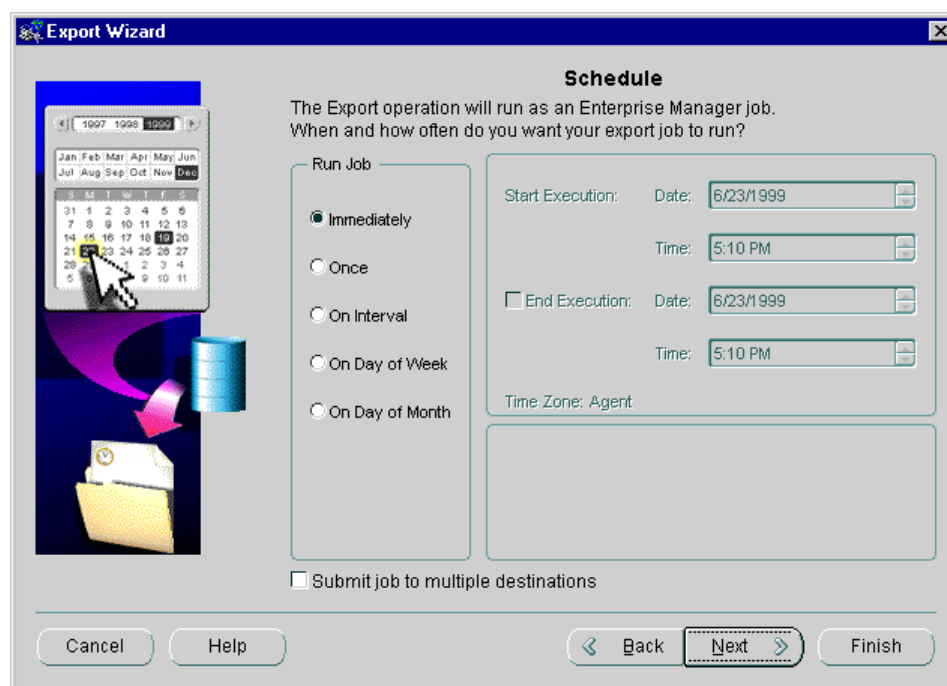
```
C:\>EXP [keyword=]{value|(value, value ...)}  
[ [ [,] keyword=]{value|(value, value ...)} ] ...
```

**Note:** Views required by the Export and Import utilities are created by running the `catexp.sql` script, which is invoked when the `catalog.sql` script is executed.



## How to Use Oracle Enterprise Manager to Export Data

- 1 Launch the Oracle Enterprise Manager console:  
(N) Start—>Programs—>Oracle - *EMV2 Home*—>Oracle Enterprise Management—>Enterprise Manager Console
- 2 Enter administrator, password, and management server. Click OK to log in to the console.
- 3 Expand the Databases folder.
- 4 Select your working database and choose Data Management—>Export from the right mouse menu.
- 5 Enter the export filename on the Export Filepage, and click Next.
- 6 Specify the type of export in the Export Type page, and click Next.
- 7 Specify the associated objects, and click Next.
- 8 Select objects to be exported on the Object Selection page of the Data Manager Wizard.
- 9 Specify associated objects, such as indexes and rows, to be exported and the path on the Associated Objects page.
- 10 Specify schedule parameters, and click Finish.



- 11 Verify the entered parameters in the Summary page and click OK.

**Command Line Parameters**

Some of the commonly used parameters are shown below.

Keyword	Default	Meaning
USERID		Oracle username and password. If password is not specified, the user will be prompted for the password.
BUFFER	OS specific	Size of the buffer that will be used for storing the rows fetched before they are written to the export file
COMPRESS	Y	A value of Y specifies that on import the initial extent size will be set to a value that is equal to the current size of the segment. A value of N will cause the current extent sizes to be retained. The choice has to be made at export because the information gets written to the export file. LOB segments are not compressed.
CONSISTENT	N	A value of Y specifies that the entire export operation be performed in one read-only transaction. Export will attempt to get a read-consistent image of all the objects exported. A value of N specifies that only table-level consistency needs to be maintained.
CONSTRAINTS	Y	A value of Y specifies that constraints are to be exported with the table. A value of N causes constraints not to be exported.
DIRECT	N	A value of Y specifies that direct path be used for the export. A value of N uses conventional path.
FEEDBACK	0	This parameter is specified as an integer <i>n</i> to request for a dot (.) to be displayed when <i>n</i> rows are exported. Zero, the default, indicates that no dots are displayed.
FILE	expdat.dmp	Output filename
FULL	N	A value of Y specifies full database export.
GRANTS	Y	A value of Y specifies that all the grants on objects exported must also be preserved on import.

Keyword	Default	Meaning
HELP	N	A value of Y displays a list of the parameters and their meanings. This parameter is not combined with other parameters.
INDEXES	Y	A value of Y causes indexes to be exported.
LOG	NULL	The name of the file to store all export messages. By default, message are displayed only on the screen.
OWNER		The names of the users for user-level export
PARFILE		Specifies the name of the file that contains a list of export parameters
RECORDLENGTH	OS specific	The size of the output record
ROWS	Y	A value of Y specifies that data is to be exported.
STATISTICS	ESTIMATE	Specifies the analyze method to be used on import
TABLES		<code>schema.table</code> for table mode export

**Note**

- Only one of the parameters, `FULL=Y`, `OWNER=user`, or `TABLES=schema.table`, can be defined.
- If direct path is specified (`DIRECT=Y`), the `CONSISTENT` parameter cannot be set to Y.
- The parameters defined here are not exhaustive. For a complete reference, see the “Export” chapter in the manual, *Oracle8i Utilities*.

```
$imp scott/tiger tables=(dept,emp) \  
> file=emp.dmp log=imp.log ignore=y
```

The diagram illustrates the workflow of the Oracle Import utility. It begins with a folder icon labeled 'emp.dmp', representing the dump file. An arrow points from this folder to a central oval labeled 'Import', which represents the utility itself. From the 'Import' oval, two arrows branch out: one points to a small cylinder icon labeled 'imp.log', representing the log file, and the other points to a larger cylinder icon labeled 'Database', representing the target database.

Import can be invoked using:

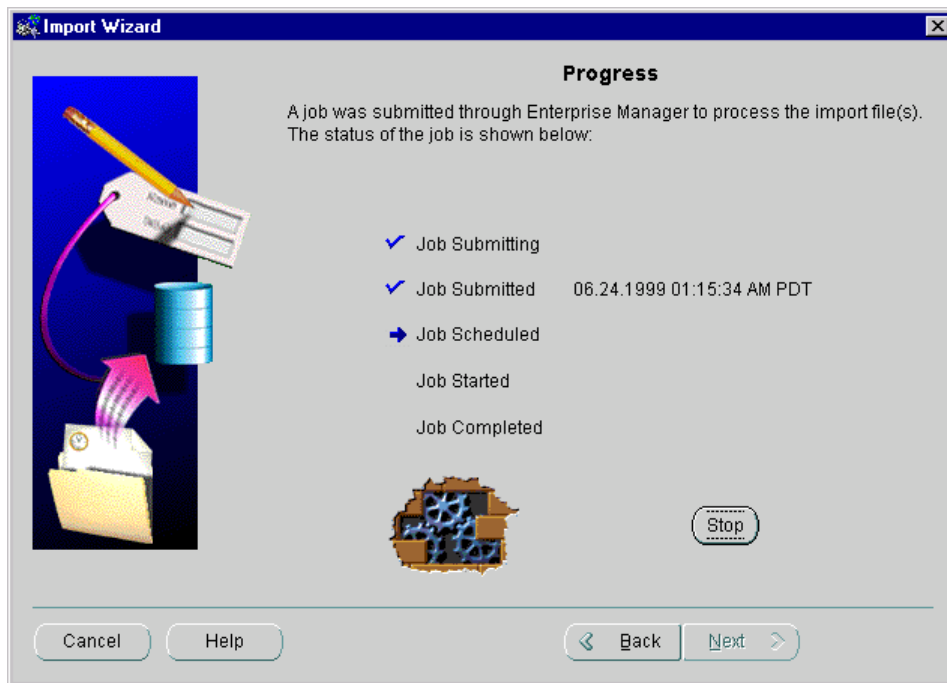
- The interactive mode is primarily provided for backward compatibility and does not offer the full range of options that the command line provides. As a result, use of command line mode is recommended.

Use the following command on UNIX or Windows NT to perform an export:

where:	keyword	is one of the keywords discussed in the next section
	value	is the value assigned to the keyword

The Windows command is the same as for UNIX:

15-14 Enterprise DBA Part 1A: Architecture and Administration



## How to Use Oracle Enterprise Manager to Import Data

- 1 Launch the Oracle Enterprise Manager console:  
(N) Start—>Programs—>Oracle - *EMV2 Home*—>Oracle Enterprise Management—>Enterprise Manager Console
- 2 Enter administrator, password, and management server. Click OK to log in to the console.
- 3 Expand the Databases folder.
- 4 Select your working database and choose Data Management—>Import from the right mouse menu.
- 5 Specify the filename of the export file that should be imported, and click Next.
- 6 Wait for the job to finish.

## Command Line Parameters

Some of the commonly used parameters are shown below.

Keyword	Default	Meaning
USERID		Oracle username and password (If password is not specified, user will be prompted for the password.)
BUFFER	OS specific	Size, in bytes, of the buffer through which data rows are transferred
COMMIT	N	A value of Y specifies that Import should commit after each array insert. By default, Import commits only after loading each table, and Import performs a rollback when an error occurs, before continuing with the next object. Specifying COMMIT=Y prevents rollback segments from growing inordinately large.
FEEDBACK	0	This parameter is specified as an integer <i>n</i> to request for a dot (.) to be displayed when <i>n</i> rows are imported. The default value suppresses the display.
FILE	expdat.dmp	Input filename
FROMUSER	NULL	A list of users whose objects are to be imported.
FULL	N	A value of Y specifies full database import.
GRANTS	Y	A value of Y specifies that all the grants on objects imported must also be imported.
HELP	N	A value of Y displays a list of the parameters and their meanings. This parameter is not combined with other parameters.
IGNORE	N	<p>If the value is set to Y, Import overlooks object creation errors when it attempts to create database objects. In this case, Import continues without reporting the error. For tables, IGNORE=Y causes rows to be imported into existing tables. No message is given. IGNORE=N causes an error to be reported, and the table is skipped if it already exists.</p> <p>Note that only object creation errors are ignored; other errors, such as operating system, database, and SQL errors, are not ignored and may cause processing to stop.</p>

Keyword	Default	Meaning
INDEXES	Y	A value of Y causes indexes to be imported.
INDEXFILE	NULL	Specifies a file to receive index-creation commands. When this parameter is specified, index-creation commands for the requested mode are extracted and written to the specified file, rather than used to create indexes in the database. Tables and other database objects are not imported.  The file can then be edited (for example, to change storage parameters) and used as a SQL script to create the indexes.
LOG	NULL	The name of the file to store all import messages. By default, the messages are displayed only on the screen.
PARFILE		Specifies the name of the file that contains a list of import parameters
RECORDLENGTH	OS specific	The size of the input record. This is necessary only if data was exported on an operating system with a different record size.
ROWS	Y	A value of Y specifies that data is to be imported.
SHOW	N	If the value is Y, the contents of the export file are listed to the display and not imported. The SQL statements contained in the export are displayed in the order in which import will execute them. If SHOW=Y, the only other parameters that can be set are FROMUSER, TOUSER, FULL, and TABLES.
TABLES	NULL	Names of the tables to import
TOUSER	NULL	A list of user names to import tables. Only users with IMP_FULL_DATABASE role can use this parameter to import objects into another user's account.

**Note**

- Only one of the parameters, FULL=Y, OWNER=*user*, or TABLES=*schema.table*, can be defined.
- The IMP\_FULL\_DATABASE role is covered in the lesson “Managing Roles.”
- The parameters defined here are not exhaustive. For a complete reference, see the “Import” chapter in the manual, *Oracle8i Utilities*.

## Import Behavior

- **Order of import:**  
**Table—>Data—>B-tree indexes—>constraints, triggers, bitmap indexes**
- **Tablespace used for the object:**
  - **Same tablespace as in the source database, if possible**
  - **User's default tablespace**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Order of Import

Table objects are imported as they are read from the export file. The export file contains objects in the following order:

- 1** Type definitions
- 2** Table definitions
- 3** Table data
- 4** Table indexes
- 5** Integrity constraints, views, procedures, and triggers
- 6** Bitmap, functional, and domain indexes

This sequence prevents data from being rejected because of the order in which tables are imported. This sequence also prevents redundant triggers from firing twice on the same data (once when it was originally inserted and again during the import).

However, some objects such as procedures may be invalidated on import because they are imported before the objects they reference. Check the objects with `STATUS=INVALID` and recompile them.



### **Considerations for Importing into Existing Tables**

When data is imported into existing tables, the order of import can still produce referential integrity failures. A similar situation occurs when a referential integrity constraint on a table references itself at the end of the import.

For the reasons mentioned previously, it is a good idea to disable referential constraints when importing into an existing table. The constraints can be reenabled after the import is completed.

### **Tablespace Used for an Object**

If a user has the necessary quota, the tables are imported into the same tablespace from which they were exported. However, if the tablespace no longer exists or the user does not have the necessary quota, Import creates the table in the default tablespace for that user. If a user is unable to access the default tablespace, the table cannot be imported.

A LOB segment can only be imported into the same tablespace from which it was exported. So a table containing LOBs will not be created if the owner of the table cannot create objects in the tablespace from which the LOB segment was exported.

**Note:** Quotas and controlling use of tablespaces by users are discussed in the lesson “Managing Users.”

### Export and Import Guidelines

- Use a parameter file to specify commonly used command line options.
- Use **CONSISTENT=Y** only if exporting a small volume of data.
- Do not use **COMPRESS=Y** if there are many deleted rows.
- Improve performance by:
  - Allocating large buffer size
  - Using direct path if using 7.3.3 or higher

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

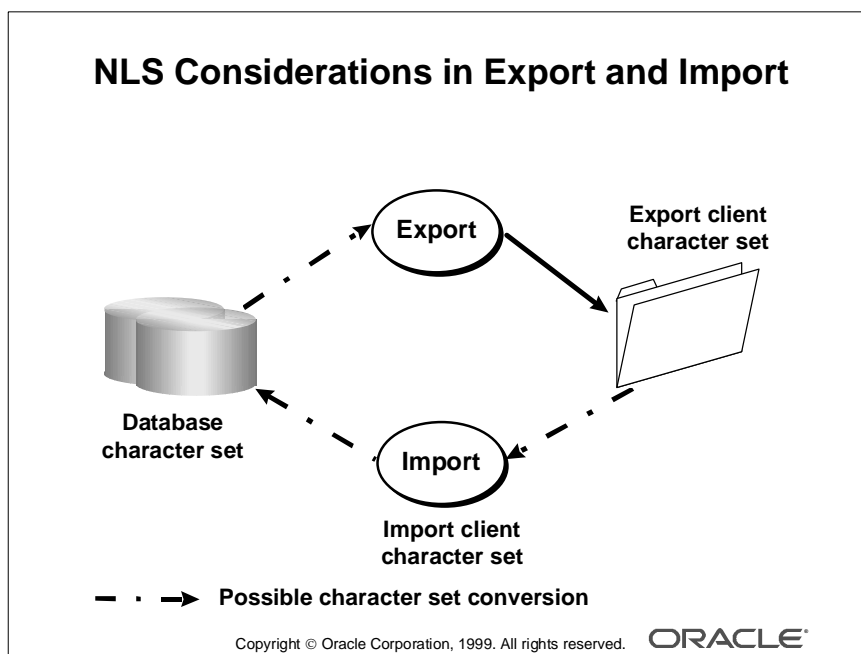
### Export and Import Guidelines

Use a parameter file to store commonly used line parameters. This minimizes errors and keeps the command line smaller.

If there is heavy update activity on tables that are being exported, using **CONSISTENT=Y** is likely to produce **SNAPSHOT TOO OLD** errors. Generally, it is preferable to run large exports during periods of low activity. Alternatively, create a large rollback segment, take all others offline, and perform the import.

The export option **COMPRESS=Y** will generate code to create an initial extent, which is equal to the sum of the sizes of all the extents currently allocated to an object. If the object has many deleted rows or if the last extent has many unused blocks, this will unnecessarily allocate a lot of space for the object.

Allocate as large a buffer as the operating system and the resources on the machine allow. Use direct path export if the data will be imported into a database running release 7.3.3 or higher.



## Export and Character Set Conversion

Conventional path export writes export files using the character set specified for the user session, for example, 7-bit ASCII or IBM CODE Page 500 (EBCDIC).

Direct path export exports in the database character set only. If the character set of the export session is not the same as the database character set when an export is initiated, Export displays a warning and aborts. Specify the session character set to be the same as that of the database before retrying the export.

The export file contains a flag that shows the character encoding scheme used for its character data.

## Import and Character Set Conversion

The import session and the target database character set can differ from the source database character set. This situation requires one or more character set conversion operations.

If necessary, the export file data is first converted during import to the character encoding scheme specified for the user session, and then to the database character set.

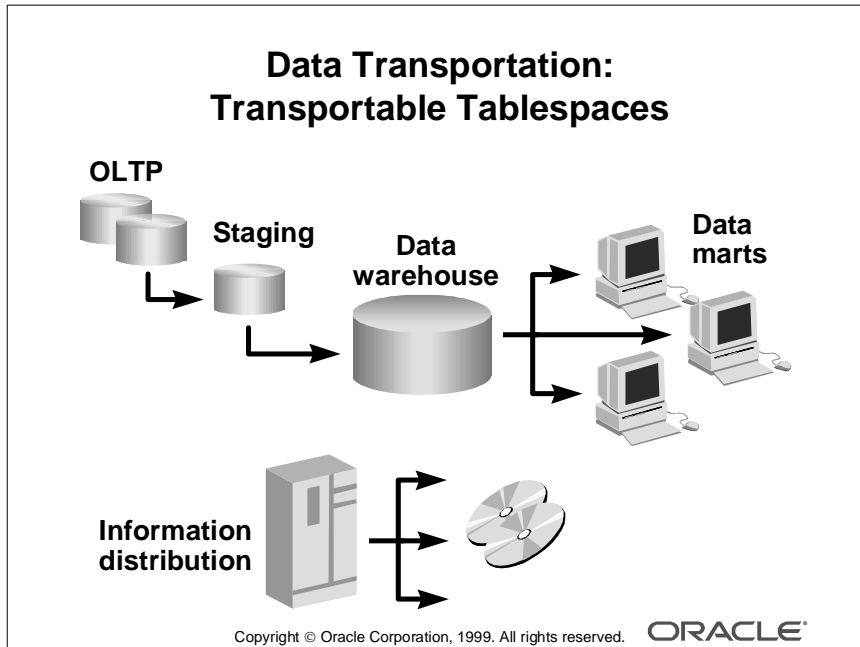
During the conversion, any characters in the export file that have no equivalent in the target character set are replaced with a default character, which is defined by the target character set. To guarantee 100% conversion, the target character set must be a superset of or equivalent to the source character set.

### **Guidelines**

Because character set conversion lengthens the processing time required for import, limit the number of character set conversions to as few as possible. In the ideal scenario, the import session and target database character sets are the same as the source database character sets, requiring no conversion.

**Note:** The setting of the character set at the session level and its impact on various database operations are covered in detail in the lesson “Using National Language Support.”

## Transportable Tablespaces



### Transportable Tablespaces

Moving data from a data warehouse to a data mart or from an OLTP system to a staging area for a data warehouse can be cumbersome and time consuming. Direct path loading through SQL\*Loader or parallel DML makes the task faster, but the process can be simpler for data movement between identical databases. Oracle8i provides a mechanism for copying data files between identical systems and allows the same data to be accessed by both systems. Now data movement can be as fast as a simple transfer of files between machines. This greatly improves performance and provides operational simplicity for transfer of data.

### Corporate Information Systems

A large amount of data may flow across a corporate information system, from an OLTP database to a staging database, then through an enterprise data warehouse to data marts. In this environment, transportable tablespaces can be used for many purposes. Typically, tables in a data warehouse are not the same as those in the OLTP database. A transportable tablespace can be used to move data from an OLTP database to a DSS or warehouse database.

## Information Distribution

This feature can also be used by:

- Companies to publish data on CD ROMs that can be easily integrated into Oracle databases at regional or area offices.

For example, PRODUCT tables with product descriptions and price can be populated and updated at the head office and moved to regional or branch offices for use in an order processing system.

- Content providers to distribute structured data such as trade directories and demographic data to organizations for easy integration into their decision support systems

## Transporting a Tablespace

### Transporting Tablespaces

1. **Make tablespace read-only**
2. **Export metadata from source**
3. **Copy data files to target system**
4. **Transfer export file**
5. **Import metadata into target**
6. **If necessary, alter the tablespace to read-write**

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Steps for Transporting a Tablespace

Use the steps shown below to move a tablespace from one database to another.

- 1 To ensure that no changes are made to the file when it is being moved, make the tablespace read-only.
- 2 Use Export to capture the data dictionary information into an operating system file.
- 3 Make a copy of the tablespace by performing a binary copy of the data files from the source to the target machine.
- 4 Transfer the export file to the target machine.
- 5 Use Import to load the data dictionary information into the target.
- 6 If necessary, alter the tablespace in the source or target database to permit writing.

## Exporting and Importing Metadata

### Exporting and Importing Metadata

```
exp sys/... FILE=s980501.dmp
TRANSPORT_TABLESPACE=y
TABLESPACES=sales_ts
TRIGGERS=N CONSTRAINTS=N
```

```
imp sys/... FILE=s980501.dmp
TRANSPORT_TABLESPACE=y
DATAFILES=( /disk1/sales01.dbf ,
            /disk2/sales02.dbf )
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Exporting Metadata

The use of the `TRANSPORT_TABLESPACE` option specifies that the data dictionary information about all of the objects in the tablespaces must be exported. The new clause `TRIGGERS`, along with `CONSTRAINTS` and `GRANTS`, can be used to control whether associated information is exported when a table is exported.

It is useful to transfer all related data while publishing structured data, while it may not be necessary to copy triggers and constraints if copying data to a data warehouse.

### Importing Metadata

Use the Import utility with the `TRANSPORT_TABLESPACE` option to update the target data dictionary with the tablespace and other object definitions. The following rules govern the import operation:

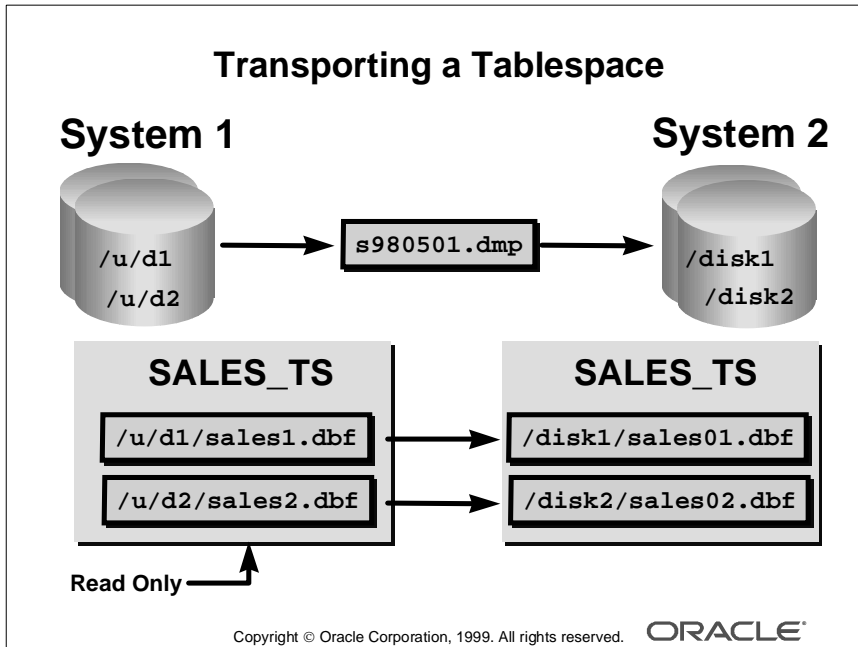
- The `DATAFILES` option must be specified. It names the files belonging to the tablespace that is being transferred. Use the names as they are stored on the target computer, even if they are different from the original filenames.
- If the `TABLESPACES` clause is specified, the supplied tablespace names are compared with those of the export file. Otherwise tablespace names are extracted from the export file.



### **Importing Metadata (continued)**

- The USERS option can be specified to compare the usernames with those of the metadata file. Otherwise the user names are extracted from the export file.
- The FROMUSER and TOUSER options can be used to import objects from a source schema into a target schema.

## Transporting a Tablespace



### Transporting a Tablespace

The slide shows the transportation of tablespace SALES\_TS from System 1 to System 2. The transportation requires that the following steps be taken:

- 1 Put the tablespace into read-only mode in the source database on System 1.  

```
SQL> ALTER TABLESPACE sales_ts READ ONLY;
```
- 2 Export the tablespace metadata from the source database on System 1.  

```
exp FILE=x980501.dmp TRANSPORT TABLESPACE=y \
TABLESPACE=sales_ts TRIGGERS=n CONSTRAINTS=n
```
- 3 Physically copy the tablespace data files from System 1 to System 2. Note that the data filenames are changed during the copy.
- 4 Physically copy the export file, s980501.dmp, from System 1 to System 2.
- 5 Import the tablespace metadata into the target database, using the data filenames that were given on System 2.  

```
imp FILE=x980501.dmp TRANSPORT TABLESPACE=y \
DATAFILES=(/disk1/sales01.dbf,/disk2/sales02.dbf)
```
- 6 The data in tablespace SALES\_TS is available for queries on both systems. The tablespace can be returned to read-write mode on either or both nodes.  

```
SQL> ALTER TABLESPACE sales_ts READ WRITE;
```

## Transportable Tablespace Uses

### Transportable Tablespaces: Uses

- Moves entire tablespace data
- Supports media recovery
- Source and target databases must:
  - Be on the same operating system
  - Run Oracle8i, release 8.1, or above
  - Have the same block size
  - Use the same character set

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Usage

Media recovery is supported through the transport operation. In the event of media damage, you can use a backup performed prior to a tablespace transport operation and recover to a point in time that is after the time the tablespace was transported.

The column PLUGGED\_IN in DBA\_TABLESPACES is set to YES after import. A plugged-in tablespace must be dropped before you downgrade to earlier releases. You can transport the tablespace to another Oracle8i database to preserve data, or use Import and Export.

## Transportable Tablespaces and Schema Objects

### Transportable Tablespaces and Schema Objects

- **Tablespaces transported in one run must be self-contained:**
  - All partitions of a table
  - LOBs must be exported with tables
- **The following objects cannot be transported:**
  - Tables containing nested tables and VARRAYs
  - Bitmap indexes

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

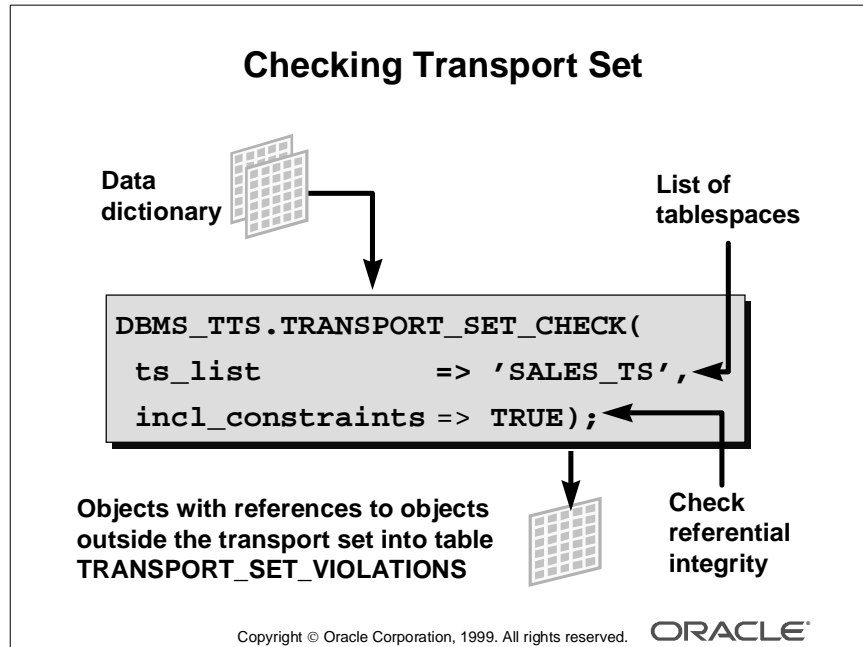
### Self-Contained Sets of Data

The set of tablespaces transported in each run must be self-contained.

The tablespace to be moved can contain tables with LOBs and user-defined data types. If a table with a BFILE column is part of the tablespace that is moved, the user needs to copy the referenced files to the target. Bitmap indexes and tables with VARRAYs or nested tables cannot be transported.

The user is responsible for resolving dependencies between objects in the tablespaces that are transported and those in the target database.

## Checking the Transport Set



### DBMS\_TTS.TRANSPORT\_SET\_CHECK

The PL/SQL procedure `DBMS_TTS.TRANSPORT_SET_CHECK` can be used to verify that a set of tablespaces is self-contained. This procedure accepts two IN arguments:

- A comma-separated list of tablespaces
- A Boolean argument specifying whether to check for referential integrity constraints when determining if the set of tablespaces is self-contained

The procedure populates the view `TRANSPORT_SET_VIOLATIONS`, indicating which objects in the tablespaces specified have relationships to objects outside of the set of tablespaces specified.

The script `dbmsplts.sql`, which is run by `catproc.sql`, creates the `DBMS_TTS` package.

### DBMS\_TTS.ISSELFCONTAINED

The function `DBMS_TTS.ISSELFCONTAINED` returns `TRUE` if the transportable set is self-contained; otherwise, it returns `FALSE`. The function has the same arguments as `TRANSPORT_SET_CHECK`.

## Summary

### Summary

**In this lesson, you should have learned how to:**

- **Use the Export and Import utilities to reorganize data**
- **Move data using transportable tablespaces**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Quick Reference

Context	Reference
Initialization parameters	None
Dynamic performance views	None
Data dictionary views	None
Commands: UNIX  Windows NT	sqlldr or sqlload exp imp EXP.EXE IMP.EXE SQLLDR.EXE
Packaged procedures and functions	DBMS_TTS

## **Managing Password Security and Resources**

## Objectives

### Objectives

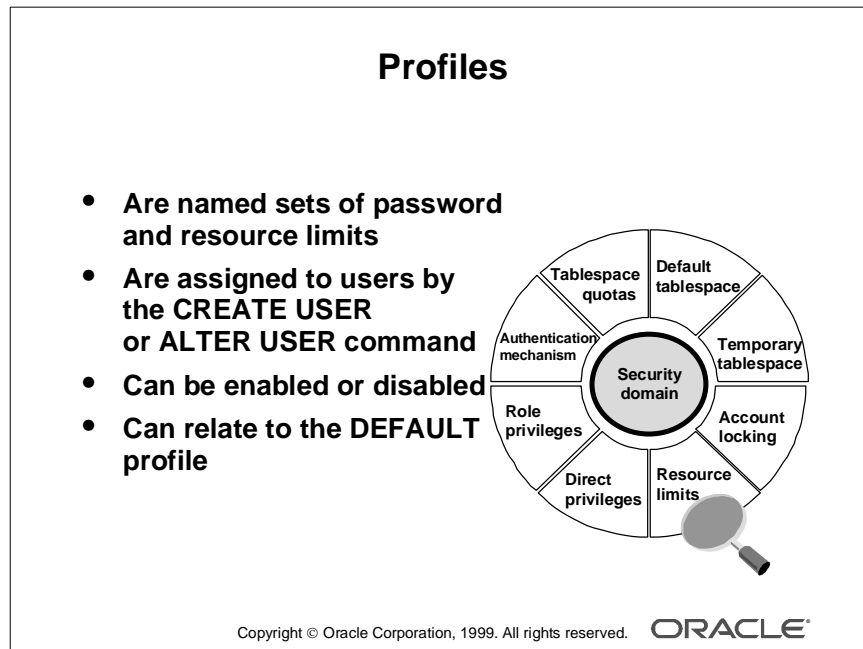
**After completing this lesson, you should be able to do the following:**

- **Manage passwords using profiles**
- **Administer profiles**
- **Control use of resources using profiles**
- **Obtain information about profiles, password management, and resources**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**



## Overview



### What Is a Profile?

A profile is a named set of the following password and resource limits:

- Password aging and expiration
- Password history
- Password complexity verification
- Account locking
- CPU time
- I/O operations
- Idle time
- Connect time
- Memory space (private SQL area for MTS only)
- Concurrent sessions

After a profile has been created, the database administrator can assign it to each user. If resource limits are enabled, the Oracle server limits the database usage and resources to the defined profile of the user.

## **DEFAULT Profile**

The Oracle server automatically creates a DEFAULT profile when the database is created.

The users who have not been explicitly assigned a specific profile conform to all the limits of the DEFAULT profile. All limits of the DEFAULT profile are initially unlimited. However, the database administrator can change the values so that limits are applied to all users by default.

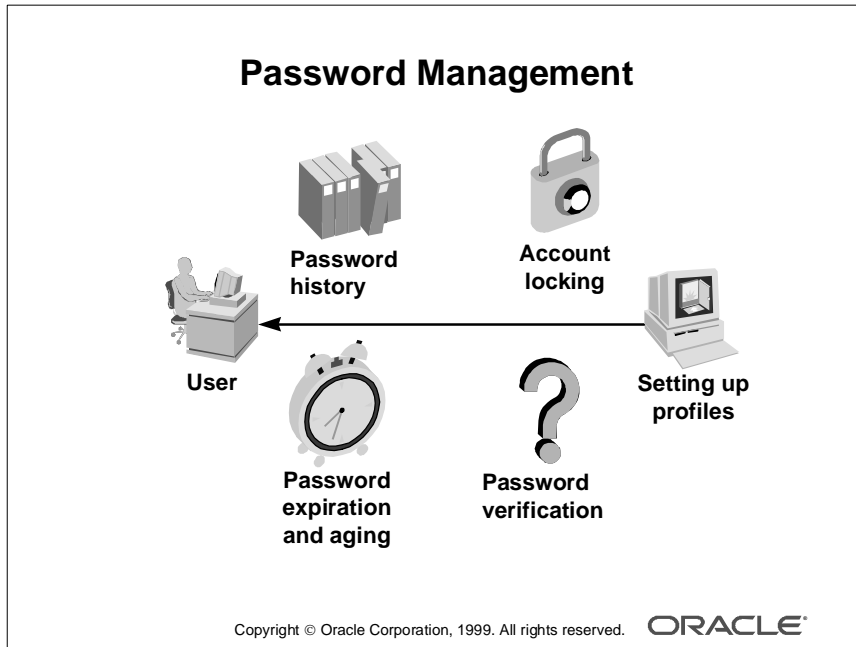
## **Profile Usage**

- Restrict users from performing some operations that require heavy use of resources
- Ensure that users log off the database when they have left their session idle for some time
- Enable group resource limits for similar users
- Easily assign resource limits to users
- Manage resource usage in large, complex multiuser database systems
- Control the use of passwords

## **Profile Characteristics**

- Profile assignments do not affect current sessions.
- Profiles can be assigned only to users and not to roles or other profiles.
- If you do not assign a profile when creating a user, the user is automatically assigned the DEFAULT profile.

## Administering Passwords



### Password Management Features

For greater control over database security, Oracle password management is controlled by database administrators with profiles.

This lesson describes the available password management features:

- **Account locking:** Enables automatic locking of an account when a user fails to log into the system in the specified number of attempts
- **Password aging and expiration:** Enables the password to have a lifetime, after which it expires and must be changed
- **Password history:** Checks the new password to ensure that the password is not reused for a specified amount of time or a specified number of password changes
- **Password complexity verification:** Makes a complexity check on the password to verify that it is complex enough to provide protection against intruders who might try to break into the system by guessing the password

## Enabling Password Management

- Set up password management by using profiles and assign them to users.
- Lock, unlock, and expire accounts using the CREATE USER or ALTER USER command.
- Password limits are always enforced.

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### How to Enable Password Management

Create the profile to limit the password settings, and assign the profile to the user by using the CREATE USER or ALTER USER command.

Password limit settings in profiles are always enforced.

When password management is enabled, the user account can be locked or unlocked by using the CREATE USER or ALTER USER command.

**Note:** The CREATE USER command will be covered in the “Managing Users” lesson.

## Password Account Locking

Parameter	Description
FAILED_LOGIN_ATTEMPTS	Number of failed login attempts before lockout of the account
PASSWORD_LOCK_TIME	Number of days for which the account remains locked upon password expiration



Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Account Locking

The Oracle server automatically locks an account after the `FAILED_LOGIN_ATTEMPTS` value is reached. The account is either automatically unlocked after a specified time (`PASSWORD_LOCK_TIME`) or it must be unlocked by the database administrator using the `ALTER USER` command.

The database account can be explicitly locked with the `ALTER USER` command. When this happens, the account is not automatically unlocked.

**Note:** The `ALTER USER` command will be demonstrated later in this lesson.

## Password Expiration and Aging

Parameter	Description
<b>PASSWORD_LIFE_TIME</b>	Lifetime of the password in days after which the password expires
<b>PASSWORD_GRACE_TIME</b>	Grace period in days for changing the password after the first successful login after the password has expired



Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Password Aging and Expiration

The **PASSWORD\_LIFE\_TIME** parameter sets the maximum lifetime after which the password must be changed.

The database administrator can specify a grace period (**PASSWORD\_GRACE\_TIME**), which begins after the first attempt to log in to the database after password expiration. A warning message is generated every time the user tries to log in until the grace period is over. The user is expected to change the password within the grace period.

If the password is not changed, the account is locked.

The user's account status is changed to **EXPIRED** by explicitly setting the password to be expired. That is, when the user logs in, the account enters the grace period. For example, this is useful when a new account is created.

## Controlling Account Lock and Password

```
ALTER USER hanne  
IDENTIFIED BY rue  
ACCOUNT UNLOCK;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Controlling Account Lock and Password

You can use the ALTER USER command to change the password and control account locking. Some of the situations where this may be useful are:

- To reset the password when a user forgets the password
- To unlock a user's account that has been locked by the system
- To explicitly lock an account
- To manually expire a password (This clause is useful when resetting user passwords.)

### Syntax

Use the following command in these situations:

```
ALTER USER user  
[ IDENTIFIED {BY password | EXTERNALLY } ]  
[ PASSWORD EXPIRE ]  
[ ACCOUNT {LOCK | UNLOCK } ] ;
```

Password changes, expiration, and locks do not affect the current session if the user is already logged on. They will be effective only for subsequent sessions.

### **Syntax (continued)**

When a user account is locked and the user attempts to connect, the following message is displayed:

```
ERROR: ORA-28000: the account is locked
Warning: You are no longer connected to ORACLE.
```

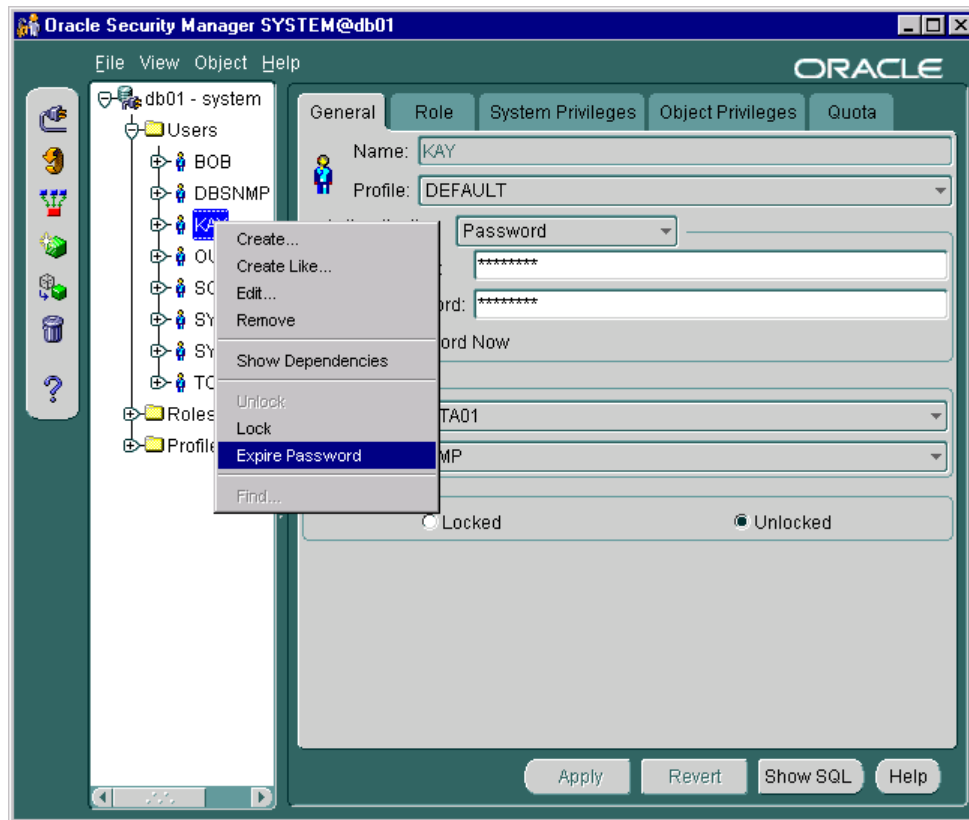


## How to Use Oracle Enterprise Manager to Control Account Lock and Password

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Users folder in the navigator tree.
- 4 Select the username.

If changing account lock or expiring password:

- 5 Select Object—>Change Account Status.
- 6 Choose Unlock, Lock, or Expire Password.

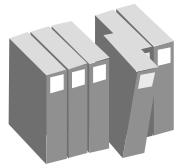


If changing password or mode of authentication:

- 7 Enter the details in the General page of the property sheet.
- 8 Click Apply

## Password History

Parameter	Description
PASSWORD_REUSE_TIME	Number of days before a password can be reused
PASSWORD_REUSE_MAX	Maximum number of times a password can be reused



Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Password History

Password history checks ensure that a user cannot reuse a password for a specified time interval. These checks can be implemented using one of the following:

- **PASSWORD\_REUSE\_TIME** to specify that a user cannot reuse a password for a given number of days
- **PASSWORD\_REUSE\_MAX** to force a user to define a password that is not identical to earlier passwords

When one parameter is set to a value other than **DEFAULT** or **UNLIMITED**, the other parameter must be set to **UNLIMITED**.

## Password Verification

Parameter	Description
<b>PASSWORD_VERIFY_FUNCTION</b>	<b>PL/SQL function that makes a password complexity check before a password is assigned</b>



Copyright © Oracle Corporation, 1999. All rights reserved.

**ORACLE**®

### Password Verification

Before assigning a new password to a user, a PL/SQL function can be invoked to verify the validity of the password.

The Oracle server provides a default verification routine or the database administrator can write a PL/SQL function.

## User-Provided Password Function

Function must be created in the SYS schema and must have the following specification:

```
function_name(  
    userid_parameter IN VARCHAR2(30),  
    password_parameter IN VARCHAR2(30),  
    old_password_parameter IN VARCHAR2(30))  
RETURN BOOLEAN
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### How to Define a Function to Verify a Password

When a new password verification function is added, the database administrator must consider the following restrictions:

- The procedure must use the specification indicated in the slide.
- The procedure returns the value TRUE for success and FALSE for failure.
- If the password function raises an exception, an error is returned and the ALTER USER or CREATE USER command is terminated.
- The password function is owned by SYS.
- If the password function becomes invalid, an error message is returned and the ALTER USER or CREATE USER command is terminated.

**Note:** The CREATE USER command is covered in the “Managing Users” lesson.

## **Password Verification Function VERIFY\_FUNCTION**

- **Minimum length is four characters.**
- **Password should not be equal to username.**
- **Password should have at least one alphabetic, one numeric, and one special character.**
- **Password should differ from the previous password by at least three letters.**



**Password  
verification**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### **The Default Verification Function**

The Oracle server provides a complexity verification function, in the form of a default PL/SQL function called `VERIFY_FUNCTION` of the script `utlpwdmg.sql`, which must be run in the `SYS` schema.

During the execution of the script `utlpwdmg.sql`, the Oracle server creates `VERIFY_FUNCTION` and changes the `DEFAULT` profile with the following `ALTER PROFILE` command:

```
ALTER PROFILE DEFAULT LIMIT
PASSWORD_LIFE_TIME 60
PASSWORD_GRACE_TIME 10
PASSWORD_REUSE_TIME 1800
PASSWORD_REUSE_MAX UNLIMITED
FAILED_LOGIN_ATTEMPTS 3
PASSWORD_LOCK_TIME 1/1440
PASSWORD_VERIFY_FUNCTION verify_function;
```

## Creating a Profile: Password Settings

```
CREATE PROFILE grace_5 LIMIT
  FAILED_LOGIN_ATTEMPTS 3
  PASSWORD_LOCK_TIME UNLIMITED
  PASSWORD_LIFE_TIME 30
  PASSWORD_REUSE_TIME 30
  PASSWORD_VERIFY_FUNCTION verify_function
  PASSWORD_GRACE_TIME 5;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### How to Create a Profile

Use the following CREATE PROFILE command to administer passwords:

```
CREATE PROFILE profile LIMIT
  [FAILED_LOGIN_ATTEMPTS      max_value]
  [PASSWORD_LIFE_TIME         max_value]
  [ {PASSWORD_REUSE_TIME
    |PASSWORD_REUSE_MAX}      max_value]
  [ACCOUNT_LOCK_TIME          max_value]
  [PASSWORD_GRACE_TIME        max_value]
  [PASSWORD_VERIFY_FUNCTION
    {function|NULL|DEFAULT} ]
```

where:     profile                    is the name of the profile to be created

          FAILED\_LOGIN\_ATTEMPTS

                                      specifies the number of failed attempts to

                                      log in to the user account before the account

                                      locked

**How to Create a Profile (continued)****PASSWORD\_LIFE\_TIME**

limits the number of days the same password can be used for authentication. The password expires if it is not changed within this period, and further connections are rejected.

**PASSWORD\_REUSE\_TIME**

specifies the number of days before which a password cannot be reused. If you set **PASSWORD\_REUSE\_TIME** to an integer value, then you must set **PASSWORD\_REUSE\_MAX** to **UNLIMITED**.

**PASSWORD\_REUSE\_MAX**

specifies the number of password changes required before the current password can be reused. If you set **PASSWORD\_REUSE\_MAX** to an integer value, then you must set **PASSWORD\_REUSE\_TIME** to **UNLIMITED**.

**PASSWORD\_LOCK\_TIME**

specifies the number of days an account will be locked after the specified number of consecutive failed login attempts

**PASSWORD\_GRACE\_TIME**

specifies the number of days after the grace period begins during which a warning is issued and login is allowed. If the password is not changed during the grace period, the password expires.

**PASSWORD\_VERIFY\_FUNCTION**

allows a PL/SQL password complexity verification script to be passed as an argument to the **CREATE PROFILE** statement

## How to Use Oracle Enterprise Manager to Create a Profile

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Select the Profiles folder and select Object—>Create.
- 4 Select Profile in the list and click Create.
- 5 Select the Password tab and enter the account password parameters.

The screenshot shows the 'Create Profile' dialog box with the 'Password' tab selected. The dialog has two tabs: 'General' and 'Password'. The 'Password' tab contains several sections with checkboxes and input fields:

- Expire Password:** Checked. 'Expire in:' is set to 30 days. 'Lock:' is set to 5 days past expiration.
- Keep Password History:** Checked. 'Keep:' is set to Unlimited passwords. 'Keep for:' is set to 30 days.
- Enforce Password Complexity:** Checked. 'Complexity function:' is set to Default.
- Lock account on failed logon:** Checked. 'Lock after:' is set to 3 failed logon attempts. 'Lock for:' is set to Unlimited days.

At the bottom of the dialog are four buttons: 'Create', 'Cancel', 'Show SQL', and 'Help'.

- 6 Click Create.

## Assigning a Profile

With the CREATE USER command or the ALTER USER command, a profile can be assigned. Each user can be assigned only one profile at a time.

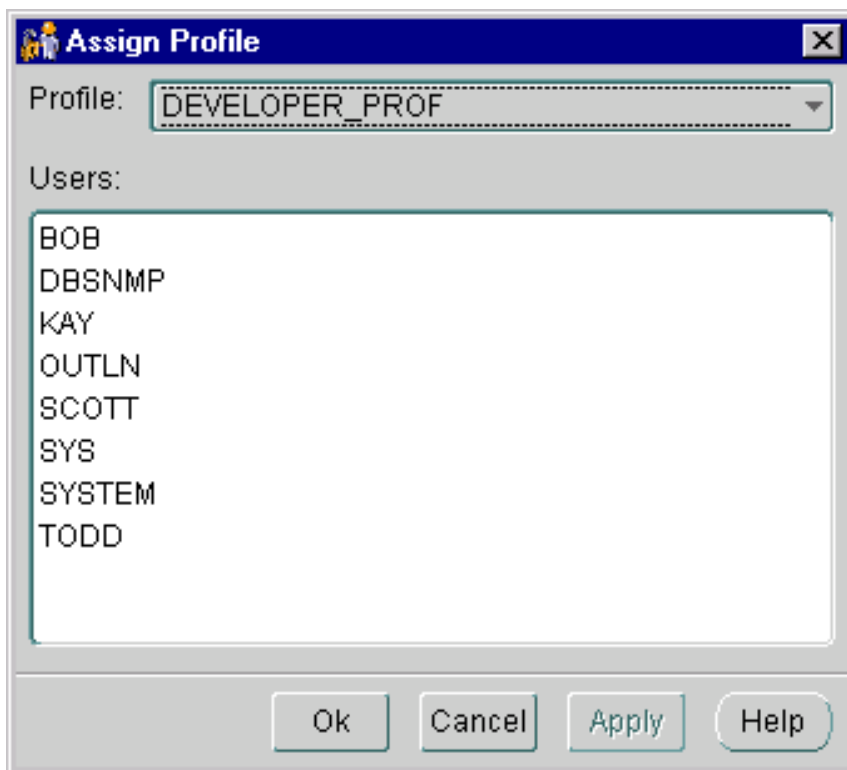
**Note:** The CREATE USER command is covered in the “Managing Users” lesson.



## How to Use Oracle Enterprise Manager to Assign a Profile to a User

To generate the ALTER USER command with Oracle Enterprise Manager use the following steps:

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Profiles folder.
- 4 Select a profile.
- 5 Select Object—>Assign a Profile to User(s).
- 6 In the Assign Profile list, select the user.



- 7 Click Ok.

## Altering and Dropping a Profile

### Altering a Profile

```
ALTER PROFILE default
FAILED_LOGIN_ATTEMPTS 3
PASSWORD_LIFE_TIME 60
PASSWORD_GRACE_TIME 10;
```

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Altering a Profile

Use the ALTER PROFILE command to change the password limits assigned to a profile:

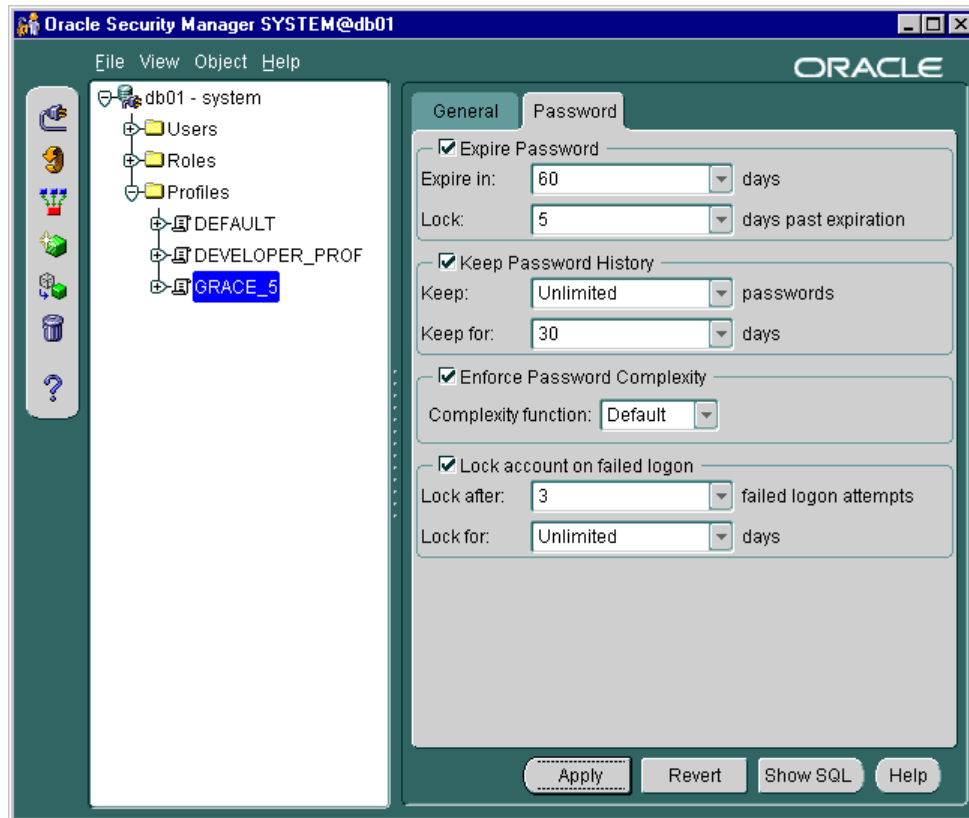
```
ALTER PROFILE profile LIMIT
  [FAILED_LOGIN_ATTEMPTS      max_value]
  [PASSWORD_LIFE_TIME         max_value]
  [ {PASSWORD_REUSE_TIME
    |PASSWORD_REUSE_MAX}      max_value]
  [ACCOUNT_LOCK_TIME          max_value]
  [PASSWORD_GRACE_TIME        max_value]
  [PASSWORD_VERIFY_FUNCTION
    {function|NULL|DEFAULT} ]
```

If you want to set the password parameters to less than a day:

1 hour:        `PASSWORD_LOCK_TIME = 1/24`  
10 minutes    `PASSWORD_LOCK_TIME = 10/1440`  
5 minutes     `PASSWORD_LOCK_TIME = 5/1440`

## How to Use Oracle Enterprise Manager to Alter a Profile

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Profiles folder.
- 4 Select the profile.
- 5 In the Password tab, change the details on the password parameters.



- 6 Click Apply.

## Guidelines

Changes to a profile do not affect current sessions. Changes are used in subsequent sessions only.

## Dropping a Profile

```
DROP PROFILE developer_prof;
```

```
DROP PROFILE developer_prof CASCADE;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Dropping a Profile

Drop a profile using the DROP PROFILE command:

```
DROP PROFILE profile [CASCADE]
```

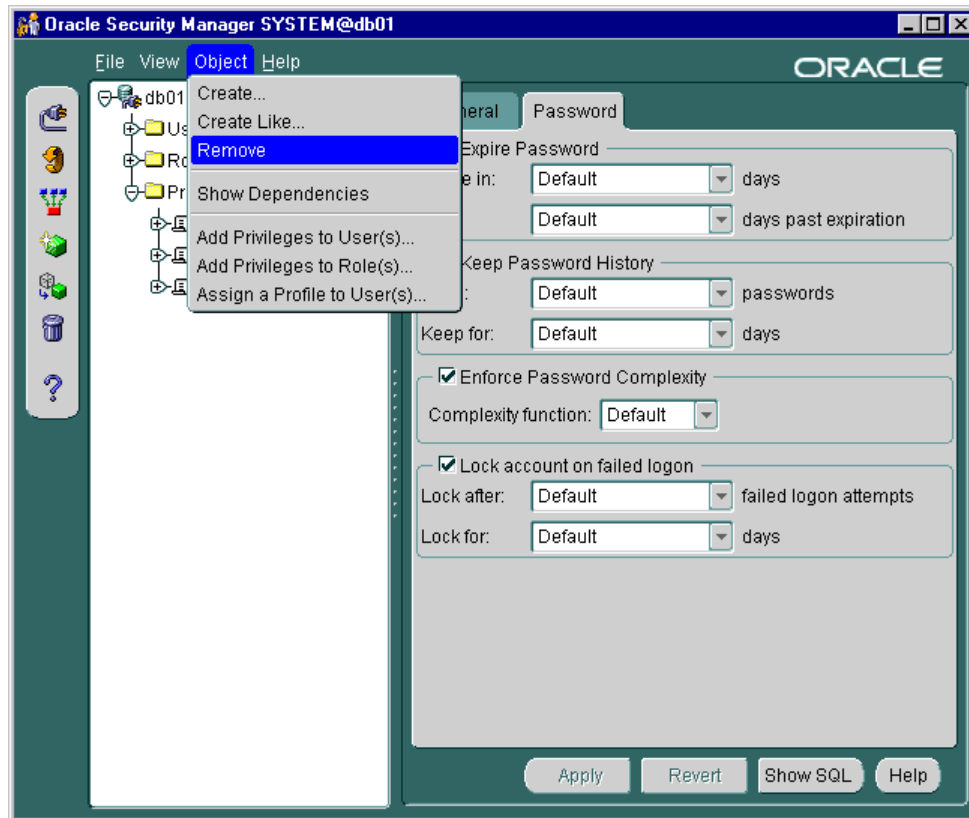
where:	profile	is the name of the profile to be dropped
	CASCADE	revokes the profile from users to whom it is assigned (The Oracle server automatically assigns the DEFAULT profile to such users. Specify this option to drop a profile that is currently assigned to users.)

## Guidelines

- The DEFAULT profile cannot be dropped.
- When a profile is dropped, this change applies to subsequently created sessions only and not to the current sessions.

## How to Use Oracle Enterprise Manager to Drop a Profile

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Profiles folder.
- 4 Select the profile.
- 5 Select Object—>Remove.



- 6 Click OK.

## Controlling Usage of Resources

### Managing Resources with Profiles

1. Create profiles with the **CREATE PROFILE** command
2. Assign profiles to the user with the **CREATE** or **ALTER USER** commands
3. Enable resource limits with the:
  - **RESOURCE\_LIMIT** initialization parameter
  - **ALTER SYSTEM** command

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Steps for Using Resource Limits

Use the following steps to control the usage of resources with profiles:

- 1 Create a profile with the **CREATE PROFILE** command to determine the resource and password limits.
- 2 Assign profiles with the **CREATE USER** or **ALTER USER** command.
- 3 Enforce resource limits with the **ALTER SYSTEM** command or by editing the initialization parameter file (and stopping and restarting the instance).

These steps are discussed in detail in the following section.

**Note:** Enforcing the resource limits is not required for enabling Oracle password management.

### Setting Resource Limits at Session Level

Resource	Description
CPU_PER_SESSION	Total CPU time measured in hundredths of seconds
SESSIONS_PER_USER	Number of concurrent sessions allowed for each username
CONNECT_TIME	Elapsed connect time measured in minutes
IDLE_TIME	Periods of inactive time measured in minutes
LOGICAL_READS_PER_SESSION	Number of data blocks (physical and logical reads)
PRIVATE_SGA	Private space in the SGA measured in bytes (for MTS only)

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Call-Level and Session-Level Limits

Profile limits can be enforced at the session level, the call level, or both. Session-level limits are enforced for each connection.

When a session-level limit is exceeded:

- An error message returns; for example:  
ORA-02391: exceeded simultaneous SESSION\_PER\_USER limit.
- The Oracle server disconnects the user.

Call-level limits are enforced for each call made while executing a SQL statement.

When a call-level limit is exceeded:

- The processing of the statement is halted.
- The statement is rolled back.
- All previous statements remain intact.
- The user's session remains connected.

### Setting Resource Limits at Call Level

Resource	Description
CPU_PER_CALL	CPU time per call in hundredths of seconds
LOGICAL_READS_PER_CALL	Number of data blocks that can be read per call

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

#### Guidelines

- IDLE\_TIME is calculated for the server process only. It does not take into account application activity. The IDLE\_TIME limit is not affected by long-running queries and other operations.
- LOGICAL\_READS\_PER\_SESSION is a limitation on the total number of reads from both memory and disk. This might be done to ensure that no I/O intensive statements can hoard memory and tie up the disk.
- PRIVATE\_SGA applies only when running the multithreaded server (MTS) architecture and can be specified in M or K.

Note: The MTS architecture is discussed in the course *Enterprise DBA Part 3: Network Administration*.



## Creating a Profile: Resource Limit

```
CREATE PROFILE developer_prof LIMIT
SESSIONS_PER_USER 2
CPU_PER_SESSION 10000
IDLE_TIME 60
CONNECT_TIME 480;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### The CREATE PROFILE Command Syntax

Create a profile using the following CREATE PROFILE command:

```
CREATE PROFILE profile LIMIT
[SESSIONS_PER_USER          max_value]
[CPU_PER_SESSION            max_value]
[CPU_PER_CALL               max_value]
[CONNECT_TIME               max_value]
[IDLE_TIME                  max_value]
[LOGICAL_READS_PER_SESSION max_value]
[LOGICAL_READS_PER_CALL     max_value]
[COMPOSITE_LIMIT            max_value]
[PRIVATE_SGA                max_bytes]
```

where:

profile	is the name of the profile
max_value	is an integer, UNLIMITED, or DEFAULT
max_bytes	is an integer optionally followed by K or M, UNLIMITED, or DEFAULT

### The CREATE PROFILE Command Syntax (continued)

UNLIMITED	indicates that a user assigned this profile can use an unlimited amount of this resource
DEFAULT	indicates this profile is subject to the limit for this resource, as specified in the DEFAULT profile
COMPOSITE_LIMIT	limits the total resource cost for a session expressed in service units; Oracle calculates the resource cost as a weighted sum of: CPU_PER_SESSION CONNECT_TIME LOGICAL_READS_PER_SESSION PRIVATE_SGA

**Note:** The data dictionary view RESOURCE\_COST provides the weightages assigned to different resources.

For information on how to specify the weight for each session resource, see the ALTER RESOURCE COST command in the *SQL Reference Guide*.

## How to Use Oracle Enterprise Manager to Create a Profile

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Select the Profiles folder and select Object—>Create.
- 4 Select Profile in the list and click Create.
- 5 Enter the resource parameters.

**Create Profile - system@db01**

General Password

Name: DEVELOPER\_PROF

Details

CPU/Session: 10000 Sec./100

CPU/Call: Default Sec./100

Connect Time: 480 Minutes

Idle Time: 60 Minutes

Database Services

Concurrent Sessions: 2 Per User

Reads/Session: Default Blocks

Reads/Call: Default Blocks

Private SGA: Default KBytes

Composite Limit: Default Service Units

Create Cancel Show SQL Help

- 6 Click Create.

## Enabling Resource Limits

- Set the initialization parameter `RESOURCE_LIMIT` to `TRUE`
- Enforce the resource limits by enabling the parameter with the `ALTER SYSTEM` command

```
ALTER SYSTEM SET RESOURCE_LIMIT=TRUE;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Controlling Enforcement of Resource Limits

Enable or disable the enforcement of resource limits by altering the `RESOURCE_LIMIT` initialization parameter or by using the `ALTER SYSTEM` command.

### `RESOURCE_LIMIT` Initialization Parameter

- To enable or disable enforcement of resource limits, alter this parameter in the initialization file and restart the instance.
- A value of `TRUE` enables enforcement.
- A value of `FALSE` disables enforcement (default).
- Use this parameter to enable enforcement when the database can be shut down.

### `ALTER SYSTEM` Command

- To enable or disable enforcement of resource limits for an instance, use the `ALTER SYSTEM` command.
- The setting specified using the `ALTER SYSTEM` command remains in effect until altered again or until the database is shut down.
- Use this command to enable or to disable enforcement when the database cannot be shut down.

## Viewing Password and Resource Limits Information

### Viewing Password and Resource Limits Information

- **DBA\_USERS**
  - profile
  - username
  - account\_status
  - lock\_date
  - expiry\_date
- **DBA\_PROFILES**
  - profile
  - resource\_name
  - resource\_type (PASSWORD, KERNEL)
  - limit

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE

### Viewing User Information

Use DBA\_USERS to obtain information about expiration and locking dates and the account status.

```
SQL> SELECT username, password, account_status,
2 lock_date, expiry_date
3 FROM dba_users;
```

USERNAME	PASSWORD	ACCOUNT_STATUS	LOCK_DATE	EXPIRY_DA
SYS	8A8F025737A9097A	OPEN		
SYSTEM	D4DF7931AB130E37	OPEN		
BOB	3B2BC8EE81975F69	OPEN		
OUTLN	4A3BA55E08595C81	OPEN		
DBSNMP	E066D214D5421CCC	OPEN		
TODD	BB69FBB77CFA6B9A	OPEN		
KAY	957C7EF29CC223FC	LOCKED	24-JUN-99	
SCOTT	F894844C34402B67	OPEN		10-JUL-99

## Viewing Profile Information

Query the DBA\_PROFILES view to display password profile information:

```
SQL> SELECT * FROM dba_profiles
      2 WHERE resource_type='PASSWORD'
      3 AND profile='GRACE_5';
```

PROFILE	RESOURCE_NAM	RESOURCE	LIMIT
GRACE_5	FAILED_LOGIN_ATTEMPTS	PASSWORD	3
GRACE_5	PASSWORD_LIFE_TIME	PASSWORD	30
GRACE_5	PASSWORD_REUSE_TIME	PASSWORD	30
GRACE_5	PASSWORD_REUSE_MAX	PASSWORD	UNLIMITED
GRACE_5	PASSWORD_VERIFY_FUNCTION	PASSWORD	DEFAULT
GRACE_5	PASSWORD_LOCK_TIME	PASSWORD	UNLIMITED
GRACE_5	PASSWORD_GRACE_TIME	PASSWORD	5

Join the data dictionary views DBA\_USERS and DBA\_PROFILES to display the resource limits for the user SCOTT.

```
SQL> SELECT p.profile, p.resource_name, p.limit
      2 FROM dba_users u, dba_profiles p
      3 WHERE p.profile=u.profile AND
      4 username='SCOTT' AND
      5 p.resource_type='KERNEL';
```

PROFILE	RESOURCE_NAME	LIMIT
DEVELOPER_PROF	COMPOSITE_LIMIT	DEFAULT
DEVELOPER_PROF	SESSIONS_PER_USER	2
DEVELOPER_PROF	CPU_PER_SESSION	10000
DEVELOPER_PROF	CPU_PER_CALL	DEFAULT
DEVELOPER_PROF	LOGICAL_READS_PER_SESSION	DEFAULT
DEVELOPER_PROF	LOGICAL_READS_PER_CALL	DEFAULT
DEVELOPER_PROF	IDLE_TIME	60
DEVELOPER_PROF	CONNECT_TIME	480
DEVELOPER_PROF	PRIVATE_SGA	DEFAULT

---

## Summary

### Summary

In this lesson, you should have learned how to:

- Administer passwords
- Administer profiles
- Administer resource limits

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Quick Reference

Context	Reference
Initialization parameters	RESOURCE_LIMIT
Dynamic performance views	None
Data dictionary views	DBA_PROFILES DBA_USERS
Commands	CREATE PROFILE ALTER PROFILE DROP PROFILE ALTER USER
Stored procedures and functions	VERIFY_FUNCTION





## **Managing Users**

## Objectives

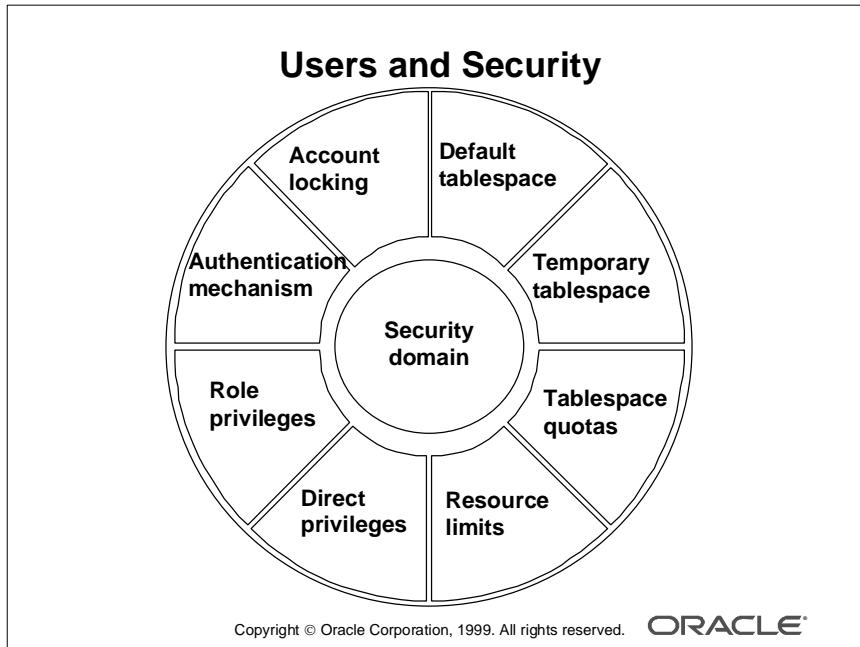
### Objectives

**After completing this lesson, you should be able to do the following:**

- **Create new database users**
- **Alter and drop existing database users**
- **Monitor information about existing users**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Overview



### Security Domain

The database administrator defines the names of the users allowed to access a database. A security domain defines the settings that apply to the user.

### Authentication Mechanism

A user who needs access to the database can be authenticated by one of the following:

- Database
- Operating system
- Network

The means of authentication is specified at the time the user is defined in the database and can be altered later. This lesson covers authentication by database and by operating system only.

### Note

- Refer to OS authentication using roles in the lesson "Getting Started with Oracle."
- Authentication through the network is covered in the course *Enterprise DBA Part 3: Network Administration*.

## Tablespace Quotas

Tablespace quotas control the amount of physical storage space allocated to a user in the tablespaces in the database.

## Default Tablespace

The default tablespace defines the location where segments created by a user are stored if the user does not explicitly specify a tablespace at the time the segment is created.

## Temporary Tablespace

Temporary tablespace defines where extents will be allocated by the Oracle server if the user performs an operation that requires writing sort data to the disk.

## Account Locking

Accounts can be locked to prevent a user from logging on to the database. This can be set to occur automatically, or the database administrator can lock or unlock accounts manually.

## Resource Limits

Limits can be placed on the use of resources such as CPU time, logical I/O, and the number of sessions opened by a user. Resource limits are discussed in the *Oracle Enterprise DBA Part 2: Performance and Tuning* course.

## Direct Privileges

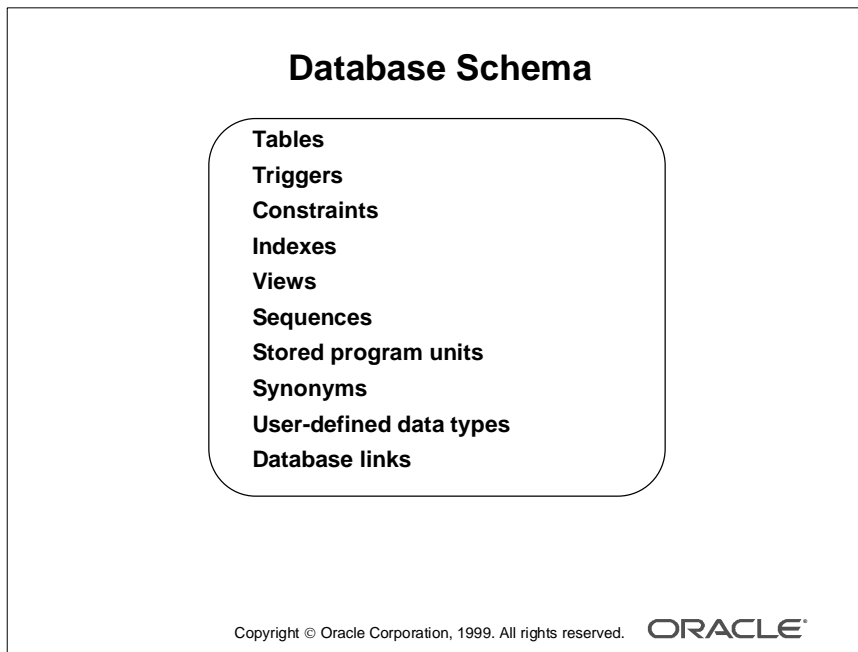
Privileges are used to control the actions a user can perform in a database.

## Role Privileges

A user can be granted privileges indirectly through the use of roles.

Privileges granted directly and through roles are discussed in the lessons “Managing Privileges” and “Managing Roles.”

This lesson covers defining a user with the appropriate authentication mechanism, limiting the use of space by the users in the system, and manually controlling account locking.



## What Is a Schema?

A schema is a named collection of objects such as tables, views, clusters, procedures, and packages associated with a particular user. When a database user is created, a corresponding schema with the same name is created for that user. A user can be associated only with a schema of the same name, and therefore *username* and *schema* are often used interchangeably.

The slide shows some of the objects that users can own in an Oracle database.

## Creating New Database Users

### Checklist for Creating Users

1. Choose a username and authentication mechanism.
2. Identify tablespaces in which the user needs to store objects.
3. Decide on quotas for each tablespace.
4. Assign a default tablespace and temporary tablespace.
5. Create a user.
6. Grant privileges and roles to the user.

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE<sup>®</sup>

## Creating a New User: Database Authentication

Set the initial password:

```
CREATE USER peter
IDENTIFIED BY mylstson
DEFAULT TABLESPACE data
TEMPORARY TABLESPACE temp
QUOTA 15m ON data
PASSWORD EXPIRE;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Syntax

Use the following command to create a new user:

```
CREATE USER user
IDENTIFIED {BY password | EXTERNALLY}
[ DEFAULT TABLESPACE tablespace ]
[ TEMPORARY TABLESPACE tablespace ]
[ QUOTA {integer [K | M ] | UNLIMITED } ON tablespace
  [ QUOTA {integer [K | M ] | UNLIMITED } ON tablespace ] ...]
[ PASSWORD EXPIRE ]
[ ACCOUNT { LOCK | UNLOCK } ]
[ PROFILE { profile | DEFAULT } ]
```

where:	user	is the name of the user
	BY password	specifies that the user is authenticated by the database and needs to supply <i>password</i> while logging on
	EXTERNALLY	specifies that the user is authenticated by the operating system
	GLOBALLY AS external_name	specifies that the user is authenticated globally by the specified external name

## Syntax (continued)

### DEFAULT TEMPORARY TABLESPACE

identifies the default or temporary tablespace for the user

### QUOTA

defines the maximum space allowed for objects owned by the user in the tablespace *tablespace* (Quota can be defined as *integer* bytes or kilobytes and megabytes. The keyword UNLIMITED is used to specify that the objects owned by the user can use as much space as is available in the tablespace. By default, no user has any quota on any tablespace.)

### PASSWORD EXPIRE

forces the user to reset the password when the user logs on to the database using SQL Plus (This option is valid only if the user is authenticated by the database.)

### ACCOUNT LOCK/ UNLOCK

can be used to lock or unlock the user's account explicitly (UNLOCK is the default.)

### PROFILE

is used to control resource usage and to specify the password control mechanism to be used for the user

**Note:** Profiles are discussed in the lesson “Managing Profiles.”

A password authentication method is mandatory. If a password is specified, it is maintained by the Oracle server in the data dictionary. Password control mechanisms provided by the Oracle server are available when users are authenticated by the server.



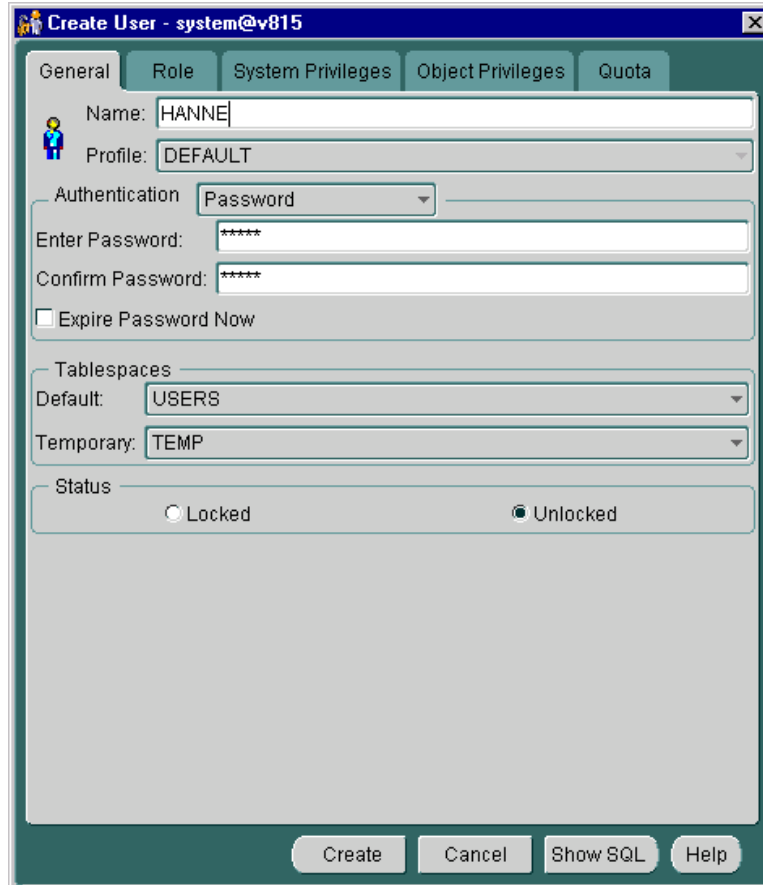
**Syntax (continued)**

Once the password expiry is set, when the user logs on using SQL Plus, the user receives the following message at logon, and is prompted to enter a new password:

```
ERROR:
ORA-28001: the account has expired
Changing password for PETER
Old password:
New password:
Retype new password:
Password changed
```

## How to Use Oracle Enterprise Manager to Create a New User

- 1 Launch Security Manager and connect directly to the database  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Select the Users folder and select Create from the right mouse menu.
- 4 Enter user information in the General page of the property sheet.



- 5 Specify quotas using the Quotas page.
- 6 Click Create.

Select a user and then select Object—>Create Like from the menu bar to create a user with the same quotas and privileges as an existing database user.

**Note:** Oracle Security Manager automatically grants the CONNECT role to any user who is created using the tool. This role is discussed in the “Managing Roles” lesson.

## Creating a New User: Operating System Authentication

Use OS\_AUTHENT\_PREFIX

Example: os User = user15

OS_AUTHENT_PREFIX	Database User	Remote Login Possible
OS_	OS_USER15	No
empty string " "	USER15	No
OPSS\$ (default)	OPSS\$USER15 (default)	Yes

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Operating System Authentication

Use the IDENTIFIED EXTERNALLY clause of the CREATE USER command to specify that a user must be authenticated by the operating system. This option is generally useful when the user logs on directly to the machine where the Oracle server is running.

### Username for Operating System Authentication

The OS\_AUTHENT\_PREFIX initialization parameter is used to specify the format of the usernames for operating system authentication. This value defaults to OPSS\$ to make it backward compatible with earlier releases of the Oracle server. To set the prefix to a NULL value, specify this initialization parameter as:

```
OS_AUTHENT_PREFIX = ''
```

The example in the slide shows how a user, USER15, is defined in the database. This specifies that the operating system user *user15* will be allowed access to the database without having to go through any validation by the Oracle server. Thus, to use SQL Plus to log on to the system UNIX user *user15* needs to enter the following command from the operating system:

```
$ sqlplus /
```

## Username for Operating System Authentication (continued)

### Note

- Using `OS_AUTHENT_PREFIX=OPS$` gives the flexibility of having a user authenticated by either the operating system or the Oracle server. In this case, the DBA can create the user by entering a command of the form:

```
CREATE USER ops$user  
IDENTIFIED BY password ...
```

A user who logs on to the machine running the Oracle server need not supply a password. If the user connects from a remote client, he or she can connect by supplying the password.

- Setting another initialization parameter, `REMOTE_OS_AUTHENT=TRUE`, specifies that a user can be authenticated by a remote operating system. The default value of `FALSE` indicates that a user can be authenticated only by the machine running the Oracle server. Use this parameter with care because there is a potential security problem.
- If there are users in the database who are authenticated by the operating system, changing `OS_AUTHENT_PREFIX` may prevent these users from logging on to the database.

### **Creating a New User: Guidelines**

- **Choose a standard password initially; use OS authentication sparingly.**
- **Use the EXPIRE keyword to force users to reset their passwords.**
- **Always assign a temporary tablespace.**
- **Restrict quotas to few users; use QUOTA UNLIMITED with caution.**
- **Educate users:**
  - **To connect**
  - **To change password**

Copyright © Oracle Corporation, 1999. All rights reserved.

**ORACLE®**

### **Guidelines for Creating a New User**

- After creating the user account, pass the information on to the user.
- Show the user how to connect to the Oracle server and how to change the password.
- The temporary tablespace defaults to SYSTEM and can cause fragmentation of the SYSTEM tablespace, so it is important to specify the temporary tablespace for each user.
- DEFAULT TABLESPACE is a convenience feature. Although this defaults to SYSTEM, a user cannot create any object in this tablespace unless explicitly assigned space.
- A user requires quota on some tablespace to store user objects. Because the temporary segments are created and removed by the Oracle server, users need not have any quota on temporary tablespaces. In a similar vein, a user need not have any quota on tablespaces meant for rollback segments.

## Altering and Dropping Database Users

### Changing User Quota on Tablespace

```
ALTER USER peter  
QUOTA 0 ON data;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

### Modifying Tablespace Quotas

You may need to modify tablespace quotas in the following situations:

- When tables owned by a user exhibit unanticipated growth
- When an application is enhanced and requires additional tables or indexes
- When objects are reorganized and placed in different tablespaces

## Syntax

Use the following command to modify tablespace quotas or to reassign tablespaces:

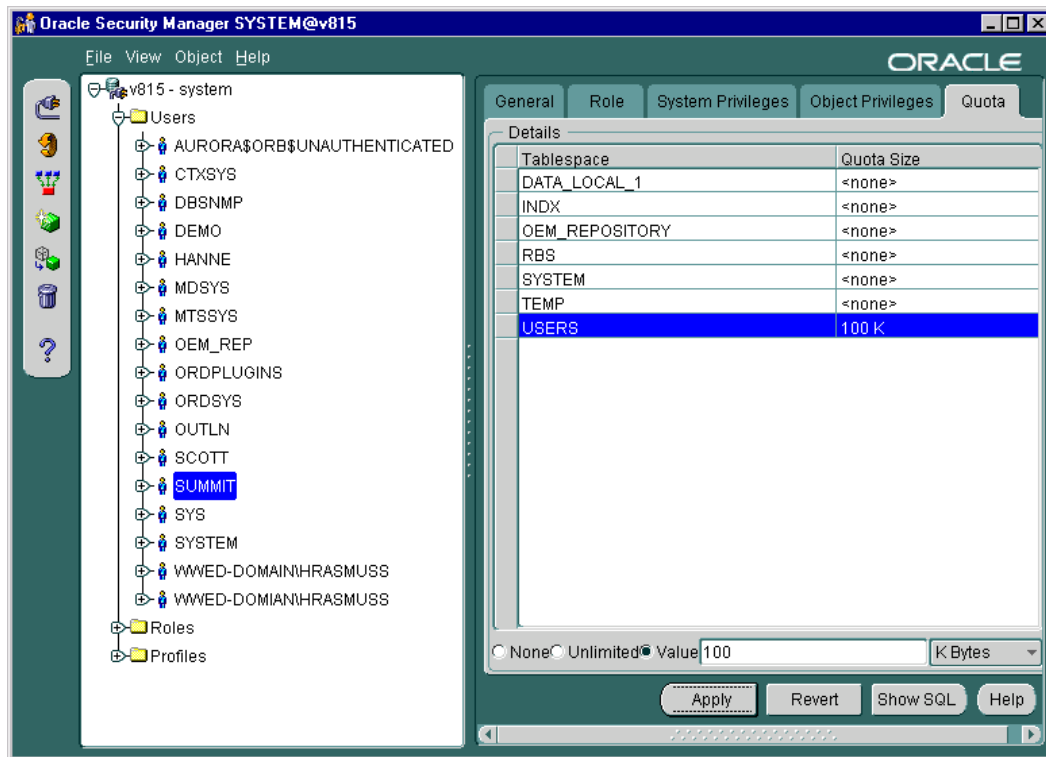
```
ALTER USER user
[ DEFAULT TABLESPACE tablespace]
[ TEMPORARY TABLESPACE tablespace]
[ QUOTA {integer [K | M] | UNLIMITED } ON tablespace
[ QUOTA {integer [K | M] | UNLIMITED } ON tablespace ] ... ]
```

Once a quota of 0 is assigned, the objects owned by the user remain in the revoked tablespace, but they cannot be allocated any new space. For example, if a table that is 10 MB exists in tablespace data01, and the tablespace data01 quota is altered to 0, no more new extents can be allocated for that table.

Any unchanged options remain unchanged.

## How to Use Oracle Enterprise Manager to Modify Tablespace Quota

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Users folder.
- 4 Select the username.
- 5 Enter the details in the Quotas page of the property sheet.



- 6 Click Apply.



## Dropping Users

### Dropping a User

```
DROP USER peter;
```

Use the **CASCADE** clause if the schema contains objects.

```
DROP USER peter CASCADE;
```

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

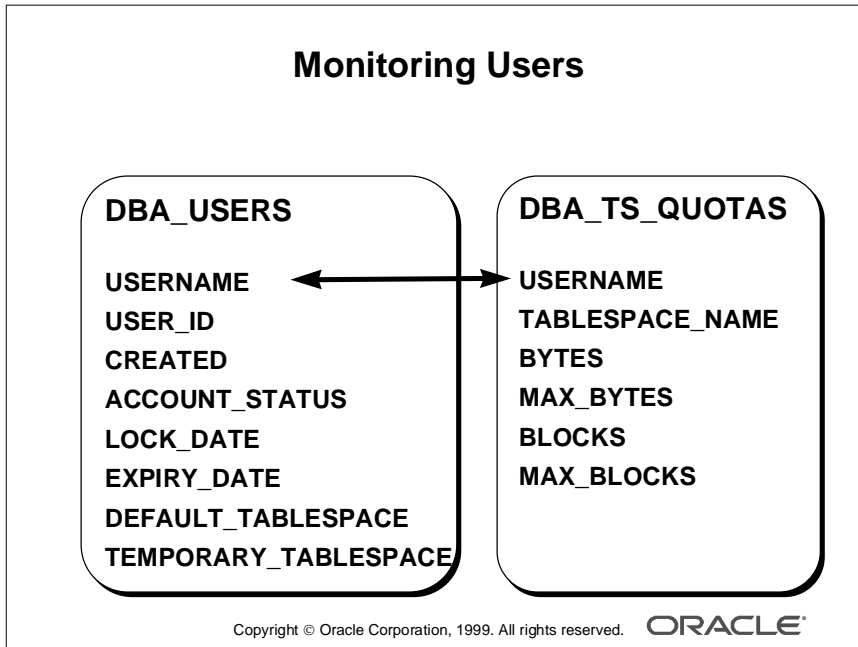
### Syntax

```
DROP USER user [CASCADE]
```

### Guidelines

- The **CASCADE** option drops all objects in the schema before dropping the user. This must be specified if the schema contains any objects.
- A user who is currently connected to the Oracle server cannot be dropped.

## Monitoring Information About Users



### Tablespace Quotas

Use the following query to verify the tablespace quotas for the user SCOTT:

```
SQL> SELECT tablespace_name, blocks, max_blocks, bytes, max_bytes
2  FROM dba_ts_quotas
3  WHERE username = 'SCOTT';
```

TABLESPACE_NAME	BLOCKS	MAX_BLOCKS	BYTES	MAX_BYTES
-----	-----	-----	-----	-----
DATA01	10	-1	20480	-1

1 row selected.

A value of -1 in the MAX\_BLOCKS or the MAX\_BYTES column indicates that the user has unlimited quota on a tablespace.

**User Account Status**

The following query lists all the users, their account status, and temporary tablespaces:

```
SQL> SELECT username, account_status, temporary_tablespace
2  FROM dba_users;
```

USERNAME	ACCOUNT_STATUS	TEMPORARY_TABLESPACE
-----	-----	-----
SYS	OPEN	TEMP
SYSTEM	OPEN	TEMP
DBSNMP	OPEN	TEMP
SCOTT	OPEN	TEMP

4 rows selected.

## Summary

### Summary

In this lesson, you should have learned how to:

- Create users specifying the appropriate password mechanism
- Control usage of space by users

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Quick Reference

Context	Reference
Initialization parameters	OS_AUTHENT_PREFIX REMOTE_OS_AUTHENT
Data dictionary views	DBA_USERS DBA_TS_QUOTAS
Commands	CREATE USER ALTER USER DROP USER

## **Managing Privileges**

## Objectives

### Objectives

**After completing this lesson, you should be able to do the following:**

- **Identify system and object privileges**
- **Grant and revoke privileges**
- **Control operating system or password file authentication**
- **Identify auditing capabilities**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Overview

### Managing Privileges

**Two types of privileges:**

- **System:** Enables users to perform particular actions in the database
- **Object:** Enables users to access and manipulate a specific object

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### System Privileges

Each system privilege enables a user to perform a particular database operation or class of database operations. These operations include creating, dropping, and altering tables, views, rollback segments, and procedures.

### Object Privileges

Each object privilege enables a user to perform a particular action on a specific object, such as a table, view, sequence, procedure, function, or package.

## System Privileges

### System Privileges

- There are about 126 system privileges.
- The **ANY** keyword in the privileges signifies that users have the privilege in every schema.
- The **GRANT** command adds a privilege to a user or a group of users.
- The **REVOKE** command deletes the privileges.

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

### What Are System Privileges?

- There are approximately 126 system privileges, and the number continues to grow.
- The privileges can be classified as follows:
  - Privileges enabling systemwide operations; for example, **CREATE SESSION**, **CREATE TABLESPACE**
  - Privileges enabling management of objects in a user's *own* schema; for example, **CREATE TABLE**
  - Privileges enabling management of objects in *any* schema; for example, **CREATE ANY TABLE**
- Privileges can be controlled with the DDL commands **GRANT** and **REVOKE**, which add and revoke system privileges to the user or to a role (see the lesson "Maintaining Roles").

**Note:** Users with ANY privileges can access dictionary tables except those with the prefix **USER\_**, **ALL**, and any views on which privileges have been granted to **PUBLIC**.



### System Privileges: Examples

Category	Examples
INDEX	CREATE ANY INDEX ALTER ANY INDEX DROP ANY INDEX
TABLE	CREATE TABLE CREATE ANY TABLE ALTER ANY TABLE DROP ANY TABLE SELECT ANY TABLE UPDATE ANY TABLE DELETE ANY TABLE
SESSION	CREATE SESSION ALTER SESSION RESTRICTED SESSION
TABLESPACE	CREATE TABLESPACE ALTER TABLESPACE DROP TABLESPACE UNLIMITED TABLESPACE

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### System Privileges: Examples

- There is no CREATE INDEX privilege.
- Privileges such as CREATE TABLE, CREATE PROCEDURE, or CREATE CLUSTER include the dropping of these objects.
- CREATE TABLE includes the CREATE INDEX and the ANALYZE commands. The user must have a quota for the tablespace or must have been granted UNLIMITED TABLESPACE.
- UNLIMITED TABLESPACE cannot be granted to a role.
- For truncating a table, the DROP ANY TABLE privilege is necessary.

**Note:** For the complete list, see the “Managing User Privileges” chapter in *Oracle Server Administrator’s Guide Release 8.1*, or query the SYSTEM\_PRIVILEGE\_MAP view.

## Granting System Privileges

### Granting System Privileges

```
GRANT CREATE SESSION, CREATE TABLE TO managers;
```

```
GRANT CREATE SESSION TO scott  
WITH ADMIN OPTION;
```

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Syntax

Use the following command to grant a system privilege:

```
GRANT {system_priv|role}  
    [, {system_priv|role} ]...  
TO    {user|role|PUBLIC}  
    [, {user|role|PUBLIC} ]...  
[WITH ADMIN OPTION]
```

where:

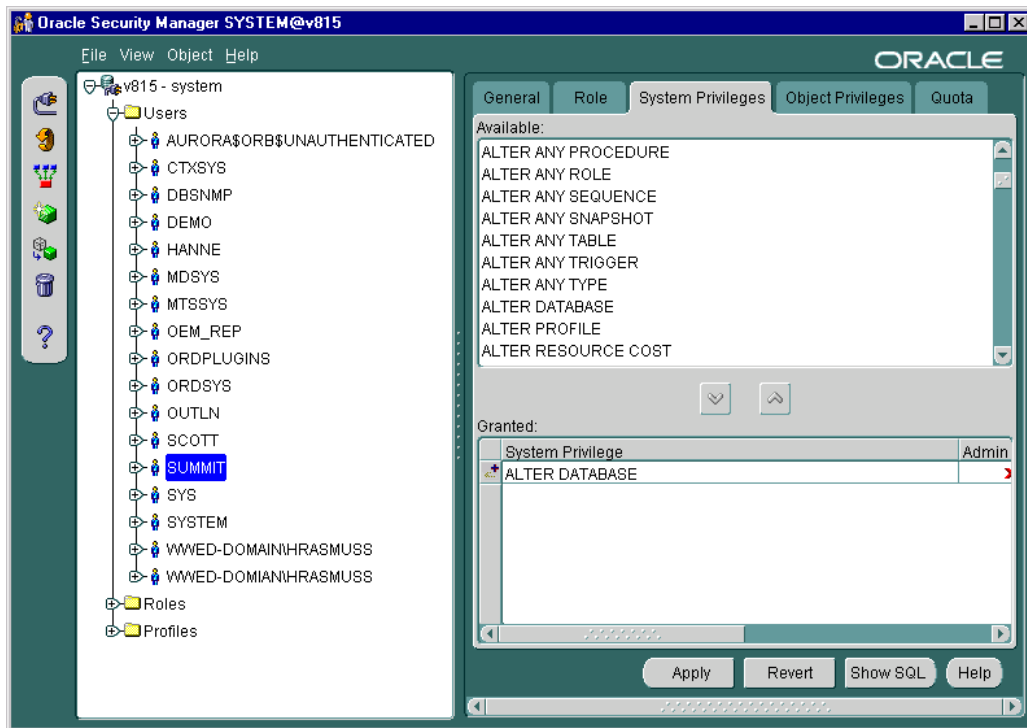
system_priv	specifies the system privilege to be granted
role	specifies the role name to be granted
PUBLIC	grants system privilege to all users
WITH ADMIN OPTION	enables the grantee to further grant the privilege or role to other users or roles

### **Guidelines**

- To grant a system privilege, you must have been granted the privilege **WITH ADMIN OPTION**.
- The grantee with the **ADMIN OPTION** can further grant the system privilege or role with the **ADMIN OPTION**.
- Any user with the **GRANT ANY ROLE** system privilege can grant any role in a database.
- The grantee with the **ADMIN OPTION** can grant the system privilege to or revoke the system privilege from any user or role in the database.

## How to Use Oracle Enterprise Manager to Grant System Privileges

- 1 Launch Security Manager and connect directly to the database  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Select the username or role to which you want to grant the privilege.
- 4 Select Tab: System Privileges.
- 5 Select the system privilege you want to grant.
- 6 Optionally, select the WITH ADMIN OPTION Box.



- 7 Click Apply.

## Password File Authentication

### **SYSDBA and SYSOPER Privileges**

Category	Examples
<b>SYSOPER</b>	STARTUP SHUTDOWN ALTER DATABASE OPEN   MOUNT ALTER DATABASE BACKUP CONTROLFILE ALTER TABLESPACE BEGIN/END BACKUP RECOVER DATABASE ALTER DATABASE ARCHIVELOG RESTRICTED SESSION
<b>SYSDBA</b>	SYSOPER privileges WITH ADMIN OPTION CREATE DATABASE RECOVER DATABASE UNTIL

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### **SYSDBA and SYSOPER Privileges**

In the lesson “Getting Started with Oracle,” the system privileges SYSDBA and SYSOPER were introduced to specify the authentication by using a password file.

Only database administrators should have the capability to connect to a database with administrator privileges. Connecting as SYSDBA gives a user unrestricted privileges to perform any operation on a database or the objects within a database.

## Password File Authentication

1. Check that the password file has been created; if not, create it using **ORAPWD**.
2. Check that the initialization parameter **REMOTE\_LOGIN\_PASSWORD\_FILE** has been set to **EXCLUSIVE**.
3. Grant **SYSOPER** and **SYSDBA** privileges to users.
4. Query **V\$PWFIL\_USERS** to verify the password file members.

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## How to Enable Password File Authentication

Check that the password file has been created; if not, use the password utility **ORAPWD** to create it. Check that the initialization parameter **REMOTE\_LOGIN\_PASSWORD\_FILE** has been set to **EXCLUSIVE**.

The database administrator can add users to the password file by granting the **SYSOPER** or **SYSDBA** system privileges. The **WITH ADMIN OPTION** cannot be used for granting these privileges. Only users currently connected as **SYSDBA** can grant or revoke **SYSDBA** or **SYSOPER** system privileges to another user. These privileges cannot be granted to roles, because a role is not available before a database startup.

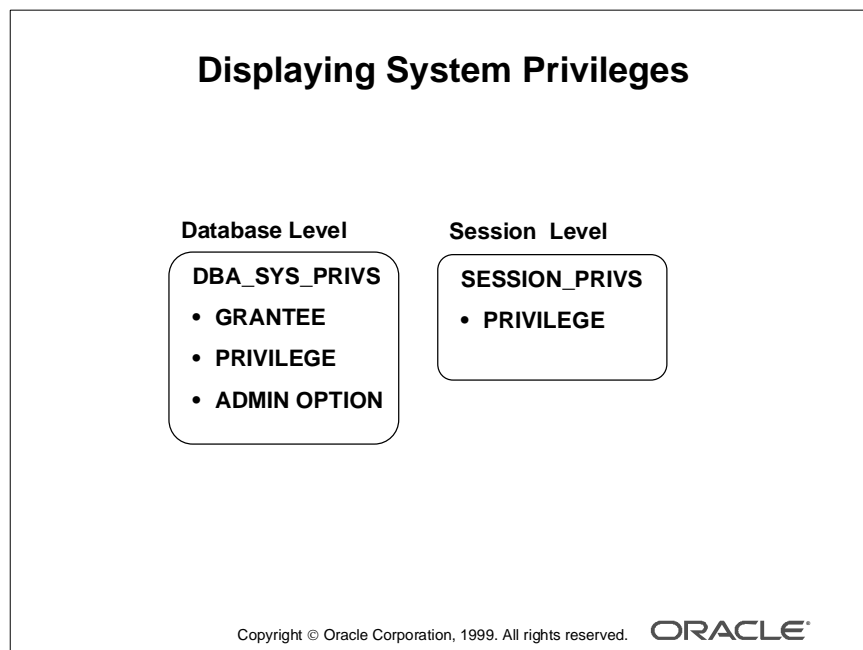
View **V\$PWFIL\_USERS** to display users who have been granted **SYSDBA** or **SYSOPER** privileges.

```
SQL> SELECT * FROM v$pwfile_users;
```

USERNAME	SYSDB	SYSOP
-----	-----	-----
INTERNAL	TRUE	TRUE
SYS	TRUE	TRUE

2 rows selected.

## Displaying System Privileges



## Querying System Privileges

Query DBA\_SYS\_PRIVS to list system privileges granted to users and roles.

```
SQL> SELECT * FROM DBA_SYS_PRIVS;
```

GRANTEE	PRIVILEGE	ADM
-----	-----	-----
...		
SUMMIT	SELECT ANY TABLE	NO
SYS	DELETE ANY TABLE	NO
SYS	EXECUTE ANY TYPE	NO
SYS	INSERT ANY TABLE	NO
SYS	SELECT ANY SEQUENCE	NO
SYS	SELECT ANY TABLE	YES
SYS	UPDATE ANY TABLE	NO
SYSTEM	UNLIMITED TABLESPAC	YES
...		

### Querying System Privileges (continued)

The `SESSION_PRIVS` view lists the privileges that are available for the current session to a user—in the example, for the user `SUMMIT`.

```
SQL> SELECT * FROM session_privs;
```

```
PRIVILEGE
```

```
-----
```

```
CREATE SESSION
```

```
ALTER SESSION
```

```
CREATE TABLE
```

```
SELECT ANY TABLE
```

```
CREATE CLUSTER
```

```
CREATE SYNONYM
```

```
CREATE VIEW
```

```
CREATE SEQUENCE
```

```
CREATE DATABASE LINK
```

```
CREATE PROCEDURE
```

```
CREATE TRIGGER
```

```
CREATE TYPE
```

```
12 rows selected.
```

**Note:** The `DBA_SYS_PRIVS` view shows all system privileges granted to roles and users at the database level, whereas `SESSION_PRIVS` shows the current privileges for the session, both from the privilege granted directly and from enabled roles (see the “Managing Roles” lesson).



## System Privilege Restrictions

**O7\_DICTIONARY\_ACCESSIBILITY = TRUE**

- Reverts to Oracle7 behavior
- Removes the restrictions on system privileges with the ANY keyword
- Defaults to TRUE

Copyright © Oracle Corporation, 1999. All rights reserved.

**ORACLE®**

## How to Prevent Unauthorized Access to the Dictionary

The dictionary protection mechanism in Oracle8 prevents unauthorized users from accessing dictionary objects.

Access to dictionary objects is restricted to the users with the system privileges SYSDBA and SYSOPER.

System privileges providing access to objects in other schemas do not give access to dictionary objects. For example, the SELECT ANY TABLE privilege enables access to views and tables in other schemas, but it does not enable you to select dictionary objects.

If the parameter is set to TRUE, which is the default, access to objects in the SYS schema is enabled (Oracle7 behavior).

If this parameter is set to FALSE, system privileges that allow access to objects in other schemas do not allow access to objects in the dictionary schema.

For example, if O7\_DICTIONARY\_ACCESSIBILITY=FALSE, then the SELECT ANY TABLE statement enables access to views or tables in any schema except the SYS schema. The system privilege EXECUTE ANY PROCEDURE enables access on the procedures in any other schema except in the SYS schema.

## Revoking System Privileges

### Revoking System Privileges

```
REVOKE CREATE TABLE FROM karen;
```

```
REVOKE CREATE SESSION FROM scott;
```

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

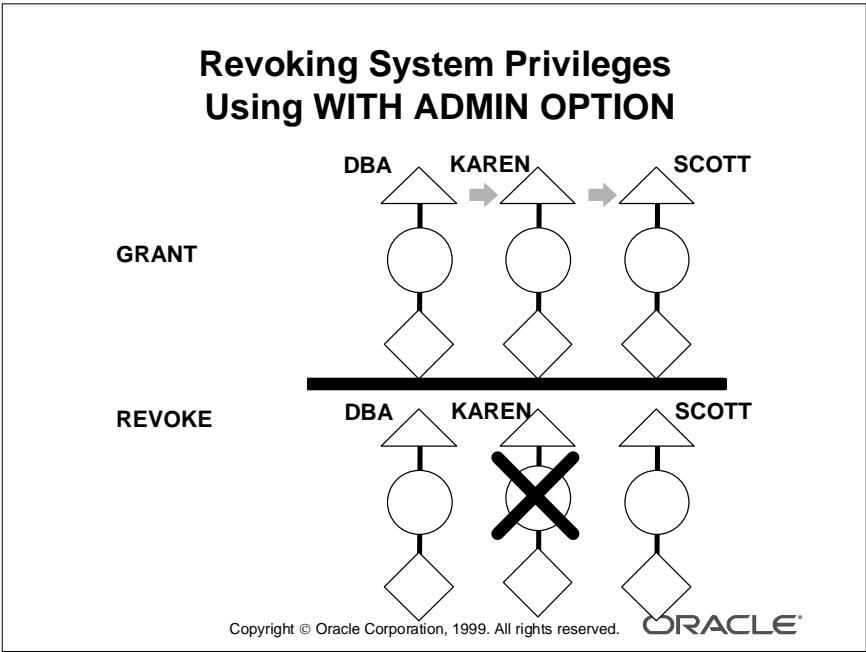
### Syntax

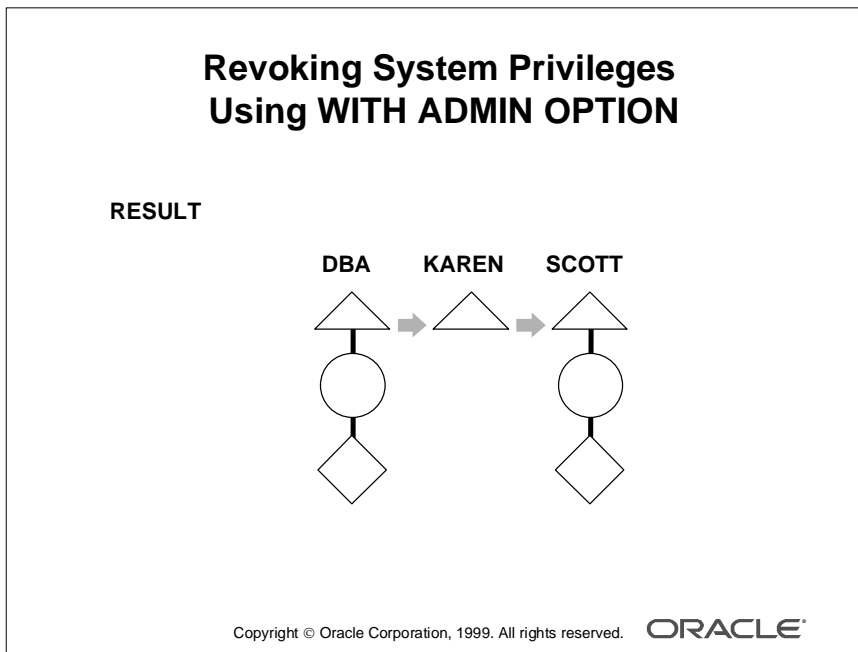
Use the following command to revoke a system privilege:

```
REVOKE {system_priv|role}
      [, {system_priv|role} ]...
FROM   {user|role|PUBLIC}
      [, {user|role|PUBLIC} ]...
```

### Note

- The REVOKE command can only revoke privileges that have been granted directly with a GRANT command.
- Revoking system privileges may have an effect on some dependent objects. For example, if SELECT ANY TABLE is granted to a user and that user has granted any procedures or views that use a table in some other schema, revoking the privilege invalidates the procedures or views.





### Revoking System Privileges Using WITH ADMIN OPTION

There are no cascading effects when a system privilege is revoked, regardless of whether it was given using WITH ADMIN OPTION.

The following scenario illustrates this:

#### Scenario

- 1 The DBA grants the CREATE TABLE system privilege to KAREN with the ADMIN OPTION.
- 2 KAREN creates a table.
- 3 KAREN grants the CREATE TABLE system privilege to SCOTT.
- 4 SCOTT creates a table.
- 5 The database administrator revokes the CREATE TABLE system privilege from KAREN.

#### Results

- The table of KAREN still exists but the user cannot create any new tables.
- SCOTT still has the table and the CREATE TABLE system privilege.

**Note:** In step 3 above, if KAREN grants privileges WITH ADMIN OPTION to SCOTT, SCOTT could potentially revoke the privilege from KAREN.

## Object Privileges

### Object Privileges

Object priv.	Table	View	Sequence	Procedure
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INDEX	√			
INSERT	√	√		
REFERENCES	√			
SELECT	√	√	√	
UPDATE	√	√		

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE<sup>®</sup>

### What Are Object Privileges?

Each object privilege that is granted authorizes the grantee to perform some operation on the object.

The table above summarizes the object privileges that can be granted on each type of object.

## Granting Object Privileges

### Granting Object Privileges

```
GRANT EXECUTE ON dbms_pipe TO public;
```

```
GRANT UPDATE(first_name, salary) ON  
employee TO karen WITH GRANT OPTION;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

### Syntax

Use the following command to grant an object privilege:

```
GRANT {    object_priv [(column_list)]  
          [, object_priv [(column_list)] ]...  
          |ALL [PRIVILEGES]}  
ON      [schema.]object  
TO      {user|role|PUBLIC}  
        [, {user|role|PUBLIC} ]...  
[WITH GRANT OPTION]
```

where:

object_priv	specifies the object privilege to be granted
column_list	specifies a table or view column (This can be specified only when granting the INSERT, REFERENCES, or UPDATE privilege.)
ALL	grants all privileges for the object that have been granted WITH GRANT OPTION

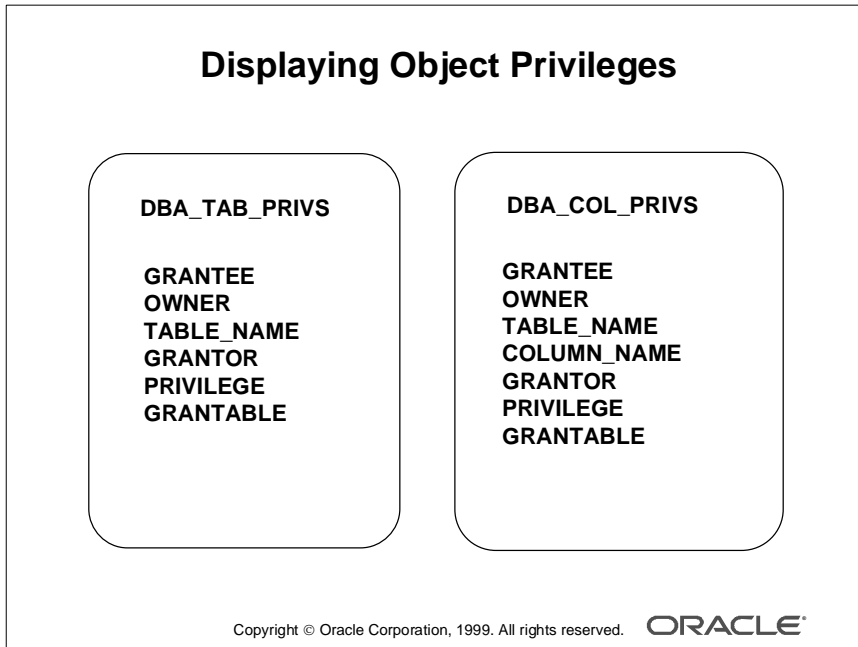
**Syntax (continued)**

ON object	identifies the object on which the privileges are to be granted
WITH GRANT OPTION	enables the grantee to grant the object privileges to other users or roles

**Guidelines**

- To grant privileges, the object must be in your schema or you must have been given the privilege WITH GRANT OPTION.
- By default, if you own an object, all privileges on that object are automatically acquired.
- Use caution when granting privileges on your objects to other users when security is a concern.
- The option WITH GRANT OPTION cannot be used for granting privileges to roles.

## Displaying Object Privileges



### Query Object Privileges

Query DBA\_TAB\_PRIVS to return all object privileges granted to the specified user.

```
SQL> SELECT * FROM dba_tab_privs
2 WHERE GRANTEE='SUMMIT'
```

GRANTEE	OWNER	TABLE_NAME	GRA	PRIVILEGE	GRA
-----	-----	-----	-----	-----	-----
SUMMIT	SYS	RESUMES	SYS	READ	NO

1 row selected.

To list all the column specific privileges that have been granted, use the following query:

```
SQL> SELECT * FROM dba_col_privs;
```

GRANTEE	OWNER	TABLE_NAME	COLUMN_NAME	GRANTOR	PRIVILEGE	GRA
-----	-----	-----	-----	-----	-----	---
SYSTEM	SUMMIT	EMPLOYEE	ID	SUMMIT	INSERT	NO
SYSTEM	SUMMIT	EMPLOYEE	SALARY	SUMMIT	UPDATE	NO

2 rows selected.



## Revoking Object Privileges

### Revoking Object Privileges

```
REVOKE execute ON dbms_pipe FROM scott;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE

### Syntax

Use the following command to revoke an object privilege:

```
REVOKE { object_priv
        [, object_priv ]...
        | ALL [PRIVILEGES] }
ON      [schema.]object
FROM    {user|role|PUBLIC}
        [, {user|role|PUBLIC} ]...
[CASCADE CONSTRAINTS]
```

where:

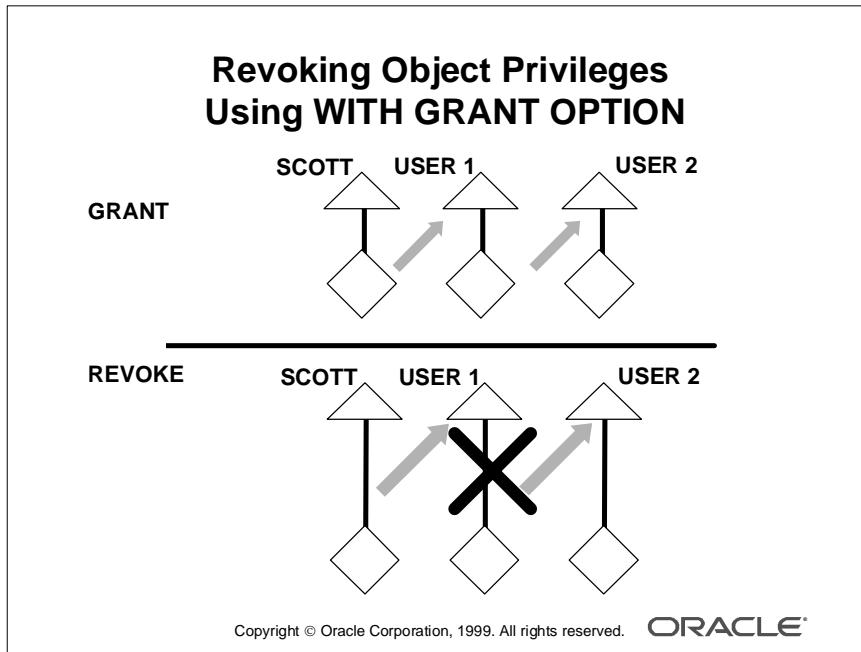
object_priv	specifies the object privilege to be granted
ALL	revokes all object privileges that are granted to the user
ON	identifies the object on which the object privileges are revoked

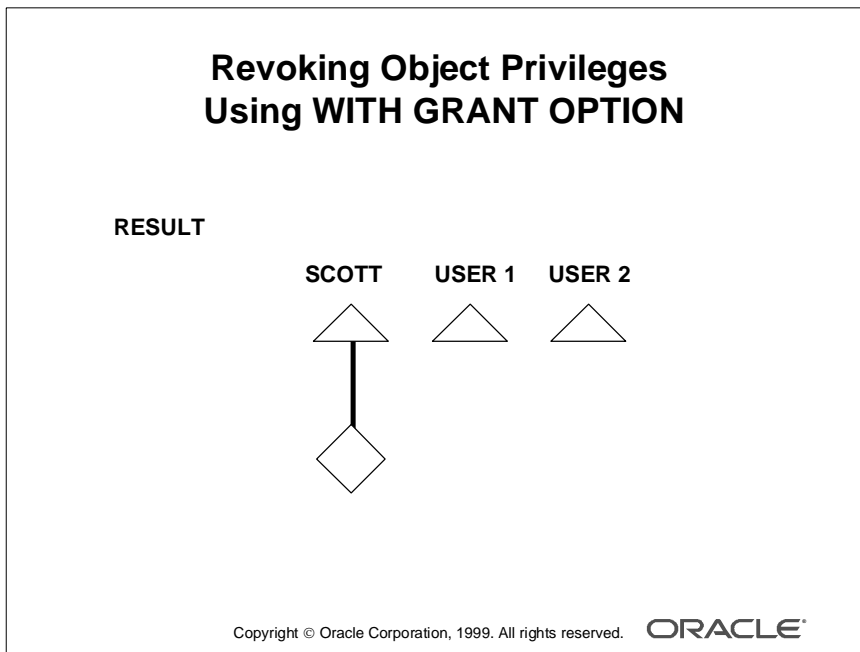
### **Syntax (continued)**

FROM	identifies users or roles from which the object privileges are revoked
CASCADE CONSTRAINTS	drops any referential integrity constraints that the revoke has defined using REFERENCES or ALL privileges

### **Restriction**

Grantors can revoke privileges from only those users to whom they have granted privileges.





### Revoking Object Privileges Using WITH GRANT OPTION

Revoking object privileges will cascade when given using WITH GRANT OPTION.  
The following scenario illustrates this:

#### Scenario

- 1 USER 1 is granted the SELECT object privilege with the GRANT OPTION.
- 2 USER 1 grants the SELECT privilege on EMP to USER 2.

#### Results

Later, the SELECT privilege is revoked from USER 1. This revoke is cascaded to USER 2 as well.

## Auditing Guidelines

### Auditing Guidelines

- **Define your purpose of auditing**
  - **Suspicious database activity**
  - **Gather historical information**
- **Define what you want to audit**
  - **Audit users, statements, or objects**
  - **By session**
  - **Successful or unsuccessful**
- **Manage your audit trail**
  - **Monitor the growth of the audit trail**
  - **Protect the audit trail from unauthorized access**

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE

### Focusing Your Auditing

Restrict auditing by first identifying the auditing requirements, and setting minimal auditing options that will cater to the requirements. Object auditing must be used where possible to reduce the number of entries generated. If statement or privilege auditing needs to be used, the following settings can minimize audit generation:

- Specifying users to audit
- Auditing by session, and not by access
- Auditing either successes or failures, but not both

Audit records may be written to either SYS.AUD\$ or the operating system's audit trail. The ability to use the operating system's audit trail is operating system-dependent.

### Monitoring the Growth of the Audit Trail

If the audit trail becomes full, no more audit records can be inserted, and audited statements will not execute successfully. Errors are returned to all users that issue an audited statement. You must free some space in the audit trail before these statements can be executed.

## Monitoring the Growth of the Audit Trail (continued)

To ensure the audit trail does not grow too rapidly:

- Enable auditing only when necessary.
- Be selective about which audit options are specified.
- Tightly control schema object auditing. Users can turn on auditing for the objects they own.
- The AUDIT ANY privilege also enables a user to turn on auditing, so grant it sparingly.

Periodically remove audit records from the audit trail with the DELETE or TRUNCATE command.

## Protecting the Audit Trail

You should protect the audit trail so that audit information cannot be added, modified, or deleted. Issue the command:

```
AUDIT delete ON sys.aud$ BY ACCESS;
```

To protect the audit trail from unauthorized deletions, only the DBA should have the DELETE\_CATALOG\_ROLE role.

## Moving the Audit Trail out of the System Tablespace

As new records get inserted into the database audit trail, the AUD\$ table can grow without bound. Although you should not drop the AUD\$ table, you can delete or truncate from it because the rows are for information only and are not necessary for the Oracle instance to run. Because the AUD\$ table grows and then shrinks, it should be stored outside of the system tablespace.

To move AUD\$ to the AUDIT\_TAB tablespace:

**1** Ensure that auditing is currently disabled.

**2** Enter the following command

```
ALTER TABLE aud$  
MOVE TABLESPACE AUDIT_TAB;
```

**3** Enter the following command

```
CREATE INDEX i_aud1 ON aud$(sessionid, ses$tid)  
TABLESPACE AUDIT_IDX;
```

**4** Enable auditing for the instance.

## Auditing Categories

- **Auditing privileged operations**
  - Always audited
  - Startup, shutdown, and SYSDBA connections
- **Database auditing**
  - Enabled by DBA
  - Cannot record column values
- **Value-based or application auditing**
  - Implemented through code
  - Can record column values
  - Used to track changes to tables

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Auditing of Privileged Operations

The Oracle server will always audit the following database-related actions into the system audit trail:

- **Instance startup:** An audit record is generated that details the OS user starting the instance, terminal identifier, the date and time stamp, and whether database auditing was enabled or disabled.
- **Instance shutdown:** An audit record is generated that details the OS user shutting down the instance, terminal identifier, the date and time stamp.
- **Connections to the database with administrator privileges:** An audit record is generated that details the OS user connecting to the Oracle server as SYSOPER or SYSDBA to provide accountability of users with administrator privileges.

## Database Auditing

Database auditing is the monitoring and recording of selected user database actions. Information about the event is stored in the audit trail.

The audit trail can be used to investigate suspicious activity. For example, if an unauthorized user is deleting data from tables, the database administrator may decide to audit all connections to the database in conjunction with successful and unsuccessful deletions of rows from tables in the database.

### **Database Auditing (continued)**

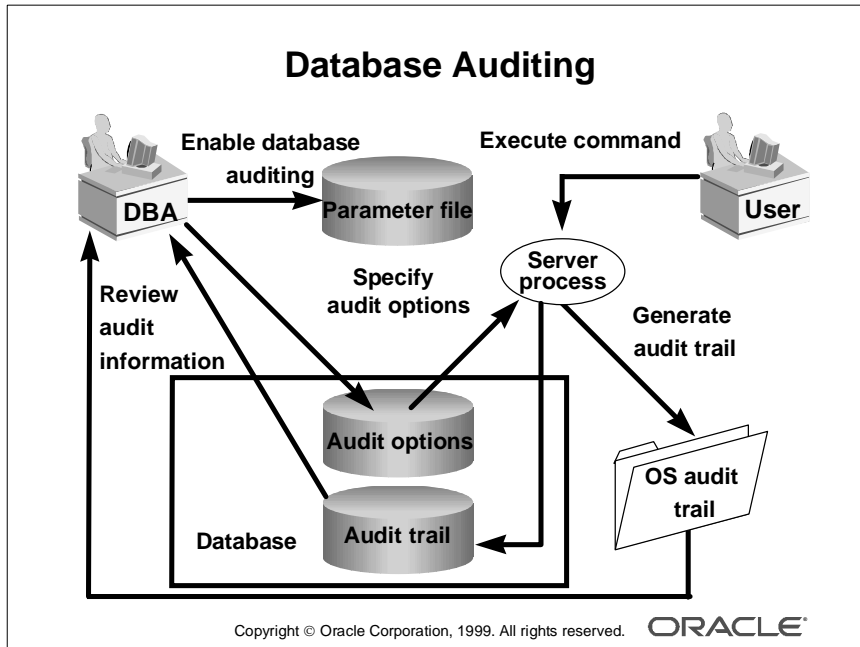
Auditing might also be used to monitor and gather data about specific database activities. For example, the database administrator can gather statistics about which tables are being updated, how many logical I/Os are performed, and how many concurrent users connect at peak times.

### **Value-Based Auditing**

Database auditing cannot record column values. If the changes to database columns need to be tracked and column values need to be stored for each change, use application auditing. Application auditing can be done either through client code, stored procedures, or database triggers.



## Using Database Auditing



### Enabling and Disabling Database Auditing

Once you have decided what to audit, you set the `AUDIT_TRAIL` initialization parameter to enable auditing for the instance. This parameter indicates whether the audit trail is written to a database table or the operating system audit trail.

**Syntax** `AUDIT_TRAIL = value`

where value can be one of the following:

DB	enables auditing and directs all audit records to the database audit trail (SYS.AUD\$)
OS	enables auditing and directs all audit records to the operating system audit trail (if permitted on the operating system)
NONE	disables auditing (This is the default value.)

### **Enabling Database Auditing (continued)**

Audit records will not be written to the audit trail unless the DBA has set the AUDIT\_TRAIL parameter to DB or OS. Although the SQL statements AUDIT and NOAUDIT can be used at any time, records will only be written to the audit trail if the DBA has set the AUDIT\_TRAIL parameter in the initialization file.

**Note:** The Installation and Configuration Guide for your OS system provides information on writing audit records to the OS audit trail.

### **Specifying Audit Options**

Next, you set specific auditing options using the AUDIT command. With the AUDIT command, you indicate which commands, users, objects, or privileges to audit. You can also indicate whether an audit record should be generated for each occurrence or once per session. If an auditing option is no longer required, you can turn off the option with the NOAUDIT command.

### **Execution of Statements**

When users execute PL/SQL and SQL statements, the server process examines the auditing options to determine if the statement being executed should generate an audit record. SQL statements inside PL/SQL program units are individually audited, as necessary, when the program unit is executed. Because views and procedures may refer to other database objects, several audit records may be generated as the result of executing a single statement.

### **Generating Audit Data**

The generation and insertion of an audit trail record is independent of a user's transaction; therefore, if a user's transaction is rolled back, the audit trail record remains intact. Since the audit record is generated during the execute phase, a syntax error, which occurs during the parse phase, will not cause an audit trail record to be generated.

### **Reviewing Audit Information**

Examine the information generated during auditing by selecting from the audit trail data dictionary views or by using an operating system utility to view the operating system audit trail. This information is used to investigate suspicious activity and to monitor database activity.

## Enabling Auditing Options

- **Statement auditing**

```
AUDIT user;
```

- **Privilege auditing**

```
AUDIT select any table  
BY summit BY ACCESS;
```

- **Schema object auditing**

```
AUDIT LOCK ON summit.employee  
BY ACCESS WHENEVER SUCCESSFUL;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Events Audited on Request

You can specify the auditing options using the AUDIT command. These audit records are never generated by sessions established by the user SYS or connections as INTERNAL. Connections by these users bypass certain internal features of the Oracle server to enable administrative operations to occur, such as database startup, shutdown, and recovery.

## Statement Auditing

You can audit by using a type of SQL statement or by a type of object. The statement auditing example audits all CREATE, ALTER, and DROP USER statements for all users.

Statement auditing options are typically broad, auditing the use of several types of related actions per option. For example, AUDIT TABLE tracks several DDL statements regardless of the table on which they are issued. You can set statement auditing to audit selected users or every user in the database.

## Privilege Auditing

Privilege auditing audits the use of system privileges. In the example in this slide, whenever SUMMIT uses the SELECT ANY TABLE privilege, an audit entry is generated; an entry will only be generated if SUMMIT queries tables belonging to other users, for which he has not received SELECT privileges. When auditing, owner privileges are checked first, then object privileges, and then system privileges. So if a user's SELECT ANY TABLE privilege is being audited, and he selects from a table he owns, then the SELECT ANY TABLE privilege would not cause an audit record to be generated, because the user can select from the table using his ownership privilege.

## Schema Object Auditing

Schema object auditing audits statements performed on a specific schema object. In the example, an audit trail entry is generated when a user successfully executes the LOCK command on the object SUMMIT.EMPLOYEE.

## Syntax

Use the following command to enable auditing options:

### Privilege or Statement Auditing

```
AUDIT {statement|system_priv}
      [, {statement|system_priv} ]...
      [BY user [, user ]... ]
      [BY {SESSION|ACCESS} ]
      [WHENEVER [NOT] SUCCESSFUL]
```

## Syntax (continued)

### Object Auditing

AUDIT statement [, statement ]...	
ON { [schema.]object   DEFAULT }	
[BY { <u>SESSION</u>   ACCESS }]	
[WHENEVER [NOT] SUCCESSFUL]	
where: statement	specifies the SQL statement type or schema-object to audit
system_priv	specifies the system privilege to audit
schema.schema-object	identifies the object chosen for auditing
DEFAULT	sets the specified object options as default object options for subsequently created objects
user	indicates to only audit the users in the list (If this clause is omitted, then all users' activities are audited.)
BY SESSION	causes the Oracle server to insert only one record per database object into the audit trail for each session, no matter how many SQL statements of the same type are submitted (This is the default, except for DDL.)
BY ACCESS	causes the Oracle server to insert a record into the audit trail each time an audited statement is submitted (For Data Definition Language (DDL) statements, the Oracle server always audits by access.)
WHENEVER	specifies that auditing is to be carried out only on successful or unsuccessful completion of SQL statements (The default is both.)

## Syntax (continued)

### Note

- Because audit records are generated during the execution phase, parse errors, such as TABLE OR VIEW DOES NOT EXIST, cannot be trapped by using the WHENEVER UNSUCCESSFUL clause.
- Statement and privilege auditing options specified by the AUDIT command apply only to subsequent sessions, not to the current session. In contrast, changes to schema object audit options become effective for current sessions immediately.

## Disabling Auditing

Use the NOAUDIT statement to stop auditing chosen by the AUDIT command.

**Note:** A NOAUDIT statement reverses the effect of a previous AUDIT statement. Note that the NOAUDIT statement must have the same syntax as the previous AUDIT statement and that it only reverses the effects of that particular statement. Therefore, if one AUDIT statement (statement A) enables auditing for a specific user, and a second (statement B) enables auditing for all users, then a NOAUDIT statement to disable auditing for all users reverses statement B, but leaves statement A in effect and continues to audit the user that statement A specified.

## Viewing Auditing Options

Data Dictionary View	Description
ALL_DEF_AUDIT_OPTS	Default audit options
DBA_STMT_AUDIT_OPTS	Statement auditing options
DBA_PRIV_AUDIT_OPTS	Privilege auditing options
DBA_OBJ_AUDIT_OPTS	Schema object auditing options

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Viewing Auditing Results

The views listed above contain information from the audit trail. The following is an example that shows the audit records generated when a series of statements are executed:

```
SQL> SELECT username, obj_name, action_name, priv_used
2  FROM sys.dba_audit_object
3  WHERE owner = 'SUMMIT'
4  AND obj_name = 'EMPLOYEE';
```

USERNAME	OBJ_NAME	ACTION_NAME	PRIV_USED
SUMMIT	EMPLOYEE	SESSION REC	
ADAMS	EMPLOYEE	SESSION REC	
SYSTEM	EMPLOYEE	SESSION REC	DELETE ANY TABLE

3 rows selected.

These results are only obtained when certain events occur in the database.

## Viewing Auditing Results

Viewing Auditing Results	
Audit Trail View	Description
DBA_AUDIT_TRAIL	All audit trail entries
DBA_AUDIT_EXISTS	Records for AUDIT EXISTS/NOT EXISTS
DBA_AUDIT_OBJECT	Records concerning schema objects
DBA_AUDIT_SESSION	All connect and disconnect entries
DBA_AUDIT_STATEMENT	Statement auditing records

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Audit Trail Views

The data dictionary views listed contain information on auditing options. These views are queried by the database administrator to determine what is being audited.

For example, the following query shows the privilege auditing options that are set:

```
SQL> SELECT * FROM dba_priv_audit_opts;
```

USER_NAME	PRIVILEGE	SUCCESS	FAILURE
SUMMIT	CREATE TABLE	BY ACCESS	BY ACCESS
SYSTEM	ALTER ANY TABLE	BY ACCESS	NOT SET
SUMMIT	ALTER ANY TABLE	BY ACCESS	NOT SET
SYSTEM	ALTER ANY PROCEDURE	BY ACCESS	NOT SET
SUMMIT	ALTER ANY PROCEDURE	BY ACCESS	NOT SET
GRANT	ANY PRIVILEGE	BY ACCESS	BY ACCESS

6 rows selected.



## Summary

### Summary

**In this lesson, you should have learned how to:**

- **Control system and object privileges**
- **Use database auditing**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Quick Reference

Context	Reference
Initialization parameters	O7_DICTIONARY_ACCESSIBILITY AUDIT_TRAIL
Dynamic performance views	None
Data dictionary views	DBA_SYS_PRIVS SESSION_PRIVS DBA_TAB_PRIVS DBA_COL_PRIVS ALL_DEF_AUDIT_OPTS AUDIT_ACTIONS DBA_AUDIT_EXISTS DBA_AUDIT_OBJECT DBA_AUDIT_SESSION DBA_AUDIT_STATEMENT DBA_AUDIT_TRAIL DBA_OBJ_AUDIT_OPTS DBA_PRIV_AUDIT_OPTS DBA_STMT_AUDIT_OPTS
Commands	GRANT REVOKE AUDIT NOAUDIT
Packaged procedures and functions	None

## **Managing Roles**

## Objectives

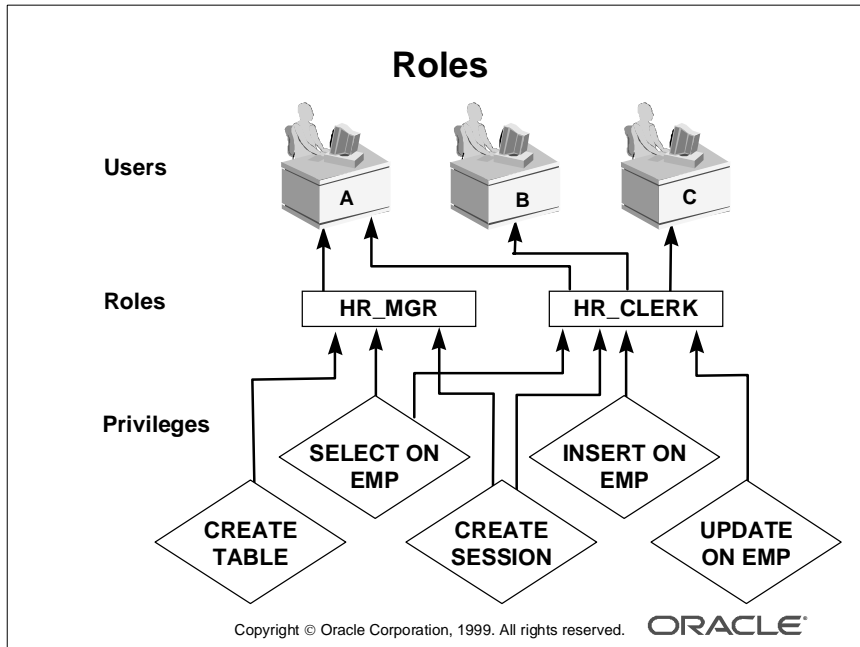
### Objectives

**After completing this lesson, you should be able to do the following:**

- **Create and modify roles**
- **Control availability of roles**
- **Remove roles**
- **Use predefined roles**
- **Display role information from the data dictionary**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Overview



### What Is a Role?

Oracle provides for easy and controlled privilege management through roles. Roles are named groups of related privileges that are granted to users or other roles. They are designed to ease the administration of privileges in the database.

### Role Characteristics

- Granted to and revoked from users with the same commands used to grant and revoke system privileges
- May be granted to any user or role, except to itself (even indirectly)
- Can consist of both system and object privileges
- May be enabled or disabled for each user granted the role
- Can require a password to enable
- Each role name must be unique among existing usernames and role names
- Are not owned by anyone; are not in any schema
- Have their descriptions stored in the data dictionary

## Benefits of Roles

- **Reduced granting of privileges**
- **Dynamic privilege management**
- **Selective availability of privileges**
- **Granted through the OS**
- **No cascading revokes**
- **Improved performance**

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### **Reduced Granting of Privileges**

Use roles to simplify privilege management. Rather than granting the same set of privileges to several users, you can grant the privileges to a role, and then grant that role to each user.

### **Dynamic Privilege Management**

If the privileges associated with a role are modified, all the users who are granted the role automatically and immediately acquire the modified privileges.

### **Selective Availability of Privileges**

Roles can be enabled and disabled to turn privileges on and off temporarily. Enabling a role can also be used to verify that a user has been granted that role.

### **Granted Through the Operating System**

Operating system commands or utilities can be used to assign roles to users in the database.

### **No Cascading Revokes**

Object privileges can be revoked without causing cascading revokes.

### **Improved Performance**

By disabling roles, there are fewer privileges to verify during statement execution. Using roles reduces the number of grants stored in the data dictionary.

## Creating and Modifying Roles

### Creating Roles

```
CREATE ROLE sales_clerk;
```

```
CREATE ROLE hr_clerk  
IDENTIFIED BY bonus;
```

```
CREATE ROLE hr_manager  
IDENTIFIED EXTERNALLY;
```

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Syntax

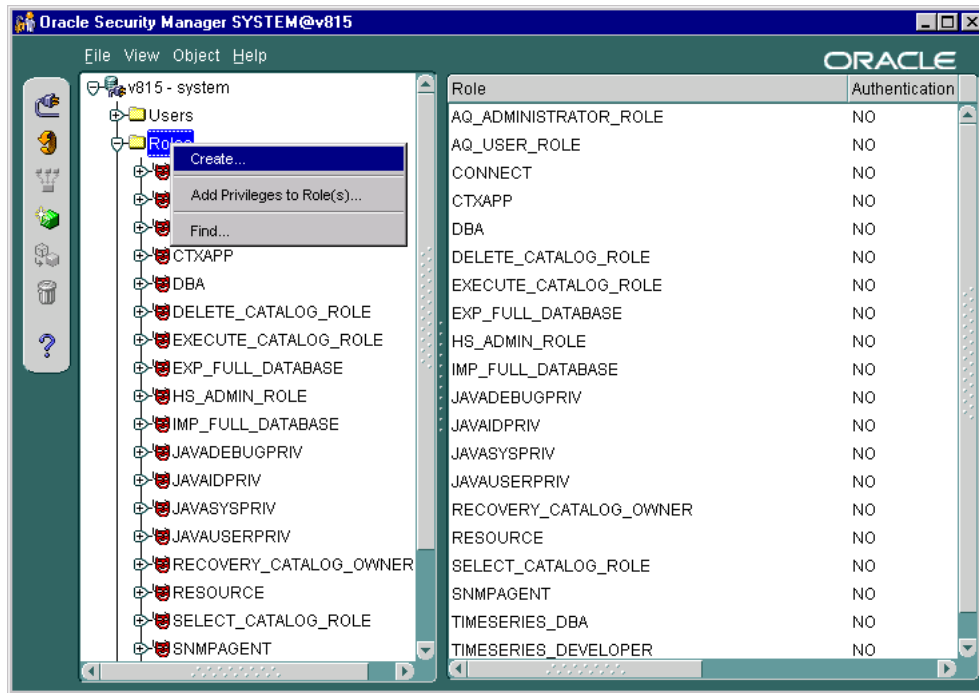
Use the following command to create a role:

```
CREATE ROLE role [NOT IDENTIFIED | IDENTIFIED  
{BY password | EXTERNALLY }]
```

where:	<i>role</i>	is the name of the role
	NOT IDENTIFIED	indicates that no verification is required when enabling the role
	IDENTIFIED	indicates that verification is required when enabling the row
	BY <i>password</i>	provides the password that the user must specify when enabling the role
	EXTERNALLY	indicates that a user must be authorized by an external service (such as the operating system or a third-party service) before enabling the role

## How to Use Oracle Enterprise Manager to Create a Role

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Select the Roles folder. Select Create from the right mouse button menu.



- 4 Enter the role name and select the identification method.
- 5 Optionally, GRANT roles and privileges to the new role by clicking the Roles, System Privileges or Object Privileges tab (covered in a previous lesson).
- 6 Click Create.

**Note:** The CREATE ROLE IDENTIFIED GLOBALLY command specifies that role verification must be done through the Oracle Security Server.

The Oracle Security Server is a security product that enables you to set up roles and users centrally in an Oracle distributed environment. Users and roles that are defined in the Oracle Security Server can be used across multiple databases. These users and roles are called global users and global roles respectively. See the manual *Oracle8i Server Distributed Database Systems* for more information.



### Using Predefined Roles

Role Name	Description
CONNECT, RESOURCE	These two roles are provided for backward compatibility.
DBA	All system privileges WITH ADMIN OPTION
EXP_FULL_DATABASE	Privileges to export the database
IMP_FULL_DATABASE	Privileges to import the database
DELETE_CATALOG_ROLE	DELETE privileges on data dictionary tables
EXECUTE_CATALOG_ROLE	EXECUTE privilege on data dictionary packages
SELECT_CATALOG_ROLE	SELECT privilege on data dictionary tables

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Predefined Roles

The roles listed are defined automatically for Oracle databases. CONNECT and RESOURCE roles are provided for backward compatibility to earlier versions of the Oracle server and can be modified in the same manner as any other role in an Oracle database.

The EXP\_FULL\_DATABASE and IMP\_FULL\_DATABASE roles are provided for convenience in using the Import and Export utilities.

The roles DELETE\_CATALOG\_ROLE, EXECUTE\_CATALOG\_ROLE, and SELECT\_CATALOG\_ROLE are provided for accessing data dictionary views and packages. These roles can be granted to users who do not have the DBA role but who require access to the views and tables in the data dictionary.

### Other Special Roles

The Oracle server also creates other roles that authorize you to administer the database. On many operating systems, these roles are called OSOPER and OSDBA. Their names may be different on your operating system.

Other roles are defined by SQL scripts provided with the database. For example, the roles AQ\_ADMINISTRATOR\_ROLE and AQ\_USER\_ROLE are created by the dbmsaqad.sql script. These roles are used with the Advanced Queuing feature.

## **Other Special Roles (continued)**

### **Note**

- On some platforms such as Solaris, grantees of the RESOURCE role also receive the UNLIMITED TABLESPACE privilege explicitly, although this privilege is not assigned to the role.
- You should not rely on these roles. Rather, it is recommended that you design your own roles for database security. These roles may not be created automatically by future versions of the Oracle server.

## Modifying Roles

```
ALTER ROLE sales_clerk
  IDENTIFIED BY commission;
```

```
ALTER ROLE hr_clerk
  IDENTIFIED EXTERNALLY;
```

```
ALTER ROLE hr_manager
  NOT IDENTIFIED;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

## Modifying Roles

A role can only be modified to change its authentication method.

### Syntax

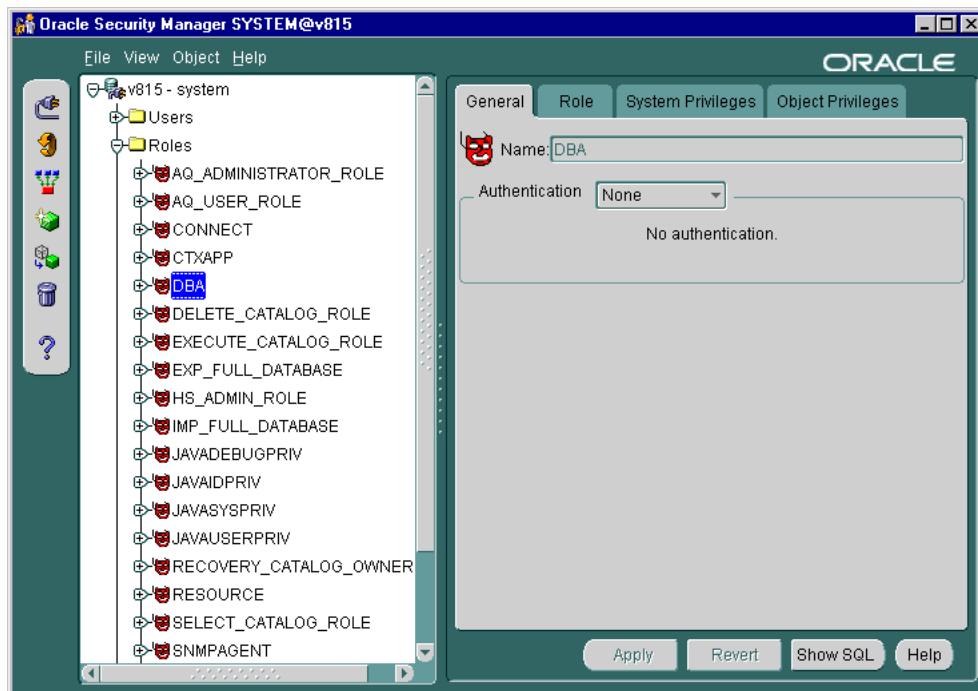
Use the following command to modify a role:

```
ALTER ROLE role {NOT IDENTIFIED | IDENTIFIED
  {BY password | EXTERNALLY } };
```

where:	role	is the name of the role
	NOT IDENTIFIED	indicates that no verification is required when enabling the role
	IDENTIFIED	indicates that verification is required when enabling the row
	BY <i>password</i>	provides the password used when enabling the role
	EXTERNALLY	indicates that a user must be authorized by an external service (such as the operating system or a third-party service) before enabling the role

## How to Use Oracle Enterprise Manager to Modify a Role

- 1 Launch Security Manager and connect directly to the database  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Roles folder.
- 4 Select the role.
- 5 Indicate the identification method.



- 6 Click Apply.

## Assigning Roles

### Assigning Roles

```
GRANT sales_clerk TO scott;
```

```
GRANT hr_clerk,  
TO hr_manager;
```

```
GRANT hr_manager TO scott  
WITH ADMIN OPTION;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE

### Syntax

To grant a role to a user, use the same syntax command that was used to grant a system privilege to a user:

```
GRANT role [, role ]...  
TO {user|role|PUBLIC}  
[, {user|role|PUBLIC} ]...  
[WITH ADMIN OPTION]
```

where:

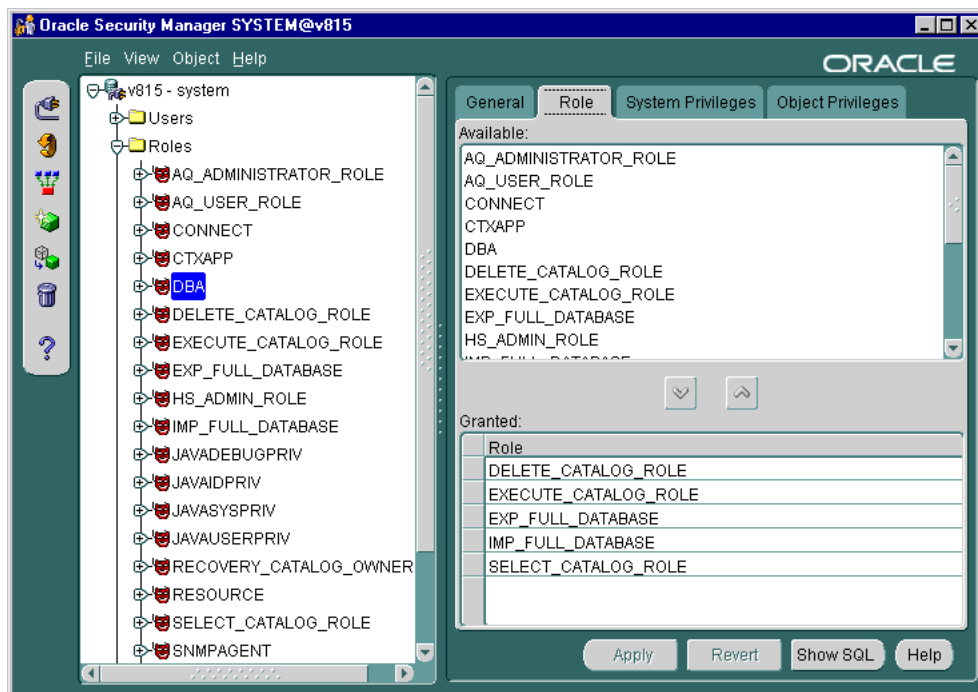
role	is a role to be granted or a role receiving the role granted
user	is a user receiving a role
role	is a role receiving a role
PUBLIC	grants the role to all users

WITH ADMIN OPTION enables the grantee to grant the role to other users or roles. (If you grant a role with this option, the grantee can grant and revoke the role from other users and alter or drop the role.)

The user who creates a role is implicitly assigned the role with ADMIN OPTION. A user who has not been granted a role with ADMIN OPTION requires the GRANT ANY ROLE system privilege to grant and revoke roles to and from others.

## How to Use Oracle Enterprise Manager to Assign a Role

- 1 Launch Security Manager and connect directly to the database  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Users or Roles folder.
- 4 Select the user or role.
- 5 Select the Role or System Privileges tab.
- 6 Select the Role to be granted.
- 7 Click the down arrow to add the role to the granted list.



- 8 Add additional roles as required.
- 9 Click Apply.

## Controlling Availability of Roles

### Establishing Default Roles

```
ALTER USER scott
DEFAULT ROLE hr_clerk, sales_clerk;
```

```
ALTER USER scott DEFAULT ROLE ALL;
```

```
ALTER USER scott DEFAULT ROLE ALL EXCEPT
hr_clerk;
```

```
ALTER USER scott DEFAULT ROLE NONE;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE

### Default Roles

A user may have many roles assigned. A *default role* is a subset of these roles that is automatically enabled when the user logs on. By default, all the roles assigned to a user are enabled at logon. Limit the default roles for a user with the ALTER USER command.

### Syntax

Use the following syntax to assign default roles to a user:

```
ALTER USER user DEFAULT ROLE
{role [,role]... | ALL [EXCEPT role [,role]... ] | NONE}
```

where:	user	is the name of the user granted the roles
	role	is the role to be made the default role for the user
	ALL	makes all of the roles granted to the user default roles, except those listed in the EXCEPT clause (This is the default.)
	EXCEPT	indicates that the following roles should not be included in the default roles

**Syntax (continued)**

NONE

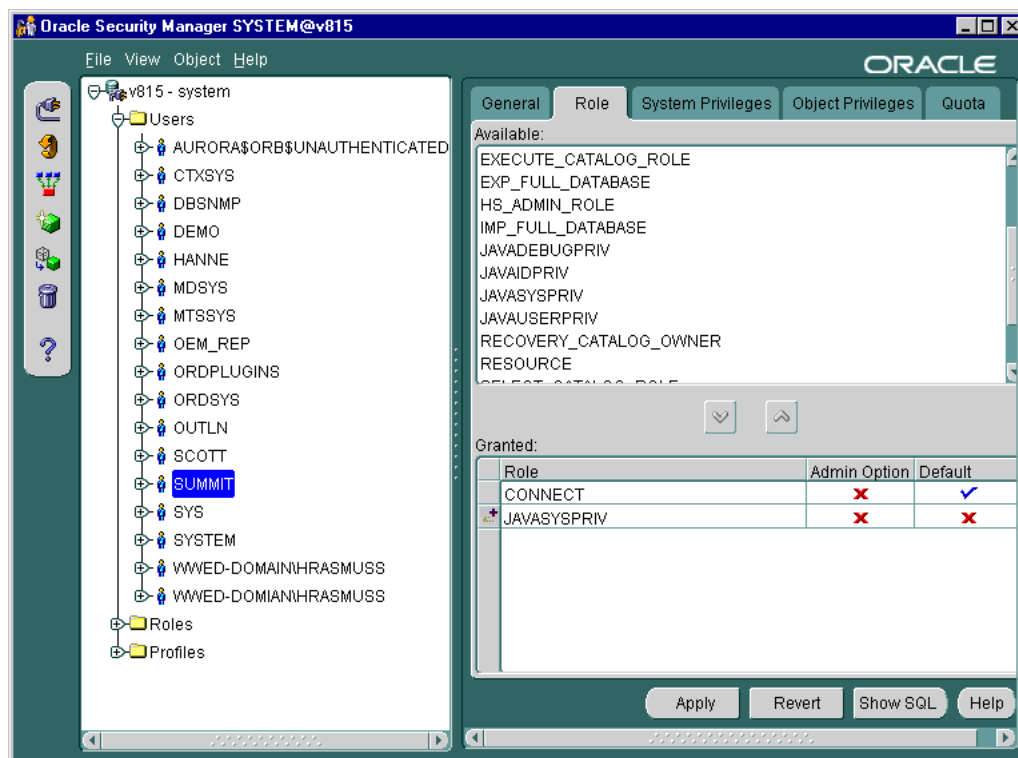
makes none of the roles granted to the user default roles (The only privileges that the user has at login are those privileges assigned directly to the user.)

Because the roles must be granted before they can be made defaults, you cannot set default roles with the CREATE USER command.

For roles that are authenticated with a password, the password is not required when the role is a default role.

**How to Use Oracle Enterprise Manager to Assign Default Roles**

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Users folder.
- 4 Select the user.
- 5 Select the Role tab.
- 6 Select the box Default for the default roles.



- 7 Click Apply.



## Enabling and Disabling Roles

- **Disable a role to temporarily revoke the role from a user.**
- **Enable a role to temporarily grant it.**
- **The SET ROLE command enables and disables roles.**
- **Default roles are enabled for a user at login.**
- **A password may be required to enable a role.**

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Enabling and Disabling Roles

Enable or disable roles to temporarily activate and deactivate the privileges associated with the roles. To enable a role, the role must first be granted to the user.

When a role is enabled, the user can use the privileges granted to that role. If a role is disabled, the user cannot use the privileges associated with that role unless that privilege is granted directly to the user or to another role enabled for that user. Roles are enabled for a session. At the next session, the user's active roles will revert to default roles.

## Specifying Roles to Be Enabled

The SET ROLE command and the DBMS\_SESSION.SET\_ROLE procedure enable all of the roles included in the command and disable all other roles. Roles can be enabled from any tool or program that allows PL/SQL commands; however, a role cannot be enabled in a stored procedure.

You can use the ALTER USER...DEFAULT ROLE command to indicate which roles will be enabled for a user at login. All other roles are disabled.

A password may be required to enable a role. The password must be included in the SET ROLE command to enable the role. Default roles assigned to a user do not require a password; they are enabled at login, the same as a role without a password.

## Restrictions

A role cannot be enabled from a stored procedure, because this action may change the security domain (set of privileges) that allowed the procedure to be called in the first place. So, in PL/SQL, roles can be enabled and disabled in anonymous blocks and application procedures (for example, Oracle Forms procedures), but not in stored procedures.

If a stored procedure contains the command SET ROLE, the following error is generated at run time:

```
ORA-06565: cannot execute SET ROLE from within stored procedure
```

## Enabling and Disabling Roles: Examples

```
SET ROLE hr_clerk;
```

```
SET ROLE sales_clerk IDENTIFIED BY  
commission;
```

```
SET ROLE ALL EXCEPT sales_clerk;
```

```
SET ROLE NONE;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

## Syntax

Use the following commands to enable and disable roles:

```
SET ROLE {role [ IDENTIFIED BY PASSWORD ]  
        [, role [ IDENTIFIED BY PASSWORD ]]...  
        | ALL [ EXCEPT role [, role ] ...]  
        | NONE }
```

The SET ROLE command turns off any other roles granted to the user.

where:	role	is the name of the role
	IDENTIFIED	
	BY <i>password</i>	provides the password required when enabling the role
	ALL	enables all roles granted to the current user, except those listed in the EXCEPT clause (You cannot use this option to enable roles with passwords.)
	EXCEPT <i>role</i>	does not enable these roles
	NONE	disables all roles for the current session (Only privileges granted directly to the user are active.)

The ALL option without the EXCEPT clause works only when every role that is enabled does not have a password.

## Removing Roles from Users

```
REVOKE sales_clerk FROM scott;
```

```
REVOKE hr_manager FROM PUBLIC;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Syntax

To revoke a role from a user, use the same command syntax that was used to revoke a system privilege from a user:

```
REVOKE role [, role ]...  
FROM {user|role|PUBLIC}  
[, {user|role|PUBLIC} ]...
```

where:	role	is the role to be revoked or the role from which roles are revoked
	user	is the user from which the system privileges or roles are revoked
	PUBLIC	revokes the privilege or role from all users

### How to Use Oracle Enterprise Manager to Revoke a Role

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Users or Roles folder.
- 4 Select the user or role.
- 5 Select the Roles or System Privileges tab.
- 6 Under Granted, select the Role to be revoked.
- 7 Click the up arrow to remove the role from the granted list.
- 8 Click Apply.

## Removing Roles

```
DROP ROLE hr_manager;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Syntax

To remove a role from the database, use the following syntax:

```
DROP ROLE role
```

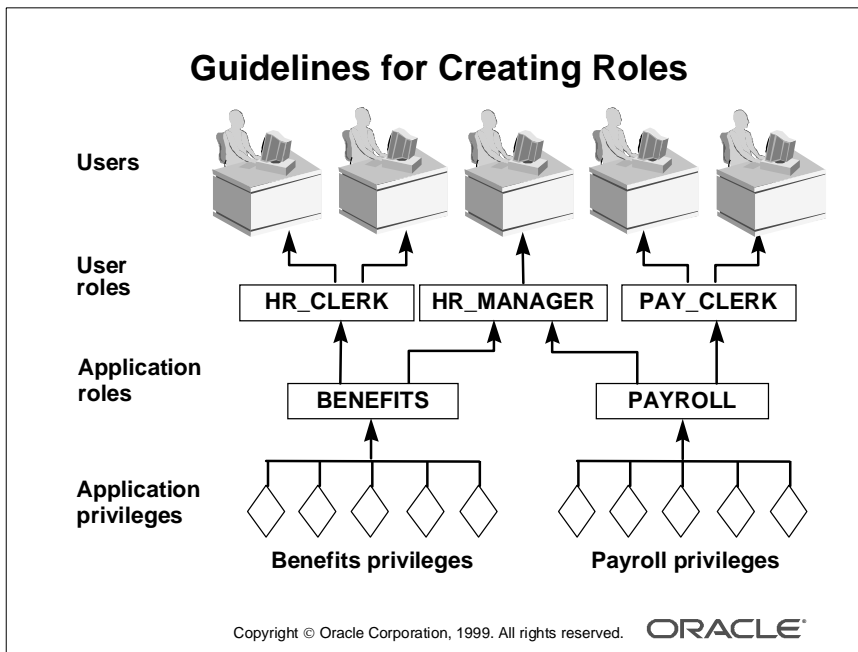
where:     role                     is the role to be removed

When you drop a role, the Oracle server revokes it from all users and roles to whom it has been granted and removes it from the database.

You must have been granted the role with ADMIN OPTION or have the DROP ANY ROLE system privilege to drop the role.

### How to Use Oracle Enterprise Manager to Remove a Role

- 1 Launch Security Manager and connect directly to the database:  
Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager
- 2 Enter the login information, and click the OK button.
- 3 Expand the Roles folder.
- 4 Select the role.
- 5 Select Object—>Remove.
- 6 In the dialog box, click Yes.

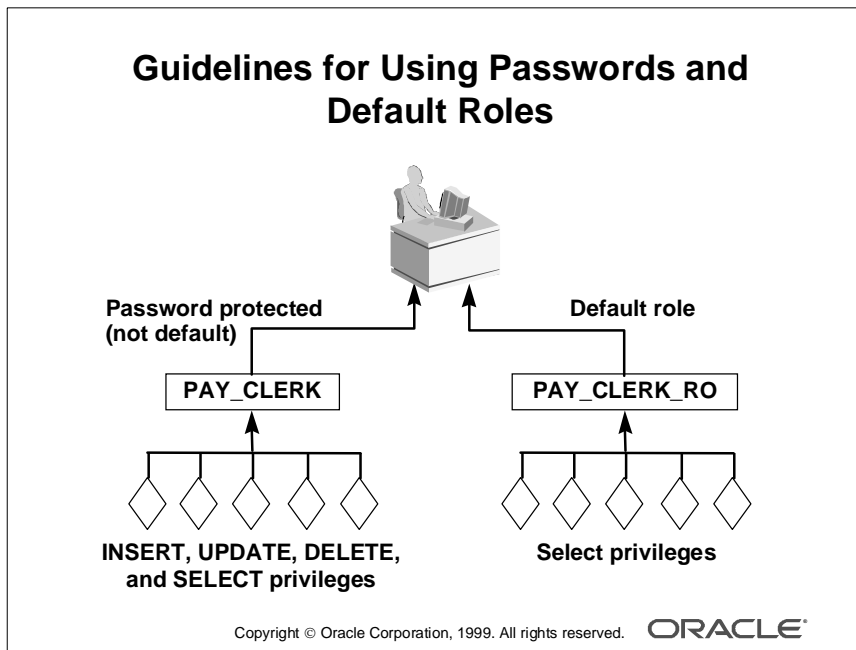


## Guidelines for Creating Roles

Because a role includes the privileges necessary to perform a task, the role name is usually an application task or a job title. The example above uses both application tasks and job titles for role names.

- 1 Create a role for each application task. The name of the application role corresponds to a task in the application, such as payroll.
- 2 Assign the privileges necessary to perform the task to the application role.
- 3 Create a role for each type of user. The name of the user role corresponds to a job title, such as `PAY_CLERK`.
- 4 Grant only application roles, not individual privileges, to user's roles.
- 5 Grant user and application roles to users.

If a modification to the application requires that new privileges are needed to perform the payroll task, then the DBA only needs to assign the new privileges to the application role, `PAYROLL`. All of the users that are currently performing this task will receive the new privileges.



### Using Passwords

- Passwords provide an additional level of security when enabling a role. For example, the application might require a user to enter a password when enabling the PAY\_CLERK role, because this role can be used to issue checks.
- Passwords allow a role to be enabled only through an application. This technique is shown in the example above.
  - The DBA has granted the user two roles, PAY\_CLERK and PAY\_CLERK\_RO.
  - The PAY\_CLERK has been granted all of the privileges needed to perform the payroll clerk function.
  - The PAY\_CLERK\_RO (RO for read only) has been granted only SELECT privileges on the tables required to perform the payroll clerk function.
  - The user can log in to SQL Plus to perform queries, but cannot modify any of the data, because the PAY\_CLERK is not a default role, and the user does not know the password for PAY\_CLERK.
  - When the user logs on to the payroll application, it enables the PAY\_CLERK by providing the password. It is coded in the program; the user is not prompted for it.

## Displaying Role Information

Displaying Role Information	
Role View	Description
DBA_ROLES	All roles that exist in the database
DBA_ROLE_PRIVS	Roles granted to users and roles
ROLE_ROLE_PRIVS	Roles that are granted to roles
DBA_SYS_PRIVS	System privileges granted to users and roles
ROLE_SYS_PRIVS	System privileges granted to roles
ROLE_TAB_PRIVS	Table privileges granted to roles
SESSION_ROLES	Roles that the user currently has enabled

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Query Role Information

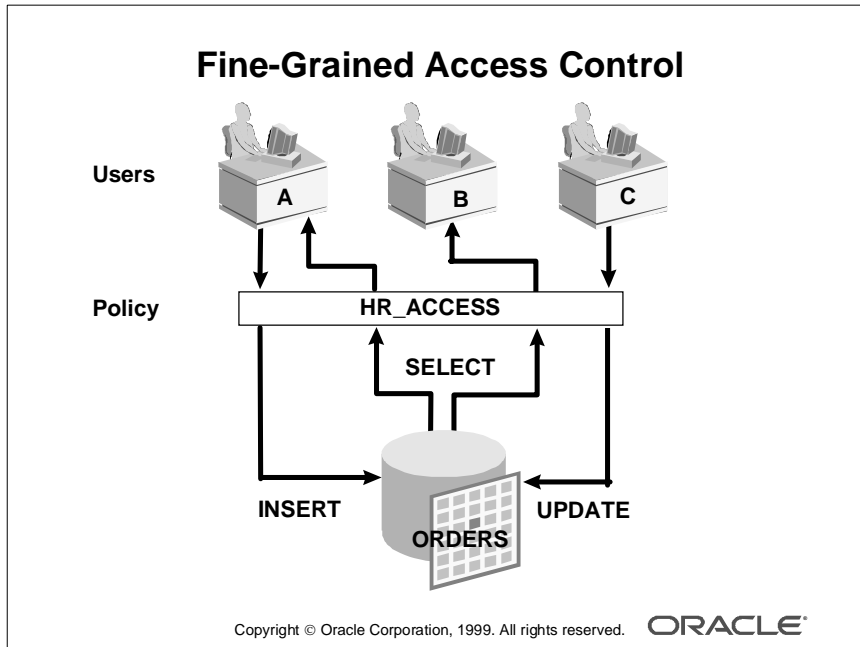
Many of the data dictionary views that contain information on privileges granted to users also contain information on privileges to roles.

```
SQL> SELECT role, password_required FROM dba_roles;
```

ROLE	PASSWORD
-----	-----
CONNECT	NO
RESOURCE	NO
DBA	NO
...	..
SELECT_CATALOG_ROLE	NO
EXECUTE_CATALOG_ROLE	NO
DELETE_CATALOG_ROLE	NO
IMP_FULL_DATABASE	NO
EXP_FULL_DATABASE	NO
SALES_CLERK	YES
HR_CLERK	EXTERNAL



## Using Fine-Grained Access Control



### What Is Fine-Grained Access Control?

Fine-grained access control allows you to implement security policies with functions and then associate those security policies with tables or views. The database automatically enforces those security policies, no matter how the data is accessed.

You can:

- Use different policies for SELECT, INSERT, UPDATE, and DELETE
- Use security policies only where you need them; for example, on salary information
- Use more than one policy for each table, including building on top of base policies in packaged applications

The PL/SQL package DBMS\_RLS allows you to administer your security policies. Using this package, you can add, drop, enable, disable, and refresh policies you create.

**Note:** How to implement fine-grained access control is not covered in this course.

### **Fine-Grained Access Control: How It Works**

- **Direct or indirect user access to object with an attached policy automatically invokes the policy.**
- **Policy package returns a predicate (a WHERE condition).**
- **The database dynamically rewrites the SQL statement by appending the predicate.**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### **How Does Fine-Grained Access Control Work?**

The implementation of fine-grained access control is through dynamic modification. When a user accesses an object (directly or through a subquery) that has a security policy attached to it, the RDBMS automatically consults the package that implements the policy for that view or table. The policy returns a predicate (access condition) that is appended to the query. The statement is then parsed, optimized, and executed.

If there are multiple policies attached to a table, the data server “ANDs” the predicates returned by each policy. For example, you might want one policy for queries and another for some or all of the DML statements (INSERT, UPDATE, DELETE).

## Summary

### Summary

In this lesson, you should have learned how to:

- Create roles
- Assign privileges to roles
- Assign roles to users or roles
- Establish default roles

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE<sup>®</sup>

## Quick Reference

Context	Reference
Initialization parameters	None
Dynamic performance views	None
Data dictionary views	DBA_ROLES DBA_ROLE_PRIVS DBA_SYS_PRIVS ROLE_ROLE_PRIVS ROLE_SYS_PRIVS ROLE_TAB_PRIVS SESSION_ROLES
Commands	CREATE_ROLE ALTER ROLE DROP ROLE SET ROLE ALTER USER ... DEFAULT ROLES GRANT REVOKE
Packaged procedures and functions	DBMS_SESSION.SET_ROLE

## **Using National Language Support**

## Objectives

### Objectives

After completing this lesson, you should be able to do the following:

- Choose character set and national character set for a database
- Specify the language-dependent behavior using initialization parameters, environment variables, and the ALTER SESSION command
- Use the different types of National Language Support (NLS) parameters
- Explain the influence on language-dependent application behavior
- Obtain information about NLS usage

Copyright © Oracle Corporation, 1999. All rights reserved.


ORACLE®

---

## Overview

### NLS Features

- Language support
- Territory support
- Character set support
- Linguistic sorting
- Message support
- Date and time formats
- Numeric formats
- Monetary formats



Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### National Language Support

The National Language Support (NLS) ensures that database utilities and error messages, sort order, date, time, monetary, numeric, and calendar conventions automatically adapt to the native language.

Oracle currently supports about 45 languages, 60 territories, 60 linguistic sorts, and many encoded character sets.

The language-dependent operations are controlled by a number of parameters and environment variables on both the client and the server sides.

Server and client may run in the same or different locations. When client and server use different character sets, the Oracle server handles character set conversion automatically.

## **National Language Support**

- Users can interact, store, process, and retrieve data in their native language, including Western European, Eastern European, Middle Eastern, East Asian, and Southeast Asian languages.
- Different countries and geographies dictate different cultural conventions that directly affect data formats.
- Many different character encoding schemes, including single-byte, multibyte, and fixed-width encoded character sets are supported.
- The Oracle server provides many different linguistic sorts for linguistically accurate sorting.
- Database utilities and error messages appear in the supported native language. The Oracle products are translated into 26 different languages.
- Date and time formats can be expressed according to ISO conventions for hour, day, month, and year.
- National calendars such as Gregorian, Japanese, Imperial, and Thai Buddha are supported.
- Numeric data is represented in the appropriate local formats.
- Currency symbols reflect the local economy and the ISO conventions. Credit and debit symbols also differ from location to location.



## Choosing a Database and a National Character Set

### Different Types of Encoding Schemes

Oracle supports different classes of character encoding schemes:

- **Single-byte character sets**
  - 7-bit
  - 8-bit
- **Varying-width multibyte character set**
- **Fixed-width multibyte character set**
- **Unicode (UTF8, AL24UTFSS)**

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Character Encoding Schemes

A character encoding scheme specifies numeric codes corresponding to characters that a computer or terminal can display and receive.

Character encoding schemes are used to interpret data into meaningful symbols from a terminal to a host machine.

Oracle provides different classes of encoding schemes:

- Single-byte
- Varying-width
- Fixed-width
- Unicode

### Single-Byte Character Sets

In a single-byte character set, each character occupies one byte. Single-byte 7-bit encoding schemes can define up to 128 ( $2^7$ ) characters; single-byte 8-bit encoding schemes can define up to 256 ( $2^8$ ) characters.

### **Examples of Single-Byte Schemes**

7-bit character set: ASCII 7-bit American (US7ASCII)

8-bit character set:

- ISO 8859-1 West European (WE8ISO8859P1)
- EBCDIC Code Page 500 8-bit West European (WE8EBCDIC500)
- DEC 8-bit West European (WE8DEC)

### **Varying-Width Multibyte Character Sets**

A multibyte character set is represented by one or more bytes per character. Multibyte character sets are commonly used for Asian language support. Some multibyte encoding schemes use the value of the most significant bit to indicate if a byte represents a single byte or is part of a series of bytes representing a character. However, other character encoding schemes differentiate single-byte from multibyte characters. A shift-out control code, sent by a device, indicates that the following bytes are double-byte characters, until a shift-in code is encountered.

### **Examples of Varying-Width Multibyte Schemes**

- Japanese Extended UNIX Code (JEUC)
- Chinese GB2312-80 (CGB2312-80)

### **Fixed-Width Multibyte Character Sets**

Fixed-width character sets provide support similar to multibyte character sets, except that the format is a fixed number of bytes for each character.

This provides the benefits of having a uniform byte size representation for each character.

### **Examples of Fixed-Width Multibyte Character Sets**

- JA16EUCFIXED, 16-bit Japanese (a fixed-width subset of JA16EUC)
- JA16SJISFIXED, 16-bit Japanese (a fixed-width subset of JA16SIJS)

## Unicode Character Set

Unicode is a worldwide character-encoding standard that can represent all characters for computer usage, including technical symbols and characters used in publishing. In total, Unicode version 2.0 can represent 38,885 characters.

The Unicode character repertoire can be represented in a number of different encoding formats.

UCS2 (Universal Character Set; 2-byte form) is a two-byte, fixed-width format; UTF8 (Universal Character Set Transformation Format) is a multibyte, varying-width format. UCS2 and UTF8 encode the same character repertoire: Unicode 1.1 or Unicode 2.0.

Oracle7 uses Unicode 1.1 encoded as UTF8 (character set: AL24UTFFSS); Oracle8 additionally provides Unicode 2.0 encoded as UTF8 (character set: UTF8). The advantage of UTF8 is that it includes ASCII using the same single-byte encoding.

## Character Sets and National Character Sets of a Database

Database Character Sets	National Character Sets
Defined at creation time	Defined at creation time
Cannot be changed without re-creation	Cannot be changed without re-creation
Store data columns of type CHAR, VARCHAR2, CLOB, LONG	Store data columns of type NCHAR, NVARCHAR2 and NCLOB
Can store varying-width character sets	Can store fixed-width and varying-width multibyte character sets

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

### Character Set Types

The CREATE DATABASE statement has the CHARACTER SET clause and the additional optional clause NATIONAL CHARACTER SET to declare the character set to be used as the *database character set* and the *national character set*. Neither character set can be changed after creating the database. If no NATIONAL CHARACTER SET clause is present, the national character set defaults to the database character set.

Because the database character set is used to identify and to hold SQL and PL/SQL source code, it must have either EBCDIC or 7-bit ASCII as a subset, whichever is native to the platform. Therefore, it is not possible to use a fixed-width, multibyte character set as the database character set, only as the national character set.

The data types NCHAR, NVARCHAR2, and NCLOB are provided to declare columns as variants of the basic types CHAR, VARCHAR2, and CLOB, to note that they are stored using the national character set and not the database character set.

- To declare a fixed-length character item that uses the national character set, use the data type specification NCHAR [(size)].
- To declare a variable-length character item that uses the national character set, use the data type specification NVARCHAR2 (size).
- To declare a character large object (CLOB) item containing fixed-width, multibyte characters that uses the national character set, use the data type specification NCLOB (size).

## **Character Set Types (continued)**

### **Note**

- Oracle does not support the national character set on the LONG data type.
- The fixed or varying-width aspect of a character set's encoding is independent of the fixed or varying-length aspect of the types CHAR and VARCHAR2. One aspect refers to the number of bytes needed by each character in a string, the other to the total space allocated for the string. Either a fixed-width or a varying-width character set can be used for a column of a fixed-length type (CHAR or NCHAR), and similarly for a varying-length type (VARCHAR2 or NVARCHAR2).

### **Guidelines**

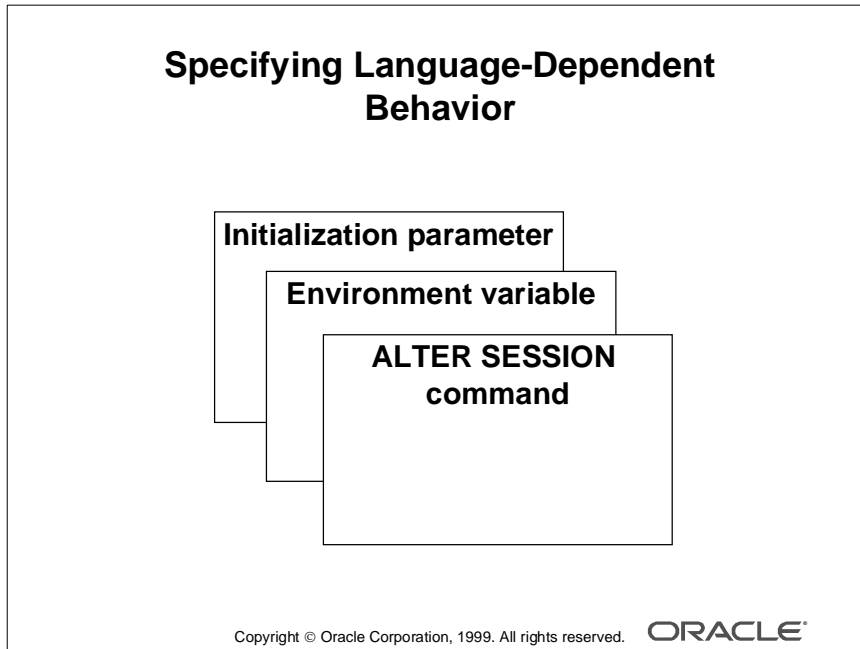
- **Choose a closely related database character set and national character set.**
- **String operations might be faster with fixed-width character sets.**
- **Variable-width character sets use space more efficiently.**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### **Guidelines**

The database character set and the national character set should be closely related; for example, Japanese customers will choose JA16EUC as the database character set and JA16EUCFIXED as the national character set.

## Specifying Language-Dependent Behavior



### Three Ways to Specify NLS Parameters

There are three ways to specify NLS parameters:

- As initialization parameters on the server side to specify the default server NLS environment (These default settings have no effect on the client side.)
- As environment variables for the client to specify locale-dependent behavior overriding the defaults set for the server
- As the ALTER SESSION parameter to override the default set for the session or the server

## Specifying Language-Dependent Behavior for the Server

- **NLS\_LANGUAGE specifies:**
  - The language for messages
  - Day and month names
  - Symbols for A.D, B.C, A.M, P.M.
  - The default sorting mechanism
- **NLS\_TERRITORY specifies:**
  - Day and week numbering
  - Default date format, decimal character, group separator, and the default ISO and local currency symbols

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### NLS Initialization Parameters

The initialization parameter `NLS_LANGUAGE` defines the value for language-dependent conventions, such as:

- Language used for Oracle messages
- Language used for day and month names and their abbreviations
- Symbols used for language-equivalents of a.m, p.m, A.D., and B.C.
- Default sorting sequence of character data

The initialization parameter `NLS_TERRITORY` defines values for territory-dependent conventions, which include:

- Default date format
- Decimal character and group separator
- Local currency symbol
- ISO currency symbol
- ISO week number calculation
- Week start day

**Note:** When the territory name contains a space, as in The Netherlands, the territory name should be enclosed in double quotes, for example “The Netherlands.”



### Dependent Language and Territory Default Values

PARAMETER	VALUES
NLS_LANGUAGE	AMERICAN
NLS_DATE_LANGUAGE	AMERICAN
NLS_SORT	BINARY
NLS_TERRITORY	AMERICA
NLS_CURRENCY	\$
NLS_ISO_CURRENCY	AMERICA
NLS_DATE_FORMAT	DD-MON-YY
NLS_NUMERIC_CHARACTERS	,.

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE

### NLS Initialization Parameters

The initialization parameter NLS\_LANGUAGE determines the default values of the following parameters:

Column	Description
NLS_DATE_LANGUAGE	Explicitly changes the language for day and month names and abbreviations and spelled values of other date format elements
NLS_SORT	Changes the linguistic sort sequence the Oracle server uses to sort character values (The sort value must be either BINARY or the name of a linguistic sort sequence.)

## NLS Initialization Parameters (continued)

NLS\_TERRITORY determines the default values for the following parameters:

Column	Description
NLS_CURRENCY	Explicitly specifies a new local currency symbol
NLS_ISO_CURRENCY	Explicitly specifies the territory whose ISO currency symbol should be used
NLS_DATE_FORMAT	Explicitly specifies a new default date format (The value must be a date format model.)
NLS_NUMERIC_CHARACTERS	Explicitly specifies a new decimal character and group separator

## Dual Currency Support for the Euro

On January 1, 1999, the new currency of the European union, the euro, made its debut. In order to support the new European Union currency, dual currency support has been added for given territories. An initialization parameter NLS\_DUAL\_CURRENCY sets an alternate currency symbol for the user session.

The following territories have the euro symbol added for dual currency support:

Austria	Italy
Belgium	Luxembourg
Finland	Netherlands
France	Portugal
Germany	Spain
Ireland	

The ISO character sets, such as WE8ISO8859P1, MS Code Pages and IBM code pages, have specified code points for the euro symbol.

## Specifying Language-Dependent Behavior for the Session

- Environment variable:  
NLS\_LANG=<language>\_<territory>.<charset>
- Additional environment variables:
  - NLS\_DATE\_FORMAT
  - NLS\_DATE\_LANGUAGE
  - NLS\_SORT
  - NLS\_NUMERIC\_CHARACTERS
  - NLS\_CURRENCY
  - NLS\_ISO\_CURRENCY
  - NLS\_CALENDAR

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### The Environment Variable NLS\_LANG

Override the default NLS behavior for an individual user with the NLS\_LANG environment variable. The value of NLS\_LANG overrides any values of the NLS initialization parameters.

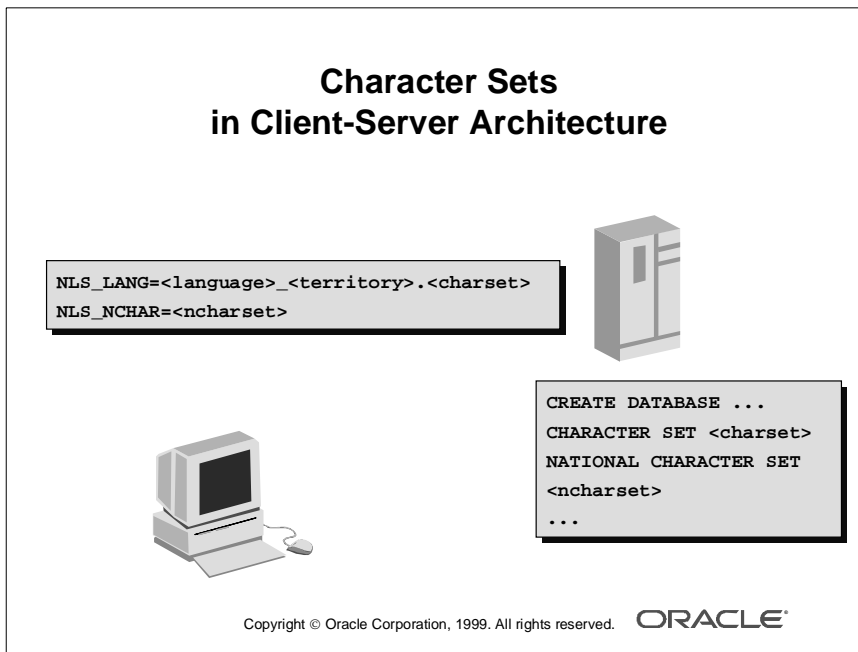
Each component controls a subset of NLS features:

NLS\_LANG=<language>\_<territory>.<charset>

For example:

NLS\_LANG=GERMAN\_GERMANY.WE8ISO8859P1

where:	language	overrides the value of NLS_LANGUAGE and controls the same features as NLS_LANGUAGE
	territory	overrides the value of NLS_TERRITORY and controls the same features as NLS_TERRITORY
	charsetset	specifies the character encoding scheme used by client application (normally that of the user's terminal)



### The Environment Variable NLS\_LANG (continued)

NLS\_LANG defines a client terminal's character encoding scheme. Different clients can use different encoding schemes. Data passed between client and server is converted automatically between the two encoding schemes. The database encoding scheme should be a superset, or equivalent of, all the client encoding schemes. The conversion is transparent to the client application.

### Additional Environment Variables

All NLS initialization parameters are available as environment variables, making it possible to specify individual NLS characteristics for each client.

In addition NLS\_CALENDAR can be used to specify which calendar system the Oracle server uses; for example, Gregorian, Persian, or Thai Buddha.

The following variables can be set only in the client environment:

- NLS\_CREDIT
- NLS\_DEBIT
- NLS\_DISPLAY
- NLS\_LANG
- NLS\_LIST\_SEPARATOR
- NLS\_MONETARY
- NLS\_NCHAR

## **Additional Environment Variables (continued)**

### **Note**

- The description of these parameters can be found in the *Oracle8i: Server Reference Manual*.
- If the environment variable ORA\_NLS33 (see the lesson “Creating a Database”) is not set, it is possible to create the database only with the default character set US7ASCII. ORA\_NLS should be set on UNIX as follows:  
`$ORACLE_HOME/ocommon/nls/admin/data`
- On Windows NT, parameters such as NLS\_LANG and ORA\_NLS33 are automatically set during the installation and stored in the registry in the folder HKEY\_LOCAL\_MACHINE\SOFTWARE\ORACLE.

## Specifying Language-Dependent Behavior for the Session

```
ALTER SESSION SET  
NLS_DATE_FORMAT='DD.MM.YYYY';
```

```
DBMS_SESSION.SET_NLS('NLS_DATE_FORMAT',  
'''DD.MM.YYYY''') ;
```

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

### Changing NLS Parameters

Change individual NLS characteristics for a session with the ALTER SESSION command. All environment variables that can be set on both the client and server sides can also be modified by issuing the ALTER SESSION command.

In the above example the date format is changed for the session.

Also, tools such as SQL\*Plus reads the environment variables and issue the corresponding ALTER SESSION command.

In addition to explicitly issuing ALTER SESSION commands, there is a database package DBMS\_SESSION.SET\_NLS that takes the name and the value of the parameter.

## NLS Parameters and SQL Functions

### Sorting

- Oracle provides a linguistic sort.
- NLS\_SORT specifies types of sort.
- The NLSSORT function reflects linguistic comparison.

```
ALTER SESSION SET NLS_SORT=GERMAN;  
SELECT letter FROM letters ORDER BY letter;  
LETTER  
-----  
ä  
z
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### How NLS Affects Sorts

A binary sort is a conventional sorting mechanism by which letters are sorted according to the binary values used to encode the characters. The alphabetic position of a character may vary for different languages.

For example *ä* is sorted before *b* in German, but *ä* occurs after *z* if you use a binary sort.

To overcome the limitations of binary sorting, the Oracle server provides linguistic sorts by setting the NLS\_SORT parameter.

## How NLS Affects Sorts (continued)

The following examples illustrate sorting behavior:

```
SQL> ALTER SESSION SET NLS_SORT=BINARY;
Session altered.
SQL> SELECT letter FROM letters ORDER BY letter;
L
-
a
b
c
z
ü
ä
6 rows selected.
SQL> ALTER SESSION SET NLS_SORT= GERMAN;
Session altered.
SQL> SELECT letter FROM LETTERS ORDER BY 1;
L
-
ä
ü
a
b
c
z
6 rows selected.
```



**How NLS Affects Sorts (continued)**

The NLSSORT function can be used, to enable comparison according to linguistic conventions and not binary values.

```
SQL> SELECT letter FROM letters WHERE letter < 'z';
```

```
L  
-  
a  
b  
c
```

```
SQL> SELECT letter FROM letters  
2  WHERE NLSSORT(letter) < NLSSORT('z');
```

```
L  
-  
a  
b  
ä  
c  
ü
```

## NLS Parameters in SQL Functions

### Using NLS Parameters in SQL Functions

```
SELECT TO_CHAR(hiredate,'DD.MON.YYYY',  
'NLS_DATE_LANGUAGE=GERMAN') FROM emp;
```

```
SELECT ename, TO_CHAR(sal,'9G999D99',  
'NLS_NUMERIC_CHARACTERS=``.``')  
FROM emp;
```

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### SQL Functions with NLS Parameters

SQL character functions support single-byte and multibyte characters.

Some SQL functions Oracle NLS parameters to be specified explicitly as part of their parameter list. Therefore SQL functions can override the behavior specified by the NLS environment.

**Examples Using NLS Parameters in SQL Functions**

```
SQL> SELECT TO_CHAR(hiredate, 'dd.mon.yyyy',
2   'NLS_DATE_LANGUAGE=GERMAN')
3   FROM emp
TO_CHAR(HIR
-----
17.dez.1980
20.feb.1981
22.feb.1981
02.apr.1981
28.sep.1981
01.mai.1981
09.jun.1981
19.apr.1987
17.nov.1981
08.sep.1981
23.mai.1987
03.dez.1981
03.dez.1981
23.jan.1982
14 rows selected.
```

### Examples Using NLS Parameters in SQL Functions (continued)

```
SQL> SELECT ename,  
2  TO_CHAR(sal, '99G999D99', 'NLS_NUMERIC_CHARACTERS='', '.')  
3  FROM emp;
```

ENAME	TO_CHAR(SA
-----	-----
SMITH	800,00
ALLEN	1.600,00
WARD	1.250,00
JONES	2.975,00
MARTIN	1.250,00
BLAKE	2.850,00
CLARK	2.450,00
SCOTT	3.000,00
KING	5.000,00
TURNER	1.500,00
ADAMS	1.100,00
JAMES	950,00
FORD	3.000,00
MILLER	1.300,00

14 rows selected.

**Examples Using NLS Parameters in SQL Functions (continued)**

The following SQL functions use NLS parameters:

Function	NLS Parameter
TO_DATE	NLS_DATE_LANGUAGE NLS_CALEDAR
TO_NUMBER	NLS_NUMERIC_CHARACTERS NLS_CURRENCY NLS_ISO_CURRENCY
TO_CHAR	NLS_DATE_LANGUAGE NLS_NUMERIC_CHARACTERS NLS_CURRENCY NLS_ISO_CURRENCY NLS_CALEDAR
NLS_UPPER, NLS_LOWER, NLS_INITCAP, NLSSORT	NLS_SORT

Several format mask elements have been defined for functions such as TO\_CHAR, TO\_DATE, and TO\_NUMBER.

**Number Format Mask Elements**

- “D” for decimal separator
- “G” for group (thousands) separator
- “L” for local currency symbol
- “C” for local ISO currency symbol
- “U” for the dual currency symbol, used for the euro

**Date Format Mask Elements**

- “RM, rm” for Roman month number
- “IW” for ISO week number
- “IYYY, IYY, IY,” and “I” for ISO year

## Linguistic Index Support

### National Language Support: Linguistic Index Support

- Linguistic indexing
- High performance with local sorting

```
CREATE INDEX nls_ename ON  
emp (NLSSORT(ename, 'NLS_SORT = German'));
```

- NLS\_COMP parameter for linguistic comparisons

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### Linguistic Indexing

Functional indexes, described in the lesson Indexes and Index-Organized Tables, can be specialized to create linguistically sorted indexes. The SQL function NLSSORT returns the string of bytes used to sort the first parameter in the given linguistic sorting sequence. In this example, an index is created on ENAME that is sorted according to German sorting order. This allows you to perform index-based queries on data sorted according to each languages rules.

### Linguistic Behavior of Comparison Operators

NLS\_COMP is a dynamic initialization parameter that controls how comparison operators such as <, >, and = handle linguistic ordering. When set to BINARY (the default), comparison is based on the binary value of the string. When set to ANSI, comparison operators use linguistic sorting sequences to determine the outcome of the operation according to the NLS\_SORT session parameter.

## Importing and Loading Data Using NLS

### Import and Loading Data Using NLS

- Data will be converted from NLS\_LANG to the database character set during the import
- LOADER:
  - Conventional: Data is converted into the session character set specified by NLS\_LANG.
  - DIRECT: Data is converted directly into the database character set.

Copyright © Oracle Corporation, 1999. All rights reserved.

ORACLE®

### NLS with Import and SQL\*Loader

During the import, the data is automatically converted to a character set for the specified session as determined by the NLS\_LANG parameter. After the data has been converted to the session character set, it is then converted to the database character set.

This means that NLS\_LANG has to be set to the character set of the export file.

SQL\*Loader also has the capability to convert data from the data file character set to the database character set.

When using conventional path, data is converted into the session character set specified by the NLS\_LANG parameter for that session.

On the direct path, data is converted directly into the database character set.

The control file of the SQL\*Loader shows how to interpret the data file.

The parameter character set tells what character set is used in each data file.

Example: `$sqlldr control=utllcase.ctl  
character set=WE8ISO9959P1`

## Obtaining Information About NLS Settings

### Obtaining Information About Character Sets

#### NLS\_DATABASE\_PARAMETERS:

- **PARAMETER**  
(NLS\_CHARACTERSET,  
NLS\_NCHAR\_CHARACTERSET)
- **VALUE**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

### Querying the Data Dictionary for NLS Information

View the database and the national character set with the following query:

```
SQL> SELECT parameter, value FROM nls_database_parameters
       2 WHERE parameter LIKE '%CHARACTERSET%';
```

PARAMETER	VALUE
-----	-----
NLS_CHARACTERSET	WE8ISO8859P1
NLS_NCHAR_CHARACTERSET	US7ASCII

2 rows selected.



## Obtaining Information About NLS Settings

- **NLS\_INSTANCE\_PARAMETERS:**
  - **PARAMETER** ( NLS initialization parameters that have been explicitly set)
  - **VALUE**
- **NLS\_SESSION\_PARAMETERS:**
  - **PARAMETER** ( NLS session parameters)
  - **VALUE**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Querying the Data Dictionary for NLS Information

This view displays only values for parameters that have been explicitly set in the `init<SID>.ora` file.

```
SQL> SELECT * FROM nls_instance_parameters;
```

PARAMETER	VALUE
-----------	-------

NLS_LANGUAGE	AMERICAN
--------------	----------

NLS_TERRITORY	AMERICA
---------------	---------

NLS_SORT	
----------	--

NLS_DATE_LANGUAGE	
-------------------	--

NLS_DATE_FORMAT	
-----------------	--

NLS_CURRENCY	
--------------	--

NLS_NUMERIC_CHARACTERS	
------------------------	--

NLS_ISO_CURRENCY	
------------------	--

8 rows selected.

## Querying the Data Dictionary for NLS Information (continued)

The following view shows session parameters.

```
SQL> SELECT * FROM nls_session_parameters;
```

PARAMETER	VALUE
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
NLS_CURRENCY	\$
NLS_ISO_CURRENCY	AMERICA
NLS_NUMERIC_CHARACTERS	.,
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	DD-MON-YY
NLS_DATE_LANGUAGE	AMERICAN
NLS_SORT	BINARY

9 rows selected.

## Obtaining Information About NLS Settings

- **NLS\_INSTANCE\_PARAMETERS:**
  - **PARAMETER** ( NLS initialization parameters that have been explicitly set)
  - **VALUE**
- **NLS\_SESSION\_PARAMETERS:**
  - **PARAMETER** ( NLS session parameters)
  - **VALUE**

Copyright © Oracle Corporation, 1999. All rights reserved. **ORACLE**

## Querying the Data Dictionary for NLS Information

List all valid values for NLS parameters.

```
SQL> SELECT * FROM v$nls_valid_values
      2 WHERE parameter='LANGUAGE' ;
```

PARAMETER	VALUE
LANGUAGE	AMERICAN
LANGUAGE	GERMAN
LANGUAGE	FRENCH
LANGUAGE	CANADIAN FRENCH
LANGUAGE	SPANISH
LANGUAGE	ITALIAN
LANGUAGE	DUTCH
LANGUAGE	SWEDISH
LANGUAGE	NORWEGIAN
LANGUAGE	DANISH
...	

## Querying the Data Dictionary for NLS Information (continued)

Display current values of NLS parameters.

```
SQL> SELECT * FROM v$nls_parameters;
```

PARAMETER	VALUE
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
NLS_CURRENCY	\$
NLS_ISO_CURRENCY	AMERICA
NLS_NUMERIC_CHARAC	. ,
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	DD-MON-YY
NLS_DATE_LANGUAGE	AMERICAN
NLS_CHARACTERSET	WE8ISO8859P1
NLS_SORT	BINARY

10 rows selected.

**Note:** Various views will contain a new column, CHARACTER\_SET\_NAME, which shows the name of the character set: CHAR\_CS for database character set and NCHAR\_CS for national character set.

For example, DBA\_TAB\_COLUMNS builds this column from COL\$.

## Summary

### Summary

In this lesson, you should have learned how to:

- Choose a database character set and a national character set for the database
- Use the various types of NLS parameters for the server, or the session

Copyright © Oracle Corporation, 1999. All rights reserved. ORACLE®

## Quick Reference

Context	Reference
Initialization parameters	NLS_LANGUAGE NLS_TERRITORY NLS_DATE_FORMAT NLS_DATE_LANGUAGE NLS_CURRENCY NLS_ISO_CURRENCY NLS_SORT NLS_NUMERIC_CHARACTERS NLS_CALENDAR
Dynamic performance views	V\$NLS_VALID_VALUES V\$NLS_PARAMETERS
Data dictionary views	NLS_DATABASE_PARAMETERS NLS_INSTANCE_PARAMETERS NLS_SESSION_PARAMETERS
Commands	ALTER SESSION SET
Packaged procedures and functions	DBMS_SESSION.SET_NLS

---

A

---

**Practices**

## Environment

Your instructor will provide you with information about the setup for the practices. Note that your environment may not need all of the information given below. Use this section to record the details for use during the practices.

### Client Setup

<b>Username</b>	
<b>Password</b>	
<b>Context</b>	
<b>Directory containing lab files</b>	
<b>Directory containing sample solutions</b>	
<b>Parameter file</b>	

### Server Setup

<b>Login</b>	
<b>Password</b>	
<b>Hostname/ IP Address</b>	
<b>Directory containing lab files</b>	
<b>Directory containing sample solutions</b>	
<b>Parameter file</b>	

### Oracle Environment

<b>Password for SYSDBA</b>	
<b>Password for SYS</b>	
<b>Password for SYSTEM</b>	
<b>Database Connect String</b>	



## Practice 1: Oracle Architectural Components

- 1 A user attempts to log on and receives the error message “ORA-01034 ORACLE not available.” Which of the following is the likely cause of the problem?
  - a The user supplied an invalid password.
  - b The user supplied an invalid username.
  - c The instance that the user is connecting to is not running.
  - d The Oracle version that the user is requesting is not installed.
- 2 A user executes an SQL command to update a row in the EMP table. Which process executes this statement?
  - a User process
  - b Server process
  - c DBWR
  - d LGWR
- 3 A user executes an SQL command to update a row in the EMP table. Where is the change made by the process identified in the previous question?
  - a Data files
  - b Database buffer cache
  - c Shared pool
  - d Parameter file
- 4 Which of the following files is used to authenticate privileged database users?
  - a Redo log file
  - b Control file
  - c Password file
  - d Archived log file
- 5 Which of the following files is not a part of the database?
  - a Redo log file
  - b Control file
  - c Password file
  - d Data file
- 6 Which of the following files store rollback segments?
  - a Redo log file
  - b Control file
  - c Password file
  - d Data file

- 7** Which of the following memory areas is not a part of the SGA?
- a** Database buffer cache
  - b** PGA
  - c** Redo log buffer
  - d** Shared pool
- 8** Which of the following memory areas is used to cache the data dictionary information?
- a** Database buffer cache
  - b** PGA
  - c** Redo log buffer
  - d** Shared pool
- 9** Which of the following stages are used to process a DML statement?
- a** Parse
  - b** Execute
  - c** Fetch
- 10** When a user executes a commit, in which of the following files are the changes recorded before the Oracle server returns a “Commit complete” message to the user?
- a** Redo log file
  - b** Control file
  - c** Password file
  - d** Data file

## Practice 2: Getting Started With Oracle

### Using SQL\*Plus

- 1 What is the size of the database buffer cache?
- 2 What is the size of the system global area?
- 3 List the first 9 rows of the OWNER, TABLE\_NAME, TABLESPACE\_NAME columns in the data dictionary view DBA\_TABLES and format the output. (The display of the first ten rows is sufficient.) Exit SQL\*Plus.
- 4 Start SQL\*Plus and run a script named `para.sql`, which spools all initialization parameters to the output file `para.lst`.
- 5 Run the script `scott.sql`.

## Practice 3: Managing an Oracle Instance

- 1 Identify the database name, instance name, and size of the database blocks.
- 2 List the name and size of the data files, online redo log files, and the name of the control files.
- 3 List the installed options.
- 4 Display the version numbers.
- 5 Display the maximum number of operating system user processes that can simultaneously connect to the instance.
- 6 Try to change the database block size. What happens?
- 7 List the default initialization parameter.
- 8 Open the database in read only mode. Connect as user `SCOTT` and add 10 percent to all salaries in the table `EMP`. What happens?  
Put the database back in read write mode.
- 9 Enable and verify timing in trace files dynamically.
- 10 Connect as user `SCOTT` and insert rows in the table `EMP`. Open second session and try to shut down the database transactional. What happens?
- 11 Ensure that there are at least two sessions open, one session as user `SCOTT` and one as user `SYS`. Enable the restricted session, verify this and ensure that only the database administrator `SYS` is connected.

**12** Examine the following sample of an alert file to identify if internal errors or exceptions have occurred.

```

Wed Jun 30 13:24:30 1999
Starting ORACLE instance (normal)
LICENSE_MAX_SESSION = 0
LICENSE_SESSIONS_WARNING = 0
LICENSE_MAX_USERS = 0
Starting up ORACLE RDBMS Version: 8.1.5.0.0.
System parameters with non-default values:
  processes                          = 60
  shared_pool_size                   = 3500000
  java_pool_size                     = 1000000
  control_files                      = $HOME/DATA/DISK1/control01.con
  db_block_buffers                    = 100
  db_block_size                      = 4096
  compatible                         = 8.1.5
  log_checkpoint_interval            = 10000
  log_checkpoint_timeout              = 1800
  db_files                           = 1024
  db_file_multiblock_read_count      = 8
  dml_locks                          = 200
  rollback_segments                  = sysrol
  db_domain                          = world
  global_names                       = TRUE
  distributed_transactions            = 10
  sort_area_size                     = 64000
  db_name                            = DB01
  job_queue_processes                = 2
  job_queue_interval                  = 10
  parallel_max_servers                = 12
  background_dump_dest                = $HOME/BDUMP
  user_dump_dest                      = $HOME/UDUMP
  max_dump_file_size                  = 10240
  core_dump_dest                      = $HOME/CDUMP
PMON started with pid=2
DBW0 started with pid=3
LGWR started with pid=4
CKPT started with pid=5
SMON started with pid=6
RECO started with pid=7
SNP0 started with pid=8
SNP1 started with pid=9
..
Corrupt block relative dba: 0x01c0003a file=7. blocknum=58.
Fractured block found during buffer read
Data in bad block - type:6. format:2. rdba:0x01c0003a
last change scn:0x0000.0000e9c5 seq:0xa0 flg:0x00
consistency value in tail 0x44c506a0
check value in block header: 0x0, check value not calculated
spare1:0x0, spare2:0x0, spare2:0x0

```

## Practice 4: Creating a Database

The directories specified for the files must be placed within `$HOME/DATA`.

**1** Create a password file using the following information:

- Password for `sys:oracle`
- Enable five privileged users

Ensure that Oracle can write to this file.

**2** Write a script for the creation of a database with the following configuration:

- Database name and instance name `DB<xx>`
- One control file named `control01.con` located in the directory `DISK5`
- Two redo log file groups each with one 150K member named `redo0101.log` and `redo0201.log` located in the directory `DISK6`
- The maximum number of five log file groups and five log file members for each group
- A 20M data file named `system01.dbf` and located in `DISK4` directory
- Maximum of 30 data files that can be created for the database
- A maximum number of 100 archived redo logs for automatic media recovery
- The Character set `WE8ISO8859P1`

The Trace file location should be in the `BDUMP` and `CDUMP` directory.

**3** Enable spooling to capture the errors and run the script.

**4** (Optional) Include the command “`spool <file_name>`” before running the command

**5** (Optional) After creation check the database state and ensure that the database files are created

**6** Try to display the names of the database users? What happens and why?

## **Practice 5: Creating Data Dictionary Views and Standard Packages**

- 1** Create the data dictionary views.
- 2** Use the data dictionary views to gather the following information:
  - a** What is the name and number of the rollback segments?
  - b** Identify the data file that makes up the SYSTEM tablespace.
  - c** How much free space is available in the database and how much is already used?
  - d** List the name and creation date of the database users.
- 3** Check which data dictionary tables are used to define the DBA\_USERS view.
- 4** Establish the usage of PL/SQL functionality. (Take a break as soon as the script is started.)
- 5** Check if any packages are invalid.

## Practice 6: Maintaining the Control File

- 1 Where is the existing control file located and what is the name?
- 2 Try to start the database without any control files. (You can simulate this by changing the name of the control file in the parameter file or just changing the name of the control file.) What happens?
- 3 Multiplex the existing control file using the directory `DISK2` and name the new control file `control02.con`. Make sure that the Oracle Server is able to write to the new control file. For example, on UNIX use the command `chmod 660`. Confirm that both control files are being used.
- 4 What is the initial sizing of the datafile section in your control file?



## Practice 7: Maintaining Redo Log Files

- 1** List the number and location of existing log files and display the number of redo log file groups and members your database has.
- 2** In which database mode is your database configured? Is archiving enabled?
- 3** Add a redo log member to each group in your database located on `DISK3`, using the following naming conventions:  
If group 1 has two existing files called `redo0101.log` and `redo0102.log`, then add a new member called `redo0103.log`. Verify the result.
- 4** Create a new redo log group located in the directory `DISK3`, `DISK4`, and `DISK5` and verify the existence of the new group.
- 5** Relocate the members `redo0103.log` and `redo0303.log` (see step 3) and locate them in the directory `DISK5` from `DISK3`.
- 6** Remove the redo log group created in step 4.
- 7** Resize all online redo log files to 1024K. (Since we cannot resize log files we have to add new logs and drop the old.)

## Practice 8: Managing Tablespaces and Data Files

- 1 Create permanent tablespaces with the following names and storage:
  - a DATA01 for tables with default storage
  - b DATA02 for large objects with default storage (Ensure that every used extent size in the tablespace is multiple of 100K.)
  - c INDX01 for indexes with the default storage (Enable automatic extension of 500K when more extents are required.)
  - d RONLY for read only tables with the default storage

Display the information from the data dictionary

Tablespace Name	Subdirectory	Data File Location (Size)
DATA01	DISK4	data01.dbf (2M)
DATA02	DISK5	data02.dbf (1M)
INDX01	DISK3	indx01.dbf (1M)
RONLY	DISK1	ronly.dbf (1M)

- 2 Allocate 500K more to the tablespace DATA02 and verify the result.
- 3 Relocate the INDX01 tablespace and move it to DISK6.
- 4 Turn RONLY tablespace to read only after creating a table in this tablespace. Attempt to create an additional table and drop the table. What happens and why?
- 5 Drop the tablespace RONLY and verify it.
- 6 Without shutting down the instance, change the SORT\_AREA\_SIZE to 2 kilobytes.
- 7 Open two connections to the database as user SYSTEM. Run srt\_dd.sql from one session and monitor the sort activity from another session. Query sort statistics and temporary segment information both during and after the completion of the script. Note the findings.
- 8 From one of the sessions, run asn\_tts.sql to prepare for the next part of the lab. This script ensures that TEMP tablespace will be used for the sorts made by the user SYSTEM, and will be covered under the lesson “Managing Users.” Connect as SYSTEM from one of the sessions and run srt\_dd.sql. From the other session monitor sort activity and statistics, as was done in question 3. Do you notice any difference?
- 9 Reset SORT\_AREA\_SIZE.

## Practice 9: Storage Structure and Relationships

- 1 As user `system`, run the script `cr_segs.sql` to create tables and indexes.
- 2 Identify the different types of segments in the database.
- 3 Write a query to check which segments are within five extents short of the maximum extents. Ignore the bootstrap segment. This query is useful in identifying any segments that are likely to generate errors during future data load.
- 4 Which files have space allocated for the EMP table?
- 5 (a) Write a query to obtain the File Number and Block Number of EMP table header.  
(b) Write a query that will accept a file number and block number as the input and return the name and type of the segment that uses the block. Test your query by supplying the file and block number of the EMP table header (obtained in the previous query. Use SQL\*Plus to run the query).
- 6 List the free space available by tablespace. The query should display the number of fragments, the total free space, and the largest free extent in each tablespace.
- 7 Run the script `cr_frgs.sql` in the LABS directory. Check if there are any adjacent free extents in the database. Coalesce them and verify again.
- 8 List segments that will generate errors because of lack of space when they try to allocate an additional extent.

## Practice 10: Managing Rollback Segments

Before starting this practice make sure you run the script `$HOME/LABS/alt_sysrol.sql` as user `SYSTEM`.

- 1 Connect as system (password = manager) and insert a record into the EMP table. Was the operation successful? Why, or why not?
- 2 You are going to be running an online order entry application on your database. The orders will be entered using 15 client stations, which have a very high volume of activity in the mornings. Create an appropriate number of rollback segments in the database. (Assume default sizing for the purpose of this exercise.)
- 3 Ensure that you can perform the INSERT in question 1. Test to check whether you are able to INSERT a record into the EMP table. Do not commit the INSERT.
- 4 Verify which rollback segments in the system are available for use by transactions.
- 5 Find which rollback segment is used by the transaction.
- 6 Invoke SQL\*Plus from the labs directory. Run the script `ins_emp.sql`. Using a separate session take the rollback segment tablespace offline.
- 7 (a) Shut down the instance, start up again, and query the rollback segment information in the data dictionary to check how many rollback segments are online.  
(b) Ensure that all rollback segments will be brought online whenever the database is opened in the future and restart the instance. Make sure that all the rollback segments are ONLINE.
- 8 Create a new rollback segment called DEMO\_RBS in the database with the following storage parameters and ensure that it can be used. This is essential for the next question:  
INITIAL = 10K  
NEXT = 10K  
OPTIMAL = 30K
- 9 Check the number of extents in DEMO\_RBS rollback segment. Log in as `system/manager` using SQL\*Plus and run the script `ext_rbs.sql`. Is there any change in the number of extents in the rollback segment?
- 10 Ensure that the rollback segment DEMO\_RBS is reduced to its optimal size and verify that it has reduced.
- 11 Run `ins_dept1.sql` connected as `SYSTEM`. Using a separate session, connect as `SYSTEM` and run `ins_emp3.sql`. Check if there is a blocking session.
- 12 Drop the DEMO\_RBS rollback segment.

## Practice 11: Managing Tables

- 1 You need to create the following tables for an order entry system that you are implementing now. The tables and the columns are shown below:

Table	Column	Data Type and Size
CUSTOMERS	CUST_CODE	VARCHAR2(3)
	NAME	VARCHAR2(50)
	REGION	VARCHAR2(5)
ORDERS	ORD_ID	NUMBER(3)
	ORD_DATE	DATE
	CUST_CODE	VARCHAR2(3)
	DATE_OF_DELY	DATE

You have been informed that in the table `ORDERS`, rows will be inserted without a value for `DATE_OF_DELY`, and it will be updated when the order is fulfilled. Log in as `system/manager` and create the tables using the appropriate block space utilization and tablespace settings. Use tablespace `DATA01`. You can use the default storage settings. Use the Table Wizard when creating table `CUSTOMERS`. Do not use the Table Wizard when creating tables `ORDERS`.

- 2 Run the script `ins_cord.sql` to insert rows into the tables.
- 3 Find which files and blocks contain the orders from the customer with `CUST_CODE=A04`.
- 4 Check the number of extents used by the table `ORDERS`.
- 5 Allocate an extent manually, with default size, for the table `ORDERS` and confirm that the extent has been added as specified.
- 6 (a) Drop the table `BIG_EMP`.  
(b) Create another table, `ORDERS2` as copy of the `ORDERS` table, but with `MINEXTENTS=10` and `PCTINCREASE=0`. Verify that the table has been created with the specified number of extents.
- 7 Truncate table `ORDERS` without releasing space and check the number of extents to verify extents have not been deallocated.
- 8 Alter the `ORDERS2` table to reduce `MINEXTENTS` to 4. Has there been a reduction in the number of extents?

- 9** Truncate the ORDERS2 table releasing space. How many extents does the table have now?
- 10** (a) Run the script `ins_ord2.sql` to insert some rows into ORDERS2 table.  
(b) Release unused space from ORDERS2 and check the number of extents. Has space been released from the table? Why, or why not?
- 11** (a) For the order entry application, you now need to add a PRODUCTS table, which has the following columns:

Column	Data Type and Size
PROD_CODE	NUMBER(6)
DESCRIPTION	VARCHAR2(30)
PRICE	NUMBER(8,2)

Create this table in tablespace DATA02 using uniform extent sizes of 10K.

- (b) Check the sizes of the extents for this table. What do you observe?

## Practice 12: Managing Indexes

- 1 You are considering creating indexes on the NAME and REGION columns of the CUSTOMERS table. What types of index are appropriate for the two columns? Create the indexes, naming them CUST\_NAME\_IDX and CUST\_REGION\_IDX, respectively, placing them in the appropriate tablespaces.
- 2 Move the CUST\_REGION\_IDX index to another tablespace.
- 3 Note the files and blocks used by the extents by CUST\_REGION\_IDX index.

File_id	Block_id	Blocks

- 4 Re-create the CUST\_REGION\_IDX index without dropping and re-creating it, and retaining it in the same tablespace as before. Does the new index use the same blocks that were used earlier?
- 5 (a) Using *system* account, run the script `cr_numb.sql` to create and populate the NUMBERS table.  
 (b) Query the table NUMBERS to find the number of distinct values in the two columns in the table.  
 (c) Using uniform extent sizes of 4K, create B-tree indexes NUMB\_OE\_IDX and NUMB\_NO\_IDX on the ODD\_EVEN and NO columns of the NUMBERS table, respectively, and check the total sizes of the indexes.

Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

(d) Once again, using uniform extent sizes of 4K, create bitmap indexes NUMB\_OE\_IDX and NUMB\_NO\_IDX on the ODD\_EVEN and NO columns of the NUMBERS table, respectively, and check the total sizes of the indexes.

Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

What can you conclude about the relationship between cardinality and sizes of the two types of indexes?



## Practice 13: Maintaining Data Integrity

- 1 Examine the script, `cr_cons.sql`. Run the script to create the constraints.
- 2 Query the data dictionary to:
  - (a) Check for constraints, whether they are deferrable, and their status.
  - (b) Check the names and types of indexes created to validate the constraints.
  - (c) Check which columns are used in the constraints created.
- 3 Insert two records with the following values into the `PRODUCTS` table:

PROD_CODE	DESCRIPTION	PRICE
100860	Ace Tennis Racket	36.20
100860	Ace Tennis Ball 3-Pack	2.40

- 4 Enable the unique constraint on the `PRODUCTS` table. Was it successful? Why, or why not?
- 5 (a) Ensure that new rows added to the table do not violate the constraint on the `PRODUCTS` table.
  - (b) Query the data dictionary to verify the effect of the change.
  - (c) Test that the constraint rejects inserts that violate the constraint by adding a row with the following values:  
 PROD\_CODE: **100860**  
 DESCRIPTION: **Yellow Jersey Bicycle Helmet**  
 PRICE: **30**
- 6 Take the necessary steps to identify existing constraint violations in the `PRODUCTS` table, modify product codes as needed and guarantee that all existing as well as new data do not violate the constraint. (Assume that the table has several thousands of rows and it is too time consuming to verify each row manually.)
- 7 Run the script `ins_ocus1.sql` to insert rows into the table. Were the inserts successful? Rollback the changes.
- 8 Now examine the script, `ins_ocus2.sql`. Notice that this script also performs the inserts in the same sequence. Run the script and check if it executes successfully.
- 9 Truncate the `CUSTOMERS` table. Was it successful? Why or why not?

## Practice 14: Loading Data

Use the account *system* for all the questions in this lab.

- 1 (a) Run the script `ins_item.sql` to insert data into the ITEMS table.  
(b) Create a table ITEMS2 as a copy of the ITEMS table.  
(c) Note down the last file/block number used by a row in the table currently \_\_\_\_\_.
- 2 (a) Delete all rows in ITEMS2 and insert a new row with the following values into the table:  
ORD\_ID: **200**  
PROD\_CODE: **200000**  
QTY: **20**  
(b) Check and note down the file/block number for the new row  
\_\_\_\_\_.
- 3 (a) Use direct-load insert to copy data into the ITEMS2 table from the ITEMS table.  
(b) Check the file/block numbers of the rows with ORD\_ID <> 200. What can you observe about the location of these rows?
- 4 Examine the scripts `ulcase1.sql`, `ulcase1ctl`, `ulcase2ctl` and `ulcase2.dat`. These are some of the standard loader demonstration files supplied with the demonstration Oracle8i Enterprise Edition. As user *system*, perform the following steps to try two load runs and get acquainted with using SQL\*Loader:  
(a) Run the script `ulcase1.sql` to create the DEPT and EMP tables.  
(b) Run SQL\*Loader to load data into the DEPT table using the control file `ulcase1ctl`. Examine the log file generated, and query the DEPT table to check the data loaded.  
(c) Run SQL\*Loader again to load data into the EMP table using the control file `ulcase2ctl`. Notice that this run uses an input data file to load data. Examine the log file generated, and query the EMP table to check the data loaded.
- 5 (a) Check the number of extents and total number of blocks in the ITEMS2 table.  
(b) Allocate an extent to the table manually, and note the number of extents and blocks now \_\_\_\_\_. Since the extent has just been created, the new extent is empty.

## Practice 15: Reorganizing Data

- 1 You want to reorganize the ITEMS2 table using export and import. Check the number and size of the extents in the ITEMS2 table before you drop the table and then do it again after importing the table. What can you infer about the behavior of export/import on space allocation?
- 2 You need to move several indexes from one tablespace to another. This practice uses one index to show how this can be done.
  - (a) Create an index named ITEM\_OID\_IDX, in the tablespace DATA01, on the ORD\_ID column of the ITEMS2 table.
  - (b) Using export and import, move the index to the tablespace INDX01.

## Practice 16: Managing Password Security

- 1 Enable password management by using the `utlpwdmg.sql` script.
- 2 Try to change the password of user `SCOTT` to `SCOTT`. What happens?
- 3 Make sure that the following applies to users assigned the `DEFAULT` profile:
  - After two login attempts, the account should be locked.
  - The password should expire after 30 days.
  - The same password should not be reuse again for at least one minute
  - The account should have grace period of five days to change an expired password.

Ensure that the requirements given have been implemented.

- 4 Log in to user `SYSTEM` supplying an invalid password. Try this twice, then log in again this time supplying the correct password.
- 5 Ensure the `SYSTEM` can reconnect.
- 6 Disable password checks for the `DEFAULT` profile.

## Practice 17: Managing Users

- 1 Create user `Bob` with a password of `ALONG`. Make sure that any objects and temporary segments created by Bob are not created in the system tablespace. Also, ensure that Bob can log in and create objects up to one megabyte in size in `DATA01` and `INDX01` tablespaces. If you are not using Oracle Enterprise Manager, run the script `bob.sql`.
- 2 (a) Create a user `Kay` with a password of `MARY`. Make sure that any objects and sort segments created by Kay are not created in the system tablespace.  
(b) Copy the `ORDERS2` table from `SYSTEM` schema to Kay's account. Call the new table `ORDERS`.
- 3 Display the information on Bob and Kay from the data dictionary.
- 4 From the data dictionary display the information on the amount of space that Bob can use in tablespaces.
- 5 (a) As user `Bob` change his temporary tablespace. What happens? Why?  
(b) As `Bob`, change his password to `SAM`.
- 6 As system, remove Bob's quota on his default tablespace.
- 7 Remove Kay's account from the database.
- 8 Bob has forgotten his password. Assign him a password of `OLINK` and require that Bob change his password the next time he logs on.

## Practice 18: Managing Privileges

- 1 As `system`, create user `Kay` and give her the capability to log on to the database and create objects in her schema.
- 2 (a) Connect as `Kay`, and create tables using the script `ulcase1.sql` to create the tables `EMP` and `DEPT`.  
(b) Connect as `system` and copy the data from `SYSTEM.EMP` to Kay's `EMP` table. Verify that records have been inserted.  
(c) As `system` give Bob the ability to select from Kay's `EMP` table. What happens and why?
- 3 Reconnect as `Kay` and give Bob the ability to select from Kay's `EMP` table. Also, enable Bob to give the select capability to other users. Examine the data dictionary views that record these actions.
- 4 Create user `Todd` with the capability to log on to the database
- 5 (a) As `Bob`, enable Todd to access Kay's `EMP` table. Give Bob the new password `sam`.  
(b) As `Kay`, remove Bob's privilege to read Kay's `EMP` table.  
(c) As `Todd`, query Kay's `EMP` table. What happens and why?
- 6 (a) Enable `Kay` to create tables in any schema. As `Kay`, create the table `DEPT` in Bob's schema as a copy of `KAY.DEPT`. What happened and why?  
(b) As `system`, examine the data dictionary view `DBA_TABLES` to check the result.
- 7 Enable Kay to start up and shut down the database without the ability to create a new database.

## Practice 19: Managing Roles

- 1 Examine the data dictionary view and list the system privileges of the RESOURCE role.
- 2 Create a role called DEV, which enables a user to create a table, create a view and select from Kay's EMP table.
- 3 (a) Assign the RESOURCE and DEV roles to Bob, but make only the RESOURCE role to be automatically enabled when he logs on.  
(b) Give Bob the ability to read all the data dictionary information.
- 4 Bob needs to check the rollback segments that are currently used by the instance. Connect as Bob and list the rollback segments used.
- 5 As `system`, try to create a view EMP\_VIEW on Kay's EMP table. What happens and why?

## Practice 20: Using National Language Support

- 1 Check the database and the national character set.
- 2 Which are valid values for the database character set.
- 3 Make sure that all dates in this session are displayed using a 4-digit year.
- 4 Define the NLS\_LANG on server and client side to enable 8-bit character encoding scheme. Run the script `nls.sql` connected as user `sys` and display the rows. Then export the table. Import the table to user `Bob` using the character set `US7ASCII`.  
Query the table; what happened and why? (optional)



---

B

---

**Hints**

## **Practice 1: Oracle Architectural Components**

There are no hints for this practice.

## Practice 2: Getting Started With Oracle

### Using SQL\*Plus

- 1 What is the size of the database buffer cache?
- 2 What is the size of the system global area?
- 3 List the columns OWNER, TABLE\_NAME, TABLESPACE\_NAME of the data dictionary view DBA\_TABLES and format the output. (The display of the first ten rows is sufficient.) Exit SQL\*Plus.
- 4 Start SQL\*Plus and run a batch script named `para.sql`, which spools all initialization parameters to the output file `para.lst`.  
**Hint:** Invoke `para.sql` from the operating system command line.
- 5 Run the script `scott.sql`.

## Practice 3: Managing an Oracle Instance

- 1 Identify the database name, instance name, and size of the database blocks.  
**Hint:** Query the dynamic performance views V\$DATABASE, V\$THREAD, and V\$PARAMETER to display the database and instance name and the size of the database blocks.
- 2 List the name and size of the data files, online redo log files, and the name of the control files.  
**Hint:** Query the dynamic performance views V\$DATAFILE, V\$LOGFILE, and V\$CONTROLFILE to display the data files, online redo log files, and the name of the control files.
- 3 Which options are installed?  
**Hint:** Query the dynamic performance views V\$OPTION to display the installed options.
- 4 Display the version number.  
**Hint:** Query the dynamic performance views V\$VERSION to display the version number.
- 5 Display the maximum number of operating system user processes that can simultaneously connect to the instance.  
**Hint:** Query the dynamic performance views V\$PARAMETER or use the SHOW PARAMETER command to display the maximum number of operating system user processes that can simultaneously connect to the instance.
- 6 Try to change the database block size. What happened?  
**Hint:** There is no hint for this question.
- 7 List the default initialization parameter.  
**Hint:** Query the dynamic performance views V\$PARAMETER to display the default initialization parameter.
- 8 Open the database in read only mode. Connect as user SCOTT and add 10 percent to all salaries in the table EMP. What happens? Put the database back in read write mode.  
**Hint:** There is no hint for this question.

- 9 Enable timing in trace files dynamically and verify it.

**Hint:** Use the ALTER SYSTEM command to enable timing in trace files dynamically and use the dynamic performance view V\$PARAMETER to verify the result.

- 10 Connect as user SCOTT and insert rows in the table EMP. Open second session and try to shut down the database transactional. What happens?

**Hint:** There is no hint for this question.

- 11 Ensure that there are at least two sessions open, one session as user SCOTT and one as user SYS. Enable the restricted session, verify this and ensure that only the database administrator SYS is connected.

**Hint**

- Use the ALTER SYSTEM command to enable the restricted session and query the dynamic performance views V\$INSTANCE to verify the result.
- Use the dynamic performance view V\$SESSION to see the values of the SID and SERIAL# column.
- Execute the ALTER SYSTEM KILL SESSION command to terminate sessions.

**12** Examine the following sample of an alert file to identify if internal errors or exceptions have occurred.

**Hint:** There is no hint for this question.

```

Wed Jun 30 13:24:30 1999
Starting ORACLE instance (normal)
LICENSE_MAX_SESSION = 0
LICENSE_SESSIONS_WARNING = 0
LICENSE_MAX_USERS = 0
Starting up ORACLE RDBMS Version: 8.1.5.0.0.
System parameters with non-default values:
  processes                = 60
  shared_pool_size         = 3500000
  java_pool_size           = 1000000
  control_files            = $HOME/DATA/DISK1/control01.con
  db_block_buffers         = 100
  db_block_size            = 4096
  compatible               = 8.1.5
  log_checkpoint_interval  = 10000
  log_checkpoint_timeout   = 1800
  db_files                 = 1024
  db_file_multiblock_read_count= 8
  dml_locks                = 200
  rollback_segments       = sysrol
  db_domain               = world
  global_names            = TRUE
  distributed_transactions = 10
  sort_area_size          = 64000
  db_name                 = DB01
  job_queue_processes     = 2
  job_queue_interval      = 10
  parallel_max_servers    = 12
  background_dump_dest    = $HOME/BDUMP
  user_dump_dest          = $HOME/UDUMP
  max_dump_file_size      = 10240
  core_dump_dest          = $HOME/CDUMP
PMON started with pid=2
DBW0 started with pid=3
LGWR started with pid=4
CKPT started with pid=5
SMON started with pid=6
RECO started with pid=7
SNP0 started with pid=8
SNP1 started with pid=9
..
Corrupt block relative dba: 0x01c0003a file=7. blocknum=58.
Fractured block found during buffer read
Data in bad block - type:6. format:2. rdba:0x01c0003a
last change scn:0x0000.0000e9c5 seq:0xa0 flg:0x00
consistency value in tail 0x44c506a0
check value in block header: 0x0, check value not calculated
spare1:0x0, spare2:0x0, spare3:0x0

```

## Practice 4: Creating a Database

The directories specified for the files must be placed within `$HOME/DATA`.

**1** Create a password file using the following information:

- Password for `sys:oracle`
- Enable five privileged users

Ensure that Oracle can write to this file.

**Hint:** Use the `orapwd` utility to create the password file and locate the password file in the `$ORACLE_HOME/dbs` directory.

**2** Write a script for the creation of a database with the following configuration:

- Database name and instance name `U<xx>`
- One control file named `control01.con` located in the directory `DISK5`
- Two redo log file groups each with one 150K member named `log1a.rdo` and `log2a.rdo` located in the directory `DISK6`
- The maximum number of five log file groups and five log file members for each group
- A 20M data file named `system01.dbf` and located in `DISK4` directory
- Maximum of 30 data files that can be created for the database
- A maximum number of 100 archived redo logs for automatic media recovery
- The Character set `WE8ISO8859P1`

The Trace file location should be in the `BDUMP` and `CDUMP` directory.

**Hint**

- Edit the parameter file.
- Start the instance.
- Generate the `CREATE DATABASE` command script.

**3** Enable spooling to capture the errors and run the script (optional): Include the command “`spool <file_name>`” before running the command.

**4** After creation check the database state and ensure that the database files are created (optional).

**Hint:** Query the dynamic performance views `V$DATABASE`, `V$THREAD`, `V$DATAFILE`, `V$LOGFILE`, and `V$CONTROLFILE` to check the database state and display the information about data files, online redo log files, and control files.

**5** Try to display the names of the database users? What happens and why? (optional)

**Hint:** There is no hint for this question.

## Practice 5: Creating Data Dictionary Views and Standard Packages

- 1 Create the data dictionary views.
- 2 Use the data dictionary views to gather the following information:
  - a What is the name and number of the rollback segments?  
**Hint:** Query the data dictionary view DBA\_ROLLBACK\_SEGS to show the name of the rollback segments.
  - b Identify the data file that makes up the SYSTEM tablespace.  
**Hint:** Query the data dictionary view DBA\_DATA\_FILES to identify the data files that make up SYSTEM tablespace.
  - c How much free space is available in the database and how much is already used?  
**Hint**
    - Query the data dictionary view DBA\_FREE\_SPACE to show how much free space is available in the database.
    - Query the data dictionary view DBA\_SEGMENTS to display how much space is already used.
  - d List the name and creation date of the database users.  
**Hint:** Query the data dictionary view dba\_users to list the name and the creation of the database users.
- 3 Check which data dictionary tables are used to define the DBA\_USERS view.  
**Hint:** Query the data dictionary view DBA\_VIEWS to display the view definition of the data dictionary view DBA\_USERS.
- 4 Establish the usage of PL/SQL functionality. (Take a break as soon as the script is started.)  
**Hint:** There is no hint for this question.
- 5 Check if any packages are invalid.  
**Hint:** Query the DBA\_OBJECTS view to obtain this information.



## Practice 6: Maintaining the Control File

- 1 Where is the existing control file located and what is the name?

**Hint:** Query the dynamic performance view `V$CONTROLFILE` or `V$PARAMETER`, or execute the `SHOW PARAMETER` command to display the name and the location of the control file.

- 2 Try to start the database without any control files (You can simulate this by changing the name of the control file in the parameter file or just changing the name of the control file.) What happens?

**Hint:** There is no hint for this question.

- 3 Multiplex the existing control file using the directory `DISK2` and name the new control file `control02.con`. Make sure that the Oracle server is able to write to the new control file. For example, on UNIX use the command `chmod 660`. Confirm that both control files are being used.

**Hint**

- Shut down the database.
  - Copy the existing control file to a new file with the name `control02.con` to the directory `DISK2`.
  - Use the command `chmod 660` on UNIX.
  - Modify the parameter file to include the new filename.
  - Start up the database.
  - Query the dynamic performance view `V$CONTROLFILE` or `V$PARAMETER` or use the `SHO PARAMETER` command to confirm both control files are being used.
- 4 What is the initial sizing of the data file section in your control file?
- Hint:** There is no hint for this question.

## Practice 7: Maintaining Redo Log Files

- 1 List the number and location of existing log files and display the number of redo log file groups and members your database has.

### Hint

- Query the dynamic performance views V\$LOGFILE to display the location of the existing log files.
- Use the dynamic performance view V\$LOG to display the number of redo log file groups and members.

- 2 In which database mode is your database configured? Is archiving enabled?

### Hint

- Query the dynamic performance views V\$DATABASE to show the database mode.
- Query the dynamic performance view V\$INSTANCE to verify if the automatic archiving is enabled.

- 3 Add a redo log member to each group in your database located on DISK3, using the following naming conventions:

If group 1 has two existing files called redo0101.log and redo0102.log, then add a new member called redo0103.log. Verify the result.

### Hint

- Execute the ALTER DATABASE ADD LOGFILE MEMBER command to add a redo log member to each group.
- Query the dynamic performance views V\$LOGFILE to verify the result.

- 4 Create a new redo log group located in the directory DISK3, DISK4, and DISK5 and verify the existence of the new group.

### Hint

- Execute the ALTER DATABASE ADD LOGFILE command to create a new group.
- Query the dynamic performance views V\$LOGFILE to display the name of the new members of the new group.
- Query the dynamic performance view V\$LOG to display the number of redo log file groups and members.

- 5** Relocate the members `redo0103.log` and `redo0303.log` (see step 3) and locate them in the directory `DISK5` from `DISK3`.

**Hint**

- Move the members to the directory `DISK5`.
  - Execute the `ALTER DATABASE RENAME FILE` command to rename the members.
  - Query the dynamic performance views `V$LOGFILE` to verify the result.
- 6** Remove the log group created in step 4.

**Hint**

- Execute the `ALTER DATABASE DROP LOGFILE GROUP` command to remove the log group.
  - Query the dynamic performance views `V$LOG` to verify the result.
- 7** Resize all online redo log files to 1024K. (Since we cannot resize logfiles we have to add new logs and drop the old.)

**Hint**

- Execute the `ALTER DATABASE ADD LOGFILE GROUP` command to add two new group with the size 1024K.
- Query the dynamic performance views `V$LOG` to check the active group.
- Execute the `ALTER SYSTEM SWITCH LOGFILE` command to force log switches and change the group stage to inactive.
- Execute the `ALTER DATABASE DROP LOGFILE` to remove the inactive groups.
- Query the dynamic performance view `V$LOG` to verify the result.

## Practice 8: Managing Tablespaces and Data Files

- 1 Create permanent tablespaces with the following names and storage:
  - a DATA01 for tables with default storage
  - b DATA02 for large objects with default storage (Ensure that every used extent size in the tablespace is multiple of 100K.)
  - c INDX01 for indexes with the default storage (Enable automatic extension of 500K when more extents are required.)
  - d RONLY for read only tables with the default storage

Display the information from the data dictionary.

### Hint

- Execute the CREATE TABLESPACE command to create the permanent tablespaces.
- Display the information from the data dictionary.
- Query the dynamic performance view DBA\_DATA\_FILES to verify the result.

Tablespace Name	Sub-directory	Data File Location (Size)
DATA01	DISK4	data01.dbf (2M)
DATA02	DISK5	data02.dbf (1M)
INDX01	DISK3	indx01.dbf (1M)
RONLY	DISK1	ronly.dbf (1M)

- 2 Allocate 500K more to the tablespace DATA02 and verify the result.

### Hint

- Use the ALTER DATABASE DATAFILE... RESIZE command to allocate another 500K.
- Query the dynamic performance view V\$DATAFILE to verify the result.

- 3 Relocate the INDX01 tablespace and move it to DISK6.

### Hint

- Take the tablespace INDX01 OFFLINE.
- Query the dynamic performance view V\$DATAFILE to verify the result.
- Execute the ALTER TABLESPACE RENAME command to rename the files.
- Take the tablespace INDX01 ONLINE.
- Query the dynamic performance views V\$DATAFILE to verify the result.

- 4 Turn RONLY tablespace to read only after creating a table in this tablespace. Attempt to create an additional table and drop the table. What happens and why?

**Hint:** There is no hint for this question.

- 5 Drop the tablespace RONLY and verify it.

**Hint**

- Execute the DROP TABLESPACE ... to remove the tablespace.
- Delete the operating-system files.
- Query the dynamic performance view V\$TABLESPACE to verify the result.

- 6 Without shutting down the instance, change the SORT\_AREA\_SIZE to 2 kilobytes.

**Hint:** SORT\_AREA\_SIZE is a dynamic parameter needing the DEFERRED option.

- 7 Open two connections to the database as user SYSTEM. Run srt\_dd.sql from one session and monitor the sort activity from another session. Query sort statistics and temporary segment information both during and after the completion of the script. Note the findings.

**Hint:** Sort segment information is available from V\$SORT\_SEGMENT and current sort activity from V\$SORT\_USAGE.

- 8 From one of the sessions, run asn\_tts.sql to prepare for the next part of the lab. This script ensures that TEMP tablespace will be used for the sorts made by the user SYSTEM, and will be covered under the lesson “Managing Users.” Connect as SYSTEM from one of the sessions and run srt\_dd.sql. From the other session monitor sort activity and statistics, as was done in question 3. Do you notice any difference?

- 9 Reset SORT\_AREA\_SIZE.

## Practice 9: Storage Structure and Relationships

- 1 As user `system`, run the script `cr_segs.sql` to create tables and indexes.
- 2 Identify the different types of segments in the database.
- 3 Write a query to check which segments are within five extents short of the maximum extents. Ignore the bootstrap segment. This query is useful in identifying any segments that are likely to generate errors during future data load.  
**Hint:** The columns `EXTENTS` and `MAX_EXTENTS` in the data dictionary view `DBA_SEGMENTS` contain the current number and the maximum number of extents.
- 4 Which files have space allocated for the EMP table?  
**Hint:** The views `DBA_EXTENTS` and `DBA_DATA_FILES` need to be joined to obtain this information.
- 5 (a) Write a query to obtain the File Number and Block Number of EMP table header.  
**Hint:** To find the header block of the EMP table, query the `DBA_SEGMENTS` view.  
(b) Write a query that will accept a file number and block number as the input and return the name and type of the segment that uses the block. Test your query by supplying the file and block number of the EMP table header (obtained in the previous query. Use SQL\*Plus to run the query.)  
**Hint:** Create a SQL\*Plus script to accept the relative file number and block number, and query the view `DBA_EXTENTS` to find the segment that uses the relative file and the block number entered..
- 6 List the free space available by tablespace. The query should display the number of fragments, the total free space, and the largest free extent in each tablespace.  
**Hint:** Query the `DBA_FREE_SPACE` view to get this information.

- 7** Run the script `cr_fragments.sql` in the LABS directory. Check if there are any adjacent free extents in the database. Coalesce them and verify again.

**Hint**

Execute the script `cr_fragments.sql`.

- Query the `DBA_FREE_SPACE_COALESCED` view for any tablespaces where `PERCENT_BLOCKS_COALESCED <> 100`.
  - Use the `ALTER TABLESPACE` command to coalesce free space in the tablespace.
  - Query the `DBA_FREE_SPACE_COALESCED` view again to check for any tablespaces where `PERCENT_BLOCKS_COALESCED <> 100`.
- 8** List segments that will generate errors because of lack of space when they try to allocate an additional extent.

**Hint:** You need to check if the biggest free extent for a tablespace from `DBA_FREE_SPACE` is not as large as the `NEXT_EXTENT` for any segment in the tablespace.

## Practice 10: Managing Rollback Segments

Before starting this practice make sure you run the script `$HOME/LABS/alt_sysrol.sql` as user `SYSTEM`.

- 1 Connect as system (password = manager) and insert a record into the EMP table. Was the operation successful? Why, or why not?
- 2 You are going to be running an online order entry application on your database. The orders will be entered using 15 client stations, which have a very high volume of activity in the mornings. Create appropriate number of rollback segments in the database. (Assume an default sizing for the purpose of this exercise.)  
**Hint:** Using a ratio of one rollback segment for every four transactions, you will need to create four rollback segments.
- 3 Ensure that you can perform the INSERT in question 1. Test to check whether you are able to INSERT a record into the EMP table. Do not commit the INSERT.  
**Hint:** At least one of the rollback segments needs to be ONLINE before you can successfully insert a row into the EMP table.
- 4 Verify which rollback segments in the system are available for use by transactions.  
**Hint:** This information can be obtained from `DBA_ROLLBACK_SEGS` view.
- 5 Find which rollback segment is used by the transaction.  
**Hint:** This information can be obtained by joining `V$ROLLSTAT` and `V$ROLLNAME` views.
- 6 Invoke SQL\*Plus from the labs directory. Run the script `ins_emp.sql`. Using a separate session take the rollback segment tablespace offline.  
**Hint**
  - The script starts another transaction that inserts a row into the EMP table. Since both transactions are using the rollback segments in this tablespace, you need to use the following steps.
    - First, take all rollback segments in the tablespace OFFLINE to prevent new transactions from using the rollback segments in the tablespace.
    - Now, identify the sessions that are using the rollback segments in the RBS tablespace.
    - Kill the sessions using rollback segments in the tablespace.
    - Take the RBS tablespace offline.



- 7 (a) Shut down the instance, start up again, and query the rollback segment information in the data dictionary to check how many rollback segments are online.

(b) Ensure that all rollback segments will be brought online whenever the database is opened in the future and restart the instance. Make sure that all the rollback segments are ONLINE.

**Hint:** This can be done using the `ROLLBACK_SEGEMENTS` initialization parameter. Also, the RBS tablespace must be brought ONLINE.

- 8 Create a new rollback segment called `DEMO_RBS` in the database with the following storage parameters and ensure that it can be used. This is essential for the next question:

`INITIAL = 10K`

`NEXT = 10K`

`OPTIMAL = 30K`

**Hint:** To ensure that the rollback segment can be used, it must be brought ONLINE after creation.

- 9 Check the number of extents in `DEMO_RBS` rollback segment. Log in as `system/manager` using `SQL*Plus` and run the script `ext_rbs.sql`. Is there any change in the number of extents in the rollback segment?

- 10 Ensure that the rollback segment `DEMO_RBS` is reduced to its optimal size and verify that it has reduced.

**Hint:** The `OPTIMAL` size can be verified from `V$ROLLSTAT`.

**Hint:** Use the `ALTER ROLLBACK SEGMENT` command to reduce the size and verify again.

- 11 Run `ins_dept1.sql` connected as `SYSTEM`. Using a separate session, connect as `SYSTEM` and run `ins_emp3.sql`. Check if there is a blocking session.

**Hint:** Determine blocking session from `V$ROLLSTAT`.

- 12 Drop the `DEMO_RBS` rollback segment.

**Hint:** The rollback must be taken offline before it is dropped. You may need to ensure that no transactions are using the rollback segment.

## Practice 11: Managing Tables

- 1 You need to create the following tables for an order entry system that you are implementing now. The tables and the columns are shown below:

Table	Column	Data Type and Size
CUSTOMERS	CUST_CODE	VARCHAR2(3)
	NAME	VARCHAR2(50)
	REGION	VARCHAR2(5)
ORDERS	ORD_ID	NUMBER(3)
	ORD_DATE	DATE
	CUST_CODE	VARCHAR2(3)
	DATE_OF_DELY	DATE

You have been informed that in the table ORDERS, rows will be inserted without a value for DATE\_OF\_DELY, and it will be updated when the order is fulfilled. Log in as `system/manager` and create the tables using the appropriate block space utilization and tablespace settings. Use tablespace DATA01. You can use the default storage settings. Use the Table Wizard when creating table CUSTOMERS. Do not use the Table Wizard when creating tables ORDERS.

**Hint:** PCTFREE has to be set carefully for the ORDERS table as rows in this table are likely to grow after updates.

- 2 Run the script `ins_cord.sql` to insert rows into the tables.
- 3 Find which files and blocks contain the orders from the customer with CUST\_CODE=A04.  
**Hint:** You need to use the DBMS\_ROWID package to translate the ROWID. Since the database only has a few files, the relative file number and absolute file numbers are the same.
- 4 Check the number of extents used by the table ORDERS.
- 5 Allocate an extent manually, with default size, for the table ORDERS and confirm that the extent has been added as specified.

- 6 (a) Drop the table BIG\_EMP.
- (b) Create another table, ORDERS2 as copy of the ORDERS table, but with MINEXTENTS=10 and PCTINCREASE=0. Verify that the table has been created with the specified number of extents.
- 7 Truncate table ORDERS without releasing space and check the number of extents to verify extents have not been deallocated.
- 8 Alter the ORDERS2 table to reduce MINEXTENTS to 4. Has there been a reduction in the number of extents?
- 9 Truncate the ORDERS2 table releasing space. How many extents does the table have now?
- 10 (a) Run the script `ins_ord2.sql` to insert some rows into ORDERS2 table.
- (b) Release unused space from ORDERS2 and check the number of extents. Has space been released from the table? Why, or why not?
- 11 (a) For the order entry application, you now need to add a PRODUCTS table, which has the following columns:

Column	Data Type and Size
PROD_CODE	NUMBER(6)
DESCRIPTION	VARCHAR2(30)
PRICE	NUMBER(8,2)

Create this table in tablespace DATA02 using uniform extent sizes of 10K.

(b) Check the sizes of the extents for this table. What do you observe?

**Hint:** Check the DBA\_EXTENTS view to get the size.

## Practice 12: Managing Indexes

- 1 You are considering creating indexes on the NAME and REGION columns of the CUSTOMERS table. What types of index are appropriate for the two columns? Create the indexes, naming them. CUST\_NAME\_IDX and CUST\_REGION\_IDX, respectively, placing them in the appropriate tablespaces.

**Hint:** A B-tree index is suitable for a column with many distinct values, and a bitmap index is suitable for columns with only a few distinct values.

- 2 Move the CUST\_REGION\_IDX index to another tablespace.

**Hint:** The index can be rebuilt specifying a different tablespace.

- 3 Note the files and blocks used by the extents by CUST\_REGION\_IDX index.

File_id	Block_id	Blocks

**Hint:** Use the view DBA\_EXTENTS to get this information.

- 4 Re-create the CUST\_REGION\_IDX index without dropping and re-creating it, and retaining it in the same tablespace as before. Does the new index use the same blocks that were used earlier?

**Hint:** Rebuild the index.

- 5 (a) Using `system` account, run the script `cr_num.sql` to create and populate the `NUMBERS` table.
- (b) Query the table `NUMBERS` to find the number of distinct values in the two columns in the table.
- (c) Using uniform extent sizes of 4K, create B-tree indexes `NUMB_OE_IDX` and `NUMB_NO_IDX` on the `ODD_EVEN` and `NO` columns of the `NUMBERS` table, respectively, and check the total sizes of the indexes.

Index	Blocks
<code>NUMB_OE_IDX</code>	
<code>NUMB_NO_IDX</code>	

### Hint

- Use `PCTINCREASE=0` to create extents of equal size.
- Check the total blocks allocated to the extents from `DBA_SEGMENTS`.

- (d) Once again, using uniform extent sizes of 4K, create bitmap indexes `NUMB_OE_IDX` and `NUMB_NO_IDX` on the `ODD_EVEN` and `NO` columns of the `NUMBERS` table, respectively, and check the total sizes of the indexes.

Index	Blocks
<code>NUMB_OE_IDX</code>	
<code>NUMB_NO_IDX</code>	

What can you conclude about the relationship between cardinality and sizes of the two types of indexes?

**Hint:** The existing indexes need to be dropped before creating the new indexes.

**Hint:** Now reexecute the query to check the sizes of the indexes.

## Practice 13: Maintaining Data Integrity

1 Examine the script, `cr_cons.sql`. Run the script to create the constraints.

2 Query the data dictionary to:

(a) Check for constraints, whether they are deferrable, and their status.

**Hint:** Use the `DBA_CONSTRAINTS` view to get this information.

(b) Check the names and types of indexes created to validate the constraints.

**Hint:** The indexes are only created for primary key and unique constraints and have the same name as the constraints.

(c) Check which columns are used in the constraints created.

**Hint:** This information may be obtained from `DBA_CONS_COLUMNS`.

3 Insert two records with the following values into the `PRODUCTS` table:

PROD_CODE	DESCRIPTION	PRICE
100860	Ace Tennis Racket	36.20
100860	Ace Tennis Ball 3-Pack	2.40

4 Enable the unique constraint on the `PRODUCTS` table. Was it successful? Why, or why not?

5 (a) Ensure that new rows added to the table do not violate the constraint on the `PRODUCTS` table.

**Hint:** This can be done by enabling the constraint `NOVALIDATE`.

(b) Query the data dictionary to verify the effect of the change.

(c) Test that the constraint rejects inserts that violate the constraint by adding a row with the following values:

**PROD\_CODE: 100860**

**DESCRIPTION: Yellow Jersey Bicycle Helmet**

**PRICE: 30**

---

## Practice 14: Loading Data

Use the account `system` for all the questions in this lab.

- 1 (a) Run the script `ins_item.sql` to insert data into the `ITEMS` table  
(b) Create a table `ITEMS2` as a copy of the `ITEMS` table.  
(b) Note down the last file/block number used by a row in the table currently \_\_\_\_\_.  
**Hint:** Query the ROWIDs using the functions in the `DBMS_ROWID` package.
- 2 (a) Delete all rows in `ITEMS2` and insert a new row with the following values into the table:  
ORD\_ID: **200**  
PROD\_CODE: **200000**  
QTY: **20**  
(b) Check and note down the file/block number for the new row  
\_\_\_\_\_.
- 3 (a) Use direct-load insert to copy data into the `ITEMS2` table from the `ITEMS` table.  
(b) Check the file/block numbers of the rows with `ORD_ID <> 200`. What can you observe about the location of these rows?
- 4 Examine the scripts `ulcase1.sql`, `ulcase1.ct1`, `ulcase2.ct1` and `ulcase2.dat`. These are some of the standard loader demonstration files supplied with the demonstration Oracle8i Enterprise Edition. As user `system`, perform the following steps to try two load runs and get acquainted with using SQL\*Loader:  
(a) Run the script `ulcase1.sql` to create the `DEPT` and `EMP` tables.  
(b) Run SQL\*Loader to load data into the `DEPT` table using the control file `ulcase1.ct1`. Examine the log file generated, and query the `DEPT` table to check the data loaded.  
(c) Run SQL\*Loader, again, to load data into the `EMP` table using the control file `ulcase2.ct1`. Notice that this run uses an input data file to load data. Examine the log file generated, and query the `EMP` table to check the data loaded.

- 5** (a) Check the number of extents and total number of blocks in the ITEMS2 table.

**Hint:** Query the DBA\_SEGMENTS view.

- (b) Allocate an extent to the table manually, and note the number of extents and blocks now \_\_\_\_\_. Since the extent has just been created, the new extent is empty.



## Practice 15: Reorganizing Data

- 1 You want to reorganize the ITEMS2 table using export and import. Check the number and size of the extents in the ITEMS2 table before you drop the table and then do it again after importing the table. What can you infer about the behavior of export/import on space allocation?

**Hint:** Perform the following steps:

- Use table level export to do this.
  - Drop the ITEMS2 table.
  - Import the ITEMS2 table.
  - Check the number and size of the extents in the ITEMS2 table using DBA\_SEGMENTS.
- 2 You need to move several indexes from one tablespace to another. This practice uses one index to show how this can be done.
    - (a) Create an index named ITEM\_OID\_IDX, in the tablespace DATA01, on the ORD\_ID column of the ITEMS2 table.
    - (b) Using export and import, move the index to the tablespace INDX01.

**Hint:** Use the following steps.

- Export the table with indexes, but without rows.
- Drop the index.
- Import from the output of the previous export and create a script to build the index.
- Edit the index file created in the previous step to change the tablespace to INDX01 and run the script to re-create the index.

## Practice 16: Managing Password Security

- 1 Enable password management by using the `utlpwdmg.sql` script.

**Hint:** There is no hint for this question.

- 2 Try to change the password of user `SCOTT` to `SCOTT`. What happens?

**Hint:** There is no hint for this question.

- 3 Make sure that the following applies to users assigned the `DEFAULT` profile:

- After two login attempts, the account should be locked.
- The password should expire after 30 days.
- The same password should not be reuse again for at least oneminute
- The account should have grace period of five days to change an expired password.

Ensure that the requirements given have been implemented.

**Hint**

- Use the `ALTER PROFILE` command to change the default profile limits.
- Query the data dictionary views `DBA_PROFILES` to verify the result.

- 4 Log in to user `SYSTEM` supplying an invalid password. Try this twice, then log in again this time supplying the correct password.

**Hint:** There is no hint for this question.

- 5 Ensure the `SYSTEM` can reconnect.

**Hint:** Execute the `ALTER USER` command to “unlock” the `SYSTEM` account.

- 6 Disable password checks for the `DEFAULT` profile.

**Hint:** Execute the `ALTER PROFILE` command to disable the password checks.

## Practice 17: Managing Users

- 1 Create user `Bob` with a password of `ALONG`. Make sure that any objects and temporary segments created by `Bob` are not created in the system tablespace. Also, ensure that `Bob` can log in and create objects up to one megabyte in size in `DATA01` and `INDX01` tablespaces. If you are not using Oracle Enterprise Manager, run the script `bob.sql`.  
**Hint:** Ensure that the temporary tablespace is assigned.
- 2 (a) Create a user `Kay` with a password of `MARY`. Make sure that any objects and sort segments created by `Kay` are not created in the system tablespace.  
(b) Copy the `ORDERS2` table from `SYSTEM` schema to `Kay`'s account. Call the new table `ORDERS`.  
**Hint:** `Kay` needs a quota on her default tablespace before objects can be created in her schema.
- 3 Display the information on `Bob` and `Kay` from the data dictionary.  
**Hint:** This can be obtained by querying `DBA_USERS`.
- 4 From the data dictionary display the information on the amount of space that `Bob` can use in tablespaces.  
**Hint:** This can be obtained by querying `DBA_TS_QUOTAS`.
- 5 (a) As user `Bob` change his temporary tablespace. What happens? Why?  
(b) As `Bob`, change his password to `SAM`.
- 6 As system, remove `Bob`'s quota on his default tablespace.
- 7 Remove `Kay`'s account from the database.  
**Hint:** Since `Kay` owns tables, you need to use the `CASCADE` option.
- 8 `Bob` has forgotten his password. Assign him a password of `OLINK` and require that `Bob` change his password the next time he logs on.

## Practice 18: Managing Privileges

- 1 As system, create user `KAY` and give her the capability to log on to the database and create objects in her schema.

**Hint:** Kay needs the `CREATE SESSION` and `CREATE TABLE` privileges.

- 2 (a) Connect as `KAY`, and create tables using the script `ulcase1.sql` to create the tables `EMP` and `DEPT`.

(b) Connect as `system` and copy the data from `SYSTEM.EMP` to Kay's `EMP` table. Verify that records have been inserted.

(c) As `system` give Bob the ability to select from Kay's `EMP` table. What happens and why?

- 3 Reconnect as `KAY` and give Bob the ability to select from Kay's `EMP` table. Also, enable Bob to give the select capability to other users. Examine the data dictionary views that record these actions.

**Hint:** Query the view `DBA_TAB_PRIVS` to see the privileges.

- 4 Create user `Todd` with the capability to log on to the database

- 5 (a) As `Bob`, enable Todd to access Kay's `EMP` table. Give Bob the new password `sam`.

(b) As `KAY`, remove Bob's privilege to read Kay's `EMP` table.

(c) As `Todd`, query Kay's `EMP` table. What happens and why?

- 6 (a) Enable Kay to create tables in any schema. As `KAY`, create the table `DEPT` in Bob's schema as a copy of `KAY.DEPT`. What happened and why?

(b) As `system`, examine the data dictionary view `DBA_TABLES` to check the result.

- 7 Enable `KAY` to start up and shutd own the database without the ability to create a new database.

**Hint:** Give Kay the `SYSOPER` privilege.

## Practice 19: Managing Roles

- 1 Examine the data dictionary view and list the system privileges of the RESOURCE role.  
**Hint:** This information is available from DBA\_SYS\_PRIVS.
- 2 Create a role called DEV, which enables a user to create a table, create a view and select from Kay's EMP table.
- 3 (a) Assign the RESOURCE and DEV roles to Bob, but make only the RESOURCE role to be automatically enabled when he logs on.  
**Hint:** Use the ALTER USER command to specify default role.  
(b) Give Bob the ability to read all the data dictionary information.  
**Hint:** Assign the SELECT\_CATALOG\_ROLE. to Bob.
- 4 Bob needs to check the rollback segments that are currently used by the instance. Connect as Bob and list the rollback segments used.  
**Hint:** Bob needs the SELECT\_CATALOG\_ROLE to check the rollback segments, but since this is not one of his default roles, it must be enabled first.
- 5 As system, try to create a view EMP\_VIEW on Kay's EMP table. What happens and why?

## Practice 20: Using National Language Support

- 1 Check the database and the national character set.

**Hint:** Query the data dictionary view

NLS\_DATABASE\_PARAMETERS to check the character sets.

- 2 Which are valid values for the database character set.

**Hint:** Query the data dictionary view V\$NLS\_VALID\_VALUES to list the valid values.

- 3 Make sure that all dates in this session are displayed using a 4-digit year

**Hint:** Set the parameter NLS\_DATE\_FORMAT..

- 4 Define the NLS\_LANG on server and client side to enable 8-bit character encoding scheme. Run the script `nls.sql` connected as user `SYS` and display the rows. Then export the table. Import the table to user `Bob` using the character set `US7ASCII`.

Query the table; what happened and why? (optional)

---

C

---

**Practice Solutions  
for SQL\*Plus**

## Practice 1 Solutions

- 1 A user attempts to log on and receives the error message “ORA-01034 ORACLE not available.” Which of the following is the likely cause of the problem?
- a The user supplied an invalid password.
  - b The user supplied an invalid username.
  - c The instance that the user is connecting to is not running.
  - d The Oracle version that the user is requesting is not—installed.

**Answer: C**

- 2 A user executes a SQL command to update a row in the EMP table. Which process executes this statement?
- a User process
  - b Server process
  - c DBWR
  - d LGWR

**Answer: B**

- 3 A user executes a SQL command to update a row in the EMP table. Where is the change made by the process identified in the previous question?
- a Data files
  - b Database buffer cache
  - c Shared pool
  - d Parameter file

**Answer: B**

- 4 Which of the following files is used to authenticate privileged database users?
- a Redo log file
  - b Control file
  - c Password file
  - d Archived log file

**Answer: C**

- 5 Which of the following files is not a part of the database?
- a Redo log file
  - b Control file
  - c Password file
  - d Data file

**Answer: C**



**6** Which of the following files store rollback segments?

- a** Redo log file
- b** Control file
- c** Password file
- d** Data file

**Answer: D**

**7** Which of the following memory areas is not a part of the SGA?

- a** Database buffer cache
- b** PGA
- c** Redo log buffer
- d** Shared pool

**Answer: B**

**8** Which of the following memory areas is used to cache the data dictionary information?

- a** Database buffer cache
- b** PGA
- c** Redo log buffer
- d** Shared pool

**Answer: D**

**9** Which of the following stages are used to process a DML statement?

- a** Parse
- b** Execute
- c** Fetch

**Answer: A, B**

**10** When a user executes a commit, in which of the following files are the changes recorded before the Oracle server returns a “Commit complete” message to the user?

- a** Redo log file
- b** Control file
- c** Password file
- d** Data file

**Answer: A**

## Practice 2 Solutions

### Using SQL\*Plus

#### 1 What is the size of the database buffer cache?

```
SQL> connect system/manager@db01
Connected.

SQL> sho sga

Total System Global Area      6315408 bytes
Fixed Size                    64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                   532480 bytes
```

#### 2 What is the size of the system global area?

```
SQL> sho sga

Total System Global Area      6315408 bytes
Fixed Size                    64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                   532480 bytes
```

- 3 List the first 9 rows of the OWNER, TABLE\_NAME, and TABLESPACE\_NAME columns in the data dictionary view DBA\_TABLES, and format the output. (The display of the first ten rows is sufficient.) Exit SQL\*Plus.

```
SQL> COL owner FORMAT a5
SQL> COL table_name FORMAT a15
SQL> COL tablespace_name FORMAT a15
SQL> SELECT owner, table_name, tablespace_name
  2   FROM dba_tables
  3  WHERE rownum < 10;
```

OWNER	TABLE_NAME	TABLESPACE_NAME
SYS	FILE\$	SYSTEM
SYS	BOOTSTRAP\$	SYSTEM
SYS	SEG\$	SYSTEM
SYS	ICOL\$	SYSTEM
SYS	IND\$	SYSTEM
SYS	CDEF\$	SYSTEM
SYS	OBJ\$	SYSTEM
SYS	PROXY\$	SYSTEM
SYS	CCOL\$	SYSTEM

```
SQL> exit
Disconnected from Oracle8i Enterprise Edition Release 8.1.5.0.0 ...
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production
```

- 4 Start SQL\*Plus and run a script named para.sql, which spools all initialization parameters to the output file para.lst.

**Hint:** Invoke para.sql from the operating system command line.

```
$ sqlplus system/manager@db01 @$HOME/LABS/para.sql

SQL*Plus: Release 8.1.5.0.0 - Production on Thu May 6 15:37:15 1999

(c) Copyright 1999, Oracle Corporation. All Rights Reserved.

Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

Connected.
```

NAME	TYPE	VALUE
...		
db_name	string	db01
...		

**5** Run the script `scott.sql`.

```
SQL> @$HOME/LABS/scott
Connected.

Grant succeeded

Connected.

Table created.

Table created.

1 row created.
....
1 row created.

Commit completed.
```

## Practice 3 Solutions

- 1 Identify the database name, instance name, and size of the database blocks.

**Hint:** Query the dynamic performance views V\$DATABASE, V\$THREAD, and V\$PARAMETER.

```
SQL> CONNECT system/manager@db01
Connected.
SQL> SELECT name FROM v$database;
NAME
-----
DB01

SQL> SELECT instance FROM v$thread;
INSTANCE
-----
DB01

SQL> SELECT value
  2 FROM v$parameter
  3 WHERE name = 'db_block_size';
VALUE
-----
4096
```

- 2** List the name and size of the data files, online redo log files, and the name of the control files.

**Hint:** Query the dynamic performance views V\$DATAFILE, V\$LOGFILE, and V\$CONTROLFILE.

```
SQL> SELECT name FROM v$datafile;
NAME
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf
/oracle/hrasmuss/DATA/DISK3/data01.dbf
/oracle/hrasmuss/DATA/DISK2/temp01.dbf
/oracle/hrasmuss/DATA/DISK2/indx01.dbf
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf
/oracle/hrasmuss/DATA/DISK1/query01.dbf

SQL> SELECT member FROM v$logfile;
MEMBER
-----
/oracle/hrasmuss/DATA/DISK3/redo0101.log
/oracle/hrasmuss/DATA/DISK4/redo0102.log
/oracle/hrasmuss/DATA/DISK3/redo0201.log
/oracle/hrasmuss/DATA/DISK4/redo0202.log

SQL> SELECT name FROM v$controlfile;
NAME
-----
/oracle/hrasmuss/DATA/DISK1/control01.ctl
```

**3** List the installed options.**Hint:** Query the dynamic performance view V\$OPTION.

```
SQL> COL paramter FORMAT a40
SQL> COL status FORMAT a10
SQL> SELECT * from v$option;
```

PARAMETER	VALUE
Partitioning	TRUE
Objects	TRUE
Parallel Server	FALSE
Advanced replication	TRUE
Bit-mapped indexes	TRUE
Connection multiplexing	TRUE
Connection pooling	TRUE
Database queuing	TRUE
Incremental backup and recovery	TRUE
Instead-of triggers	TRUE
Parallel backup and recovery	TRUE
Parallel execution	TRUE
Parallel load	TRUE
Point-in-time tablespace recovery	TRUE
Fine-grained access control	TRUE
N-Tier authentication/authorization	TRUE
Function-based indexes	TRUE
Plan Stability	TRUE
Online Index Build	TRUE
Coalesce Index	TRUE
Managed Standby	TRUE
Materialized view rewrite	TRUE
Materialized view warehouse refresh	TRUE
Database resource manager	TRUE
Spatial	TRUE
Visual Information Retrieval	TRUE
Export transportable tablespaces	TRUE
Transparent Application Failover	TRUE
Fast-Start Fault Recovery	TRUE
Sample Scan	TRUE
Duplexed backups	TRUE
Java	TRUE

**4** Display the version numbers.

**Hint:** Query the dynamic performance view V\$VERSION.

```
SQL> SELECT * FROM v$version;
BANNER
-----
Oracle8i Enterprise Edition Release 8.1.5.0.0 - Production
PL/SQL Release 8.1.5.0.0 - Production
CORE Version 8.1.3.0.0 - Production
TNS for Solaris: Version 8.1.5.0.0 - Production
NLSRTL Version 3.4.0.0.0 - Production
```

**5** Display the maximum number of operating system user processes that can simultaneously connect to the instance.

**Hint:** Query the dynamic performance view V\$PARAMETER or use the SHOW PARAMETER.

```
SQL> SELECT value
  2 FROM v$parameter
  3 WHERE name = 'processes';
VALUE
-----
60
```



**6** Try to change the database block size. What happens?

```
SQL> connect sys/oracle@db01 as sysdba

Connected.

SQL> shutdown immediate;

Database closed.
Database dismounted.
ORACLE instance shut down.

SQL> exit

a Edit initdb01.ora, and add the following line.
DB_BLOCK_SIZE=8192

b Then startup the database as follows:

SQL> connect sys/oracle@db01 as sysdba

Connected.

SQL> startup pfile=$HOME/initdb01.ora

ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                     64912 bytes
Variable Size                  5308416 bytes
Database Buffers               409600 bytes
Redo Buffers                   532480 bytes
ORA-00209: control file blocksize mismatch, check alert log for
more info

SQL> shutdown immediate;
SQL> exit

c Edit init<SID>.ora file to undo the db_block_size change made above.

SQL> connect sys/oracle as sysdba
SQL> startup pfile=$HOME/initDB01.ora
```

**7** List the default initialization parameter.**Hint:** Query the dynamic performance view V\$PARAMETER.

```
SQL> SELECT name
       2 FROM v$parameter
       3 WHERE isdefault='TRUE';
NAME
-----
spin_count
sessions
...
session_max_open_files
aq_tm_processes
hs_autoregister

174 rows selected.
```

**8** Open the database in read-only mode. Connect as user SCOTT and add 10 percent to all salaries in the table EMP. What happens?

**a** Put the database back in read-write mode.

```
SQL> connect sys/oracle@db01 as sysdba
Connected.
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> exit
```

**b** Then Startup the database as follows:

```
SQL> connect sys/oracle@db01 as sysdba
Connected
SQL> startup pfile=$HOME/initDB01.ora mount;
ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                    64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                  532480 bytes
Database mounted.
```

```
SQL> alter database open read only;
SQL> Database altered.
SQL> connect scott/tiger@db01
SQL> update emp set sal=sal*1.1;
0 rows updated.
SQL> connect sys/oracle@db01 as sysdba
Connected
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> exit
```

**c** Then start up the database as follows:

```
SQL> connect sys/oracle@db01 as sysdba
Connected
SQL> startup pfile=$HOME/initDB01.ora;
ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                    64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                  532480 bytes
Database mounted.
Database opened.
```

**9** Enable and verify timing in trace files dynamically.

**Hint:** Use the ALTER SYSTEM command. Use the dynamic performance view V\$PARAMETER to verify the result.

```
SQL> SELECT value, isdefault, ismodified
  2 FROM v$parameter
  3 WHERE name = 'timed_statistics';
```

VALUE	ISDEFAULT	ISMODIFIED
FALSE	TRUE	FALSE

1 row selected.

```
SQL> ALTER SYSTEM SET timed_statistics=true;
System altered.
```

```
SQL> SELECT value, isdefault, ismodified
  2 FROM v$parameter
  3 WHERE name = 'timed_statistics';
```

VALUE	ISDEFAULT	ISMODIFIED
TRUE	TRUE	SYSTEM_MOD

**10** Connect as user SCOTT and insert rows in the table EMP. Open a second session and try to shut down the database transactional. What happens?

**Hint:** There is no hint for this question.

```
SQL> connect scott/tiger@db01
Connected.
SQL> INSERT INTO emp (empno, ename, deptno)
  2 VALUES (1, 'Vijay', 10);
1 row created.
a In the second session,
SQL> connect sys/oracle@db01 as sysdba
SQL> shutdown transactional
```

The Oracle server waits for SCOTT's transaction to end before shutting down. Wait for the instance to shut down at the second session. Then bring it back up.

```
SQL> startup pfile=$HOME/initDB01.ora
```

**11** Ensure that there are at least two sessions open; one session as user SCOTT and one as user SYS. Enable the restricted session, verify this, and ensure that only the database administrator SYS is connected.

**Hints:**

- Use the ALTER SYSTEM command to enable the restricted session and query the dynamic performance views V\$INSTANCE to verify the result.
- Use the dynamic performance view V\$SESSION to see the values of the SID and SERIAL# column.
- Execute the ALTER SYSTEM KILL SESSION command to terminate sessions.

```
SQL> connect scott/tiger@db01
Connected.
```

**a** In the first session,

```
SQL> INSERT INTO emp (empno, ename, deptno)
      2 VALUES (1, 'Vijay', 10);
1 row inserted.
```

**b** In the second session,

```
SQL> connect sys/oracle@db01 as sysdba
Connected.
SQL> ALTER SYSTEM ENABLE RESTRICTED SESSION;
System altered.
SQL> SELECT logins FROM v$instance;
LOGINS
```

```
-----
RESTRICTED
```

```
SQL> SELECT sid, serial#, username
      2 FROM v$session
      3 WHERE username!= 'SYS';
```

SID	SERIAL#	USERNAME
7	3	SCOTT

```
-----
```

```
SQL> ALTER SYSTEM
      2 KILL SESSION '7,3';
```

```
System altered.
```

```
SQL> SELECT username, status
      2 FROM v$session
      3 WHERE type='USER';
```

USERNAME	STATUS
SCOTT	KILLED
SYS	ACTIVE

**c** In the first session,

```
SQL> SELECT user
      2 FROM dual;
Your session has been killed.
```

**12** Examine the following sample of an alert file to identify if internal errors or exceptions have occurred.

```

Wed Jun 30 13:24:30 1999
Starting ORACLE instance (normal)
LICENSE_MAX_SESSION = 0
LICENSE_SESSIONS_WARNING = 0
LICENSE_MAX_USERS = 0
Starting up ORACLE RDBMS Version: 8.1.5.0.0.
System parameters with non-default values:
  processes                          = 60
  shared_pool_size                    = 3500000
  java_pool_size                      = 1000000
  control_files                       = $HOME/DATA/DISK1/control01.con
  db_block_buffers                     = 100
  db_block_size                       = 4096
  compatible                          = 8.1.5
  log_checkpoint_interval              = 10000
  log_checkpoint_timeout               = 1800
  db_files                            = 1024
  db_file_multiblock_read_count       = 8
  dml_locks                           = 200
  rollback_segments                   = sysrol
  db_domain                           = world
  global_names                        = TRUE
  distributed_transactions             = 10
  sort_area_size                      = 64000
  db_name                             = DB01
  job_queue_processes                 = 2
  job_queue_interval                  = 10
  parallel_max_servers                 = 12
  background_dump_dest                = $HOME/BDUMP
  user_dump_dest                      = $HOME/UDUMP
  max_dump_file_size                  = 10240
  core_dump_dest                      = $HOME/CDUMP
PMON started with pid=2
DBW0 started with pid=3
LGWR started with pid=4
CKPT started with pid=5
SMON started with pid=6
RECO started with pid=7
SNP0 started with pid=8
SNP1 started with pid=9
..
Corrupt block relative dba: 0x01c0003a file=7. blocknum=58.
Fractured block found during buffer read
Data in bad block - type:6. format:2. rdba:0x01c0003a
last change scn:0x0000.0000e9c5 seq:0xa0 flg:0x00
consistency value in tail 0x44c506a0
check value in block header: 0x0, check value not calculated
spare1:0x0, spare2:0x0, spare2:0x0

```

**There is a block corruption message as shown above. Also note other useful messages, such as:**

- **Startup time**
- **Nondefault initialization parameters and background processes**
- **Tablespace creation and log switches (to be discussed in a later lesson)**

## Practice 4 Solutions

The directories specified for the files must be placed within `$HOME/DATA`.

**1** Create a password file using the following information:

- Password for `sys:oracle`
- Enable five privileged users

Ensure that Oracle can write to this file.

**Hint:** Use the `orapwd` utility to create the password file and locate the password file in the `$ORACLE_HOME/dbs` directory.

```
$rm $ORACLE_HOME/dbs/orapwDB01  
  
$orapwd file=$ORACLE_HOME/dbs/orapwDB01 entries=5 \  
> password=oracle  
  
$chmod 777 $ORACLE_HOME/dbs/orapwDB01
```

**2** Write a script for the creation of a database with the following configuration:

- Database name and instance name `DB<xx>`
- One control file named `control01.con`, located in the directory `DISK5`
- Two redo log file groups each with one 150 KB member named `redo0101.log` and `redo0201.log`, located in the directory `DISK6`
- The maximum number of five log file groups and five log file members for each group.
- A 20 MB data file named `system01.dbf`, located in `DISK4` directory
- A maximum of 30 data files that can be created for the database
- A maximum of 100 archived redo logs for automatic media recovery
- The character set `WE8ISO8859P1`

The trace file location should be in the `BDUMP` and `CDUMP` directory.

**Hints:**

- Edit the parameter file.
- Start the instance.
- Generate the `CREATE DATABASE` command script

```
SQL> STARTUP NOMOUNT PFILE=initdb01.ora;
ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                     64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                   532480 bytes
SQL> CREATE DATABASE "DB01"
  2     MAXLOGFILES 5
  3     MAXLOGMEMBERS 5
  4     MAXDATAFILES 30
  5     MAXLOGHISTORY 100
  6 LOGFILE
  7   GROUP 1
  8     '$HOME/DATA/DISK6/redo0101.log'  SIZE 150K,
  9   GROUP 2
 10     '$HOME/DATA/DISK6/redo0201.log'  SIZE 150K
 11 DATAFILE
 12     '$HOME/DATA/DISK4/system01.dbf' SIZE 20M
 13 CHARACTER SET WE8ISO8859P1;
Database created.
```

- 3 Enable spooling to capture the errors and run the script.
- 4 (Optional) Include the command “spool <file\_name>” before running the command.

- 5** (Optional) After creation, check the database state and ensure that the database files are created.

**Hint:** Query the dynamic performance views V\$DATABASE, V\$THREAD, V\$DATAFILE, V\$LOGFILE, and V\$CONTROLFILE.

```
SQL> connect sys/oracle@db01 as sysdba;

Connected.

SQL> SELECT name, created, log_mode FROM v$database;

NAME          CREATED      LOG_MODE
-----
DB01          10-JUN-99    NOARCHIVELOG

SQL> SELECT status, instance FROM v$thread;

STATUS INSTANCE
-----
OPEN      DB01

SQL> SELECT name FROM v$datafile;

NAME
-----
/oracle/hrasmuss/DATA/DISK4/system01.dbf

SQL> SELECT member FROM v$logfile;

MEMBER
-----
/oracle/hrasmuss/DATA/DISK6/redo0101.log
/oracle/hrasmuss/DATA/DISK6/redo0201.log

SQL> SELECT name FROM v$controlfile;

NAME
-----
/oracle/hrasmuss/DATA/DISK5/control01.con

SQL> show sga

Total System Global Area      6315408 bytes
Fixed Size                     64912 bytes
Variable Size                 5308416 bytes
Database Buffers              409600 bytes
Redo Buffers                   532480 bytes
```



**6** Try to display the names of the database users. What happens and why?

```
SQL> SELECT username
      2 FROM dba_users;
FROM dba_users
*
ORA-00942: table or view does not exist
```

**This fails because the data dictionary views have not yet been created.**

## Practice 5 Solutions

### 1 Create the data dictionary views.

```
SQL> @@?/rdbms/admin/catalog
....
Role created.

Grant succeeded.

Commit completed.
```

### 2 Use the data dictionary views to gather the following information:

#### a What is the name and number of the rollback segments?

**Hint:** Query the data dictionary view DBA\_ROLLBACK\_SEGS to show the name of the rollback segments.

```
SQL> SELECT segment_name
       2 FROM dba_rollback_segs;
SEGMENT_NAME
-----
SYSTEM
SYSROL
```

#### b Identify the data file that makes up the SYSTEM tablespace.

**Hint:** Query the data dictionary view DBA\_DATA\_FILES to identify the data files that make up SYSTEM tablespace.

```
SQL> SELECT file_name
       2 FROM dba_data_files
       3 WHERE tablespace_name='SYSTEM';
FILE_NAME
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf
```

#### c How much free space is available in the database and how much is already used?

**Hints:**

- Query the data dictionary view DBA\_FREE\_SPACE to show how much free space is available in the database.
- Query the data dictionary view DBA\_SEGMENTS to display how much space is already used.

```
SQL> SELECT sum(bytes)/1024 "free space in KB"
       2 FROM dba_free_space;
free space in KB
-----
          146976

SQL> SELECT sum(bytes)/1024 "used space in KB"
       2 FROM dba_segments;
used space in KB
-----
          43460
```

**d** List the name and creation date of the database users.

**Hint:** Query the data dictionary view DBA\_USERS to list the name and the creation of the database users.

```
SQL> SELECT username, created FROM dba_users;
USERNAME                                CREATED
-----
SYS                                     10-JUN-99
SYSTEM                                 10-JUN-99
SCOTT                                  10-JUN-99
OUTLN                                  10-JUN-99
DBSNMP                                10-JUN-99
```

**3** Check which data dictionary tables are used to define the DBA\_USERS view.

**Hint:** Query the data dictionary view DBA\_VIEWS to display the view definition of the data dictionary view DBA\_USERS.

```
SQL> SET PAGESIZE 100
SQL> SET LONG 1400
SQL> SELECT text
2 FROM dba_views
3 WHERE view_name = 'DBA_USERS';
TEXT
-----
-----
select u.name, u.user#, u.password,
      m.status,
      decode(u.astatus, 4, u.ltime,
                  5, u.ltime,
                  6, u.ltime,
                  8, u.ltime,
                  9, u.ltime,
                  10, u.ltime, to_date(NULL)),
      decode(u.astatus,
              1, u.exptime,
              2, u.exptime,
              5, u.exptime,
              6, u.exptime,
              9, u.exptime,
              10, u.exptime,
              decode(u.ptime, '', to_date(NULL),
                    decode(pr.limit#, 2147483647, to_date(NULL),
                          decode(pr.limit#, 0,
                                decode(dp.limit#, 2147483647, to_date(NULL),
                                      dp.limit#/86400),
                                u.ptime + pr.limit#/86400))))),
      dts.name, tts.name, u.ctime, p.name, u.defschclass,
u.ext_username
from sys.user$ u, sys.ts$ dts, sys.ts$ tts, sys.profname$ p,
     sys.user_astatus_map m, sys.profile$ pr, sys.profile$ dp
where u.datats# = dts.ts#
and u.resource$ = p.profile#
and u.tempts# = tts.ts#
and u.astatus = m.status#
and u.type# = 1
and u.resource$ = pr.profile#
and dp.profile# = 0
and dp.type#=1
and dp.resource#=1
and pr.type# = 1
and pr.resource# = 1
```

- 4 Establish the usage of PL/SQL functionality. (Take a break as soon as the script is started.)

```
SQL> @@?/rdbms/admin/catproc
....
Package created.

Package body created.
..
Library created.
..
```

- 5 Check if any packages are invalid.

**Hint:** Query the DBA\_OBJECTS view to obtain this information.

```
SQL> COL object_name FORMAT a30
SQL> SELECT object_name, object_type
2   FROM dba_objects
3  WHERE object_type LIKE 'PACKAGE%'
4  AND status = 'INVALID';
```

OBJECT_NAME	OBJECT_TYPE
-----	-----
DBMSZEXP_SYSPKGGRNT	PACKAGE
DBMSZEXP_SYSPKGGRNT	PACKAGE BODY
DBMS_ALERT	PACKAGE
DBMS_ALERT	PACKAGE BODY
...	
UTL_HTTP	PACKAGE BODY
UTL_REF	PACKAGE
UTL_REF	PACKAGE BODY

```
213 rows selected.
```

## Practice 6 Solutions

- 1 Where is the existing control file located and what is the name?

**Hint:** Query the dynamic performance view V\$CONTROLFILE or V\$PARAMETER, or execute the SHOW PARAMETER command to display the name and the location of the control file.

```
SQL> connect sys/oracle@db01 as sysdba
Connected.
SQL> COL name FORMAT a50
SQL> SELECT * FROM v$controlfile;
STATUS      NAME
-----
/oracle/hrasmuss/DATA/DISK1/control01.ctl
```

- 2 Try to start the database without any control files. (You can simulate this by changing the name of the control file in the parameter file or just changing the name of the control file.) What happens?

```
SQL> connect sys/oracle@db01 as sysdba;
Connected.
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> exit
```

**a** Change the name of the control file by moving it into a new filename:

```
mv $HOME/DATA/DISK1/control01.ctl $HOME/DATA/DISK1/control01.bak
SQL> connect sys/oracle@db01 as sysdba;
Connected.
SQL> startup pfile=initDB01.ora
ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                     64912 bytes
Variable Size                  5308416 bytes
Database Buffers               409600 bytes
Redo Buffers                   532480 bytes
ORA-00205: error in identifying controlfile, check alert log for
more info
SQL> shutdown immediate
SQL> exit;
```

**b** Rename your control file to the original name and location:

```
mv $HOME/DATA/DISK1/control01.bak $HOME/DATA/DISK1/control01.ctl
```

**c** Do not startup the database as yet.

- 3** Multiplex the existing control file, using the directory `DISK2`, and name the new control file `control02.con`. Make sure that the Oracle Server is able to write to the new control file. For example, on UNIX use the command `chmod 660`. Confirm that both control files are being used.

**Hints:**

- Shut down the database.
- Copy the existing control file to a new file with the name *control02.con* in the directory `DISK2`.
- Use the command `chmod 660` on UNIX.
- Modify the parameter file to include the new filename.
- Start up the database.
- Query the dynamic performance view `V$CONTROLFILE` or `V$PARAMETER`, or use the `SHOW PARAMETER` command to confirm that both control files are being used.

Ensure the database is shutdown before doing this.

```
$ cp $HOME/DATA/DISK1/control01.con $HOME/DATA/DISK2/control02.con
$ chmod 660 $HOME/DATA/DISK2/control02.con
```

**a** Edit the (init.ora) parameter file to include:

```
control_files = ($HOME/DATA/DISK1/control01.con,
                 $HOME/DATA/DISK2/control02.con)
```

```
SQL> startup pfile=initdb01.ora
```

```
ORACLE instance started.
```

```
Total System Global Area      6315408 bytes
```

```
Fixed Size                     64912 bytes
```

```
Variable Size                  5308416 bytes
```

```
Database Buffers               409600 bytes
```

```
Redo Buffers                   532480 bytes
```

```
Database mounted.
```

```
Database opened.
```

```
SQL> SELECT name
```

```
2 FROM v$controlfile;
```

```
NAME
```

```
-----
/oracle/hrasmuss/DATA/DISK1/control01.con
```

```
/oracle/hrasmuss/DATA/DISK2/control02.con
```

**4** What is the initial sizing of the data file section in your control file?

```
SQL> SELECT records_total
      2 FROM v$controlfile_record_section
      3 WHERE type = 'DATAFILE';
RECORDS_TOTAL
-----
              254
```



## Practice 7 Solutions

- 1 List the number and location of existing log files and display the number of redo log file groups and members your database has.

**Hints:**

- Query the dynamic performance view V\$LOGFILE.
- Use the dynamic performance view V\$LOG.

```
SQL> SELECT member FROM v$logfile;
MEMBER
-----
/oracle/hrasmuss/DATA/DISK3/redo0101.log
/oracle/hrasmuss/DATA/DISK4/redo0102.log
/oracle/hrasmuss/DATA/DISK3/redo0201.log
/oracle/hrasmuss/DATA/DISK4/redo0202.log

SQL> SELECT group#,members
      2 FROM v$log;
GROUP#  MEMBERS
-----  -
        1         2
        2         2
```

- 2 In which database mode is your database configured? Is archiving enabled?

**Hints:**

- Query the dynamic performance view V\$DATABASE.
- Query the dynamic performance view V\$INSTANCE.

```
SQL> SELECT log_mode FROM v$database;
LOG_MODE
-----
NOARCHIVELOG

SQL> SELECT archiver FROM v$instance;
ARCHIVE
-----
STOPPED
```

**3** Add a redo log member to each group in your database located on DISK3, using the following naming conventions:

If group 1 has two existing files called redo0101.log and redo0102.log, then add a new member called redo0103.log.

Verify the result.

**Hints:**

- Execute the ALTER DATABASE ADD LOGFILE MEMBER command to add a redo log member to each group.
- Query the dynamic performance view V\$LOGFILE to verify the result.

```
SQL> ALTER DATABASE
  2  ADD LOGFILE MEMBER
  3  '$HOME/DATA/DISK3/redo0103.log' TO GROUP 1,
  4  '$HOME/DATA/DISK3/redo0203.log' TO GROUP 2;
Database altered.
SQL> COL member FORMAT a40
SQL> SELECT * FROM v$logfile;
```

GROUP#	STATUS	MEMBER
1		/oracle/hrasmuss/DATA/DISK3/redo0101.log
1		/oracle/hrasmuss/DATA/DISK4/redo0102.log
2		/oracle/hrasmuss/DATA/DISK3/redo0201.log
2		/oracle/hrasmuss/DATA/DISK4/redo0202.log
1	INVALID	/oracle/hrasmuss/DATA/DISK3/redo0103.log
2	INVALID	/oracle/hrasmuss/DATA/DISK3/redo0203.log

6 rows selected.

- 4 Create a new redo log group located in directories DISK3, DISK4, and DISK5, and verify the existence of the new group.

**Hints:**

- Execute the ALTER DATABASE ADD LOGFILE command to create a new group.
- Query the dynamic performance view V\$LOGFILE to display the name of the new members of the new group.
- Query the dynamic performance view V\$LOG to display the number of redo log file groups and members.

```
SQL> ALTER DATABASE
2  ADD LOGFILE GROUP 3(
3  '$HOME/DATA/DISK3/redo0301.log',
4  '$HOME/DATA/DISK4/redo0302.log',
5  '$HOME/DATA/DISK5/redo0303.log')
6  SIZE 1024K;
Database altered.
SQL> SELECT * FROM v$logfile;
GROUP# STATUS  MEMBER
-----
1          /oracle/hrasmuss/DATA/DISK3/redo0101.log
1          /oracle/hrasmuss/DATA/DISK4/redo0102.log
2          /oracle/hrasmuss/DATA/DISK3/redo0201.log
2          /oracle/hrasmuss/DATA/DISK4/redo0202.log
1 INVALID  /oracle/hrasmuss/DATA/DISK3/redo0103.log
2 INVALID  /oracle/hrasmuss/DATA/DISK3/redo0203.log
3          /oracle/hrasmuss/DATA/DISK4/redo0301.log
3          /oracle/hrasmuss/DATA/DISK4/redo0302.log
3          /oracle/hrasmuss/DATA/DISK5/redo0303.log

9 rows selected.
SQL> SELECT group#, members FROM V$log;
GROUP#  MEMBERS
-----
1          3
2          3
3          3

3 rows selected.
```

- 5** Relocate the members redo0103.log and redo0203.log (see step 3) and locate them in the directory DISK5 from DISK3.

**Hints:**

- Move the members to the directory DISK5.
- Execute the ALTER DATABASE RENAME FILE command to rename the members.
- Query the dynamic performance view V\$LOGFILE to verify the result.

```
$ cd $HOME/DATA/DISK5
$ mv $HOME/DATA/DISK3/redo0103.log.
$ mv $HOME/DATA/DISK3/redo0203.log.
```

**a** Use the following statement to decide which group is current:

```
SQL> SELECT group#, status
       2 FROM v$log;
GROUP# STATUS
-----
1 INACTIVE
2 CURRENT
3 UNUSED
```

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
System altered.
SQL> ALTER DATABASE
       2 RENAME FILE '$HOME/DATA/DISK3/redo0103.log',
       3 '$HOME/DATA/DISK3/redo0203.log'
       4 TO '$HOME/DATA/DISK5/redo0103.log',
       5 '$HOME/DATA/DISK5/redo0203.log';
Database altered.
```

```
SQL> COL member FORMAT a50
SQL> SELECT * FROM v$logfile;
GROUP# STATUS MEMBER
-----
1 /oracle/hrasmuss/DATA/DISK3/redo0101.log
1 /oracle/hrasmuss/DATA/DISK4/redo0102.log
2 /oracle/hrasmuss/DATA/DISK3/redo0201.log
2 /oracle/hrasmuss/DATA/DISK4/redo0202.log
1 INVALID /oracle/hrasmuss/DATA/DISK5/redo0103.log
2 INVALID /oracle/hrasmuss/DATA/DISK5/redo0203.log
3 /oracle/hrasmuss/DATA/DISK4/redo0301.log
3 /oracle/hrasmuss/DATA/DISK4/redo0302.log
3 /oracle/hrasmuss/DATA/DISK5/redo0303.log

9 rows selected.
```

**6** Remove the redo log group created in step 4.**Hints:**

- Execute the ALTER DATABASE DROP LOGFILE GROUP command to remove the log group.
- Query the dynamic performance view V\$LOG to verify the result.

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
System altered.
SQL> ALTER DATABASE DROP LOGFILE GROUP 3;
Database altered.
SQL> SELECT group#, members FROM v$log;
GROUP#      MEMBERS
-----
          1          2
          2          2
2 rows selected.
$rm $HOME/DATA/DISK5/redo0301.log
$rm $HOME/DATA/DISK5/redo0302.log
$rm $HOME/DATA/DISK5/redo0303.log
```

- 7** Resize all online redo log files to 1024 KB. (Because we cannot resize log files, we have to add new logs and drop the old.)

**Hints:**

- Execute the ALTER DATABASE ADD LOGFILE GROUP command to add two new groups with the size 1024 KB.
- Query the dynamic performance view V\$LOG to check the active group.
- Execute the ALTER SYSTEM SWITCH LOGFILE command to force log switches and change the group stage to inactive.
- Execute the ALTER DATABASE DROP LOGFILE command to remove the inactive groups.
- Query the dynamic performance view V\$LOG to verify the result.

```
SQL> ALTER DATABASE ADD LOGFILE
  2 GROUP 3 ('$HOME/DATA/DISK3/redo0301.log',
  3 '$HOME/DATA/DISK4/redo0302.log') SIZE 1024K,
  4 GROUP 4 ('$HOME/DATA/DISK3/redo0401.log',
  5 '$HOME/DATA/DISK4/redo0402.log') size 1024K;
Database altered.
SQL> SELECT group#, status FROM v$log;
GROUP# STATUS
-----
1 INACTIVE
2 CURRENT
3 UNUSED
4 UNUSED
4 rows selected.
SQL> ALTER SYSTEM SWITCH LOGFILE;
System altered.
SQL> ALTER SYSTEM SWITCH LOGFILE;
System altered.
SQL> ALTER DATABASE DROP LOGFILE GROUP 1, GROUP 2;
Database altered.
SQL> SELECT group#, bytes FROM v$log;
GROUP# BYTES
-----
3 1048576
4 1048576
2 rows selected.
```

## Practice 8 Solutions

- 1 Create permanent tablespaces with the following names and storage:
  - a DATA01 for tables with default storage
  - b DATA02 for large objects with default storage (Ensure that every used extent size in the tablespace is multiple of 100 KB.)
  - c INDX01 for indexes with the default storage (Enable automatic extension of 500 KB when more extents are required.)
  - d RONLY for read-only tables with the default storage

Display the information from the data dictionary

**Hints:**

- Execute the CREATE TABLESPACE command to create the permanent tablespaces.
- Display the information from the data dictionary.
- Query the dynamic performance view DBA\_DATA\_FILES to verify the result.

Tablespace Name	Subdirectory	Data File Location (Size)
DATA01	DISK4	data01.dbf (2 MB)
DATA02	DISK5	data02.dbf (1 MB)
INDX01	DISK3	indx01.dbf (1 MB)
RONLY	DISK1	ronly.dbf (1 MB)

```

SQL> CREATE TABLESPACE DATA01 DATAFILE
      2 '$HOME/DATA/DISK4/data01.dbf' SIZE 2M;
Tablespace created.
SQL> CREATE TABLESPACE DATA02 DATAFILE
      2 '$HOME/DATA/DISK5/data02.dbf' SIZE 1M
      3 MINIMUM EXTENT 100k;
Tablespace created.
SQL> CREATE TABLESPACE INDX01 DATAFILE
      2 '$HOME/DATA/DISK3/indx01.dbf' SIZE 1M
      3 AUTOEXTEND ON NEXT 500K;
Tablespace created.
SQL> CREATE TABLESPACE RONLY DATAFILE
      2 '$HOME/DATA/DISK1/ronly.dbf' SIZE 1M;
Tablespace created.
SQL> COL file_name FORMAT A40
SQL> COL tablespace_name FORMAT A15
SQL> SELECT file_name, tablespace_name, bytes,
      2 autoextensible, increment_by
      3 FROM DBA_DATA_FILES;

```

FILE_NAME	TABLESPACE_NAME	BYTES
/oracle/hrasmuss/DATA/DISK4/data01.dbf	DATA01	2097152
/oracle/hrasmuss/DATA/DISK5/data02.dbf	DATA02	1048576
/oracle/hrasmuss/DATA/DISK3/indx01.dbf	INDX01	1048576
/oracle/hrasmuss/DATA/DISK1/ronly.dbf	RONLY	1048576
/oracle/hrasmuss/DATA/DISK1/system01.dbf	SYSTEM	52428800
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf	RBS	15728640
/oracle/hrasmuss/DATA/DISK3/data01.dbf	DATA	52428800
/oracle/hrasmuss/DATA/DISK2/temp01.dbf	TEMP	10485760
/oracle/hrasmuss/DATA/DISK2/indx01.dbf	INDX	5242880
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf	OEM	5242880
/oracle/hrasmuss/DATA/DISK1/query01.dbf	QUERY_DATA	1048576

11 rows selected.



## 2 Allocate 500 KB more to the tablespace DATA02 and verify the result.

### Hints:

- Use the ALTER DATABASE DATAFILE... RESIZE command to allocate another 500 KB.
- Query the dynamic performance view V\$DATAFILE to verify the result.

```
SQL> ALTER DATABASE DATAFILE
      2 '$HOME/DATA/DISK5/data02.dbf' RESIZE 1500K;
Database altered.
SQL> COL name FORMAT a40
SQL> SELECT name, bytes, create_bytes FROM v$datafile;
NAME                                     BYTES  CREATE_BYTES
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf 52428800 52428800
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf   15728640 15728640
/oracle/hrasmuss/DATA/DISK3/data01.dbf   52428800 52428800
/oracle/hrasmuss/DATA/DISK2/temp01.dbf   10485760 10485760
/oracle/hrasmuss/DATA/DISK2/indx01.dbf    5242880 5242880
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf  5242880 5242880
/oracle/hrasmuss/DATA/DISK1/query01.dbf  1048576 1048576
/oracle/hrasmuss/DATA/DISK4/data01.dbf   2097152 2097152
/oracle/hrasmuss/DATA/DISK5/data02.dbf   1536000 1048576
/oracle/hrasmuss/DATA/DISK3/indx01.dbf   1048576 1048576
/oracle/hrasmuss/DATA/DISK1/ronly.dbf    1048576 1048576

11 rows selected.
```

**3 Relocate the INDX01 tablespace and move it to DISK6.****Hints:**

- Take the tablespace INDX01 OFFLINE.
- Query the dynamic performance view V\$DATAFILE to verify the result
- Execute the ALTER TABLESPACE RENAME command to rename the files.
- Take the tablespace INDX01 ONLINE.
- Query the dynamic performance views V\$DATAFILE to verify the result

```
SQL> ALTER TABLESPACE indx01 OFFLINE;
Tablespace altered.
SQL> SELECT name, status FROM v$datafile;
NAME                                STATUS
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf SYSTEM
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK3/data01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK2/temp01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK2/indx01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf ONLINE
/oracle/hrasmuss/DATA/DISK1/query01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK4/data01.dbf   ONLINE
/oracle/hrasmuss/DATA/DISK5/data02.dbf   ONLINE
/oracle/hrasmuss/DATA/DISK3/indx01.dbf   OFFLINE
/oracle/hrasmuss/DATA/DISK1/ronly.dbf    ONLINE
11 rows selected.
SQL> !mv $HOME/DATA/DISK3/indx01.dbf $HOME/DATA/DISK6/indx01.dbf
SQL> ALTER TABLESPACE indx01
  2  RENAME DATAFILE
  3  '$HOME/DATA/DISK3/indx01.dbf' TO
  4  '$HOME/DATA/DISK6/indx01.dbf';
Tablespace altered.
SQL> ALTER TABLESPACE indx01 ONLINE;
Tablespace altered.
SQL> SELECT name, status FROM v$datafile;
NAME                                STATUS
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf SYSTEM
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK3/data01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK2/temp01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK2/indx01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf ONLINE
/oracle/hrasmuss/DATA/DISK1/query01.dbf  ONLINE
/oracle/hrasmuss/DATA/DISK4/data01.dbf   ONLINE
/oracle/hrasmuss/DATA/DISK5/data02.dbf   ONLINE
/oracle/hrasmuss/DATA/DISK6/indx01.dbf   ONLINE
/oracle/hrasmuss/DATA/DISK1/ronly.dbf    ONLINE
11 rows selected.
```

- 4 Change the RONLY tablespace to read-only after creating a table in this tablespace. Attempt to create an additional table and drop the table. What happens and why?

```
SQL> CREATE TABLE t1 (t1 number) TABLESPACE ronly;
Table altered.
SQL> ALTER TABLESPACE ronly READ ONLY;
Tablespace altered.
SQL> SELECT name, enabled, status FROM v$datafile;
NAME                                ENABLED    STATUS
-----
/oracle/hrasmuss/DATA/DISK1/system01.dbf  READ WRITE SYSTEM
/oracle/hrasmuss/DATA/DISK2/rbs01.dbf     READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK3/data01.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK2/temp01.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK2/indx01.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK3/oemrep01.dbf  READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK1/query01.dbf   READ ONLY  ONLINE
/oracle/hrasmuss/DATA/DISK4/data01.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK5/data02.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK6/indx01.dbf    READ WRITE ONLINE
/oracle/hrasmuss/DATA/DISK1/ronly.dbf     READ ONLY  ONLINE

11 rows selected.
SQL> CREATE TABLE t2 (t2 number) TABLESPACE ronly;
CREATE TABLE t2 (t2 number) TABLESPACE ronly
*
ORA-01647: tablespace 'RONLY' is read only, cannot allocate space
in it
SQL> DROP TABLE t1;
Table dropped.
```

**5** Drop the tablespace RONLY and verify it.**Hints:**

- Execute the DROP TABLESPACE ...command to remove the tablespace.
- Delete the operating system files.
- Query the dynamic performance view V\$TABLESPACE to verify the result.

```
SQL> DROP TABLESPACE ronly;
Tablespace dropped.
SQL> SELECT * FROM v$tablespace;
TS#          NAME
-----
0 SYSTEM
1 RBS
2 DATA01
3 DATA02
4 TEMP
5 INDX01
6 rows selected.
$rm $HOME/DATA/DISK1/ronly.dbf
```

**6** Without shutting down the instance, change the SORT\_AREA\_SIZE to 2 kilobytes.

**Hint:** SORT\_AREA\_SIZE is a dynamic parameter requiring the DEFERRED option.

```
SQL> ALTER SYSTEM SET sort_area_size=2048 DEFERRED;
System altered.
```

**7** Open two connections to the database as user SYSTEM. Run srt\_dd.sql from one session and monitor the sort activity from another session. Query sort statistics and temporary segment information both during and after the completion of the script. Note the findings.

**Hint:** Sort segment information is available from V\$SORT\_SEGMENT and current sort activity from V\$SORT\_USAGE.

**8** From one of the sessions, run asn\_tts.sql to prepare for the next part of the practice. This script ensures that the TEMP tablespace will be used for the sorts made by the user SYSTEM; this will be covered under the lesson “Managing Users.” Connect as SYSTEM from one of the sessions and run srt\_dd.sql. From the other session monitor sort activity and statistics, as was done in question 3. Do you notice any difference?

**The sort in this case uses the temporary tablespace, TEMP, and the extents allocated are not released at the end of the sort.**

**9** Reset SORT\_AREA\_SIZE.

```
SQL> ALTER SYSTEM SET sort_area_size=65536 DEFERRED;  
System altered.
```

## Practice 9 Solutions

- 1 As user `system`, run the script `cr_segs.sql` to create tables and indexes.

```
SQL> connect system/manager
Connected.
SQL> @$HOME/LABS/cr_segs
SQL> -- Script cr_segs.sql to create segments
SQL> -- for Practice 09, Q1 08iDBA class
SQL> -- Dependencies :
SQL> -- needs SYSTEM account to run
SQL> -- needs DATA01 tablespace with exactly 2M free space
SQL> -- needs INDX01 tablespace with at least 100K free space
SQL>
SQL> CREATE TABLE emp(
  2 empno NUMBER(4),
  3 ename VARCHAR2(30))
  4 TABLESPACE data01
  5 STORAGE (INITIAL 100K
  6           NEXT 100K
  7           PCTINCREASE 0
  8           MINEXTENTS 8
  9           MAXEXTENTS 10)
 10 /
Table created.
SQL> SQL> CREATE TABLE fragment1(
  2 a NUMBER)
  3 TABLESPACE data01
  4 STORAGE(INITIAL 10K)
  5 /
Table created.
SQL> CREATE TABLE dept(
  2 deptno NUMBER,
  3 dname VARCHAR2(15))
  4 TABLESPACE data01
  5 STORAGE(INITIAL 50K
  6 NEXT 50K)
  7 /
Table created.
```

```
SQL> CREATE TABLE fragment2(
  2 a NUMBER)
  3 TABLESPACE data01
  4 STORAGE(INITIAL 8K)
  5 /
Table created.
SQL>
SQL> CREATE TABLE big_emp(
  2 empno NUMBER(4),
  3 ename VARCHAR2(30))
  4 TABLESPACE data01
  5 STORAGE (INITIAL 1M
  6          NEXT 1M
  7          MAXEXTENTS 10)
  8 /
Table created.
SQL>
SQL> CREATE INDEX i_e_empno
  2      ON emp(ename)
  3      TABLESPACE indx01
  4      STORAGE(INITIAL 50K
  5              NEXT 50K)
  6 /
Index created.
SQL>
SQL> DROP TABLE fragment1
  2 /
Table dropped.
SQL>
SQL> DROP TABLE fragment2
  2 /
Table dropped.
```

## 2 Identify the different types of segments in the database.

```
SQL> connect system/manager
Connected.
SQL> SELECT DISTINCT segment_type
  2 FROM dba_segments;
SEGMENT_TYPE
-----
CACHE
CLUSTER
INDEX
LOBINDEX
LOBSEGMENT
ROLLBACK
TABLE

7 rows selected.
```

- 3 Write a query to check which segments are within five extents short of the maximum extents. Ignore the bootstrap segment. This query is useful in identifying any segments that are likely to generate errors during future data load.

**Hint:** The columns EXTENTS and MAX\_EXTENTS in the data dictionary view DBA\_SEGMENTS contain the current number and the maximum number of extents.

```
SQL> SELECT segment_name, segment_type, max_extents, extents
       3 FROM dba_segments
       4 WHERE extents+5 > max_extents
       5 AND segment_type<>'CACHE';
```

SEGMENT_NAME	SEGMENT_TYPE	MAX_EXTENTS	EXTENTS
EMP	TABLE	10	8

- 4 Which files have space allocated for the EMP table?

**Hint:** The views DBA\_EXTENTS and DBA\_DATA\_FILES need to be joined to obtain this information.

```
SQL> SELECT DISTINCT f.file_name
       2 FROM dba_extents e,dba_data_files f
       3 WHERE e.segment_name='EMP'
       4 AND e.file_id=f.file_id;
```

FILE_NAME
/oracle/hrasmuss/DATA/DISK3/data01.dbf
/oracle/hrasmuss/DATA/DISK4/data01.dbf

- 5 (a) Write a query to obtain the file number and block number of the EMP table header.

**Hint:** To find the header block of the EMP table, query the DBA\_SEGMENTS view.

```
SQL> SELECT relative_fno, header_block
       2 FROM dba_segments
       3 WHERE owner='SYSTEM'
       4 AND segment_name = 'EMP';
```

RELATIVE_FNO	HEADER_BLOCK
8	2



(b) Write a query that will accept a file number and block number as the input and return the name and type of the segment that uses the block. Test your query by supplying the file and block number of the EMP table header (obtained in the previous query). Use SQL\*Plus to run the query.

**Hint:** Create a SQL\*Plus script to accept the relative file number and block number, and query the view DBA\_EXTENTS to find the segment that uses the relative file and the block number entered.

```
SQL> COL segment_name FORMAT A30
SQL> COL segment_type FORMAT A15
SQL> SELECT segment_name,segment_type
  2   FROM dba_extents
  3   WHERE file_id=&1
  4   AND &2 BETWEEN block_id AND (block_id+blocks-1);
Enter value for 1: 8
old   3: WHERE file_id=&1
new   3: WHERE file_id=8
Enter value for 2: 2
old   4: AND &2 between block_id and (block_id+blocks+1)
new   4: AND 2 between block_id and (block_id+blocks+1)
```

SEGMENT_NAME	SEGMENT_TYPE
EMP	TABLE

- 6 List the free space available by tablespace. The query should display the number of fragments, the total free space, and the largest free extent in each tablespace.

**Hint:** Query the DBA\_FREE\_SPACE view to get this information.

```
SQL> SELECT tablespace_name,COUNT(*) AS fragments,
  2   SUM(bytes) AS total,
  3   MAX(bytes) AS largest
  4 FROM dba_free_space
  5 GROUP BY tablespace_name;
```

TABLESPACE_NAME	FRAGMENTS	TOTAL	LARGEST
DATA	1	50376704	50376704
DATA01	3	147456	126976
DATA02	1	1531904	1531904
INDX	1	5238784	5238784
INDX01	1	983040	983040
OEM	1	5238784	5238784
QUERY_DATA	1	1044480	1044480
RBS	1	1593344	1593344
SYSTEM	1	64565248	64565248
TEMP	5	10481664	9662464

10 rows selected.

- 7 Run the script `cr_fragments.sql` in the LABS directory. Check if there are any adjacent free extents in the database. Coalesce them and verify again.

**Hint:** Execute the script `cr_fragments.sql`.

```
SQL> @$HOME/LABS/cr_fragments
```

**Hint:** Query the `DBA_FREE_SPACE_COALESCED` view for any tablespaces where `PERCENT_BLOCKS_COALESCED <> 100`.

```
SQL> SELECT tablespace_name, total_blocks,
2 blocks_coalesced, percent_blocks_coalesced
3 FROM dba_free_space_coalesced
4 WHERE percent_blocks_coalesced<>100;
TABLESPACE TOTAL_BLOCKS BLOCKS_COALESCED PERCENT_BLOCKS_COALESCED
-----
TEMP                2559                50                1.9538882
DATA02              374                 5                1.3368984
```

**Hint:** Use the `ALTER TABLESPACE` command to coalesce free space in the tablespace.

```
SQL> ALTER TABLESPACE data02 COALESCE;
```

**Hint:** Query the `DBA_FREE_SPACE_COALESCED` view again to check for any tablespaces where `PERCENT_BLOCKS_COALESCED <> 100`.

```
SQL> SELECT tablespace_name, total_blocks,
2 blocks_coalesced, percent_blocks_coalesced
3 FROM dba_free_space_coalesced
4 WHERE percent_blocks_coalesced<>100;
TABLESPACE TOTAL_BLOCKS BLOCKS_COALESCED PERCENT_BLOCKS_COALESCED
-----
TEMP                2559                50                1.9538882
```

- 8 List segments that will generate errors because of lack of space when they try to allocate an additional extent.

**Hint:** You need to check if the biggest free extent for a tablespace from DBA\_FREE\_SPACE is not as large as the NEXT\_EXTENT for any segment in the tablespace.

```
SQL> SELECT s.segment_name,s.segment_type,
2          s.tablespace_name,s.next_extent
3 FROM dba_segments s
4 WHERE NOT EXISTS
5        (SELECT 1 FROM dba_free_space f
6          WHERE s.tablespace_name=f.tablespace_name
7            HAVING max(f.bytes) > s.next_extent);
```

SEGMENT_NAME	SEGMENT_TYPE	TABLESPACE	NEXT_EXTENT
BIG_EMP	TABLE	DATA01	1048576

## Practice 10: Managing Rollback Segments

Before starting this practice make sure that you run the script \$HOME/LABS/alt\_sysrol.sql as user SYSTEM.

- 1 Connect as system (password = manager) and insert a record into the EMP table. Was the operation successful? Why or why not?

```
SQL> INSERT INTO emp VALUES
  2 (7369,'SMITH','CLERK',7902,
  3  to_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);
ORA-01552: cannot use system rollback segment for non-system
tablespace 'DATA01'
```

This operation fails because DML on a non-SYSTEM tablespace requires a non-SYSTEM rollback segment, and currently there are none available.

- 2 You are going to be running an online order entry application on your database. The orders will be entered using 15 client stations, which have a very high volume of activity in the mornings. Create an appropriate number of rollback segments in the database. (Assume default sizing for the purpose of this exercise.)

**Hint:** Using a ratio of one rollback segment for every four transactions, you will need to create four rollback segments.

```
SQL> CREATE ROLLBACK SEGMENT r01
  2 TABLESPACE rbs;
Rollback segment created.
SQL> CREATE ROLLBACK SEGMENT r02
  2 TABLESPACE rbs;
Rollback segment created.
SQL> CREATE ROLLBACK SEGMENT r03
  2 TABLESPACE rbs;
Rollback segment created.
SQL> CREATE ROLLBACK SEGMENT r04
  2 TABLESPACE rbs;
Rollback segment created.
```

- 3 Ensure that you can perform the INSERT in question 1. Test whether you are able to insert a record into the EMP table. Do not commit the INSERT.

**Hint:** At least one of the rollback segments needs to be ONLINE before you can successfully insert a row into the EMP table.

```
SQL> ALTER ROLLBACK SEGMENT r01 ONLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r02 ONLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r03 ONLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r04 ONLINE;
Rollback segment altered.
SQL> INSERT INTO emp VALUES
  2 (7369,'SMITH','CLERK',7902,
  3  to_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);
```

- 4 Verify which rollback segments in the system are available for use by transactions.

**Hint:** This information can be obtained from DBA\_ROLLBACK\_SEGS view.

```
SQL> SELECT segment_name, status
  2 FROM dba_rollback_segs;
SEGMENT_NAME STATUS
-----
SYSTEM          ONLINE
SYSROL           OFFLINE
R01              ONLINE
R02              ONLINE
R03              ONLINE
R04              ONLINE

6 rows selected.
```

- 5 Find which rollback segment is used by the transaction.

**Hint:** This information can be obtained by joining the V\$ROLLSTAT and V\$ROLLNAME views.

```
SQL> SELECT n.name
  2 FROM V$rollname n, V$rollstat s
  3 WHERE n.usn=s.usn
  4 AND s.xacts>0;
NAME
-----
R01
```

- 6 Invoke SQL\*Plus from the labs directory. Run the script `ins_emp.sql`. Using a separate session take the rollback segment tablespace offline.

**Hint:** The script starts another transaction that inserts a row into the EMP table. Because both transactions are using the rollback segments in this tablespace, you need to use the following steps:

First, take all rollback segments in the tablespace OFFLINE to prevent new transactions from using the rollback segments in the tablespace.

```
SQL> ALTER ROLLBACK SEGMENT r01 OFFLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r02 OFFLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r03 OFFLINE;
Rollback segment altered.
SQL> ALTER ROLLBACK SEGMENT r04 OFFLINE;
Rollback segment altered.
```

**Hint:** Now, identify the sessions that are using the rollback segments in the RBS tablespace.

```
SQL> SELECT s.sid, s.serial#
2 FROM v$session s
3 WHERE s.saddr in
4 (SELECT t.ses_addr
5 FROM V$transaction t, dba_rollback_segs r
6 WHERE t.xidusn=r.segment_id
7 AND r.tablespace_name='RBS');
SID SERIAL#
-----
9 362
11 306
```

**Hint:** Kill the sessions using rollback segments in the tablespace.

```
SQL> ALTER SYSTEM KILL SESSION '9,362';
System altered.
SQL> ALTER SYSTEM KILL SESSION '11,306';
System altered.
```

**Hint:** Take the RBS tablespace offline.

```
SQL> ALTER TABLESPACE rbs OFFLINE;
```

- 7 (a) Shut down the instance, start up again, and query the rollback segment information in the data dictionary to check how many rollback segments are online.

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup pfile=$HOME/initDB01.ora
ORACLE instance started.
Total System Global Area      26676624 bytes
Fixed Size                     64912 bytes
Variable Size                 18149376 bytes
Database Buffers              8388608 bytes
Redo Buffers                   73728 bytes
Database mounted.
Database opened.
SQL> SELECT segment_name, status
        2 FROM dba_rollback_segs;
SEGMENT_NAME STATUS
-----
SYSTEM          ONLINE
SYSROL          OFFLINE
R01             OFFLINE
R02             OFFLINE
R03             OFFLINE
R04             OFFLINE

6 rows selected.
```

- (b) Ensure that all rollback segments will be brought online whenever the database is opened in the future and restart the instance. Make sure that all the rollback segments are ONLINE.

**Hint:** This can be done using the ROLLBACK\_SEGMENTS initialization parameter. Also, the RBS tablespace must be brought ONLINE.

```
SQL> ALTER TABLESPACE rbs ONLINE;
Tablespace altered.
SQL> shutdown
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> exit
```

**a** Edit the initdb01.ora and add the following line:

```
rollback_segments = (r01,r02,r03,r04)
```

```
SQL> startup pfile=$HOME/initDB01.ora
ORACLE instance started.
Total System Global Area      6315408 bytes
Fixed Size                     64912 bytes
Variable Size                  5308416 bytes
Database Buffers               409600 bytes
Redo Buffers                   532480 bytes
Database mounted.
Database opened.
SQL> SELECT segment_name, status
2 FROM dba_rollback_segs;
SEGMENT_NAME STATUS
-----
SYSTEM          ONLINE
SYSROL          OFFLINE
R01             ONLINE
R02             ONLINE
R03             ONLINE
R04             ONLINE

6 rows selected.
```



- 8 Create a new rollback segment called DEMO\_RBS in the database with the following storage parameters, and ensure that it can be used. This is essential for the next question:

INITIAL = 10K

NEXT = 10K

OPTIMAL = 30K

**Hint:** To ensure that the rollback segment can be used, it must be brought ONLINE after creation.

```
SQL> CREATE ROLLBACK SEGMENT demo_rbs
  2 TABLESPACE rbs
  3 STORAGE(INITIAL 16k NEXT 16K OPTIMAL 32K);
Rollback segment created.
SQL> ALTER ROLLBACK SEGMENT demo_rbs ONLINE;
Rollback segment altered.
```

- 9 Check the number of extents in the DEMO\_RBS rollback segment. Log in as system/manager using SQL\*Plus and run the script ext\_rbs.sql. Is there any change in the number of extents in the rollback segment?

```
SQL> SELECT extents
  2 FROM dba_segments
  3 WHERE segment_name='DEMO_RBS';
EXTENTS
-----
          2

SQL> @$HOME/LABS/ext_rbs
Table truncated.
Transaction set.
PL/SQL procedure successfully completed.

SQL> SELECT extents
  2 FROM dba_segments
  3 WHERE segment_name='DEMO_RBS';
EXTENTS
-----
          5
```

- 10** Ensure that the rollback segment DEMO\_RBS is reduced to its optimal size and verify that it has reduced.

**Hint:** The OPTIMAL size can be verified from V\$ROLLSTAT.

```
SQL> SELECT extents, rssize, optsize
  2 FROM v$rollstat
  3 WHERE usn = (SELECT usn
  4   FROM v$rollname
  5   WHERE name='DEMO_RBS');
EXTENTS      RSSIZE      OPTSIZE
-----
          5      86016      32768
```

**Hint:** Use the ALTER ROLLBACK SEGMENT command to reduce the size, and verify again.

```
SQL> ALTER ROLLBACK SEGMENT demo_rbs SHRINK;
Rollback segemnt altered
SQL> SELECT extents, rssize, optsize
  2 FROM v$rollstat
  3 WHERE usn = (SELECT usn
  4   FROM v$rollname
  5   WHERE name='DEMO_RBS');
EXTENTS      RSSIZE      OPTSIZE
-----
          2      36864      32768
```

- 11** Run `ins_dept1.sql` connected as `SYSTEM`. Using a separate session, connect as `SYSTEM` and run `ins_emp3.sql`. Check if there is a blocking session.

**Hint:** Determine blocking session from `V$ROLLSTAT`.

**a** In the first Session:

```
SQL> @$HOME/LABS/ins_dept1
```

**b** In the Second Session:

```
SQL> @$HOME/LABS/ins_emp3
```

Table truncated.

Transaction set.

PL/SQL procedure successfully completed.

```
SQL> SELECT s.sid, s.serial#, t.start_time, t.xidusn
2      FROM v$session s, v$transaction t, v$rollstat r
3      WHERE s.saddr = t.ses_addr
4      AND t.xidusn = r.usn
5      AND ((r.curext = t.start_uext-1) OR
6      ((r.curext = r.extents-1) AND t.start_uext=0));
```

SID	SERIAL#	START_TIME	XIDUSN
7	3	01/03/98 20:07:14	5
10	9	01/03/98 20:07:24	5

- 12** Drop the `DEMO_RBS` rollback segment.

**Hint:** The rollback must be taken offline before it is dropped. You may need to ensure that no transactions are using the rollback segment.

```
SQL> ALTER ROLLBACK SEGMENT demo_rbs OFFLINE;
```

Rollback segment altered.

```
SQL> DROP ROLLBACK SEGMENT demo_rbs;
```

Rollback segment dropped.

## Practice 11 Solutions

- 1 You need to create the following tables for an order entry system that you are implementing now. The tables and the columns are shown below:

Table	Column	Data Type and Size
CUSTOMERS	CUST_CODE	VARCHAR2(3)
	NAME	VARCHAR2(50)
	REGION	VARCHAR2(5)
ORDERS	ORD_ID	NUMBER(3)
	ORD_DATE	DATE
	CUST_CODE	VARCHAR2(3)
	DATE_OF_DELY	DATE

You have been informed that in the table ORDERS, rows will be inserted without a value for DATE\_OF\_DELY, and it will be updated when the order is fulfilled. Log in as `system/manager` and create the tables using the appropriate block space utilization and tablespace settings. Use tablespace DATA01. You can use the default storage settings. Use the Table Wizard when creating table CUSTOMERS. Do not use the Table Wizard when creating table ORDERS.

**Hint:** PCTFREE has to be set carefully for the ORDERS table because rows in this table are likely to grow after updates.

```
SQL> CREATE TABLE SYSTEM.customers(
  2 cust_code varchar2(3),
  3 name varchar2(50),
  4 region varchar2(5))
  5 TABLESPACE data01;
Table created.
SQL> CREATE TABLE SYSTEM.orders(
  2 ord_id NUMBER(3),
  3 ord_date DATE,
  4 cust_code varchar2(3),
  5 date_of_dely DATE)
  6 TABLESPACE data01
  7 PCTFREE 35;
Table created.
```

**2** Run the script `ins_cord.sql` to insert rows into the tables.

```
SQL> @$HOME/LABS/ins_cord
1 row created.

1 row created.
....
```

**3** Find which files and blocks contain the orders from the customer with `CUST_CODE=A04`.

**Hint:** You need to use the `DBMS_ROWID` package to translate the ROWID. Since the database only has a few files, the relative file number and absolute file numbers are the same.

```
SQL> SELECT DISTINCT dbms_rowid.rowid_relative_fno(rowid) AS
"File",
2          dbms_rowid.rowid_block_number(rowid) AS "Block"
3 FROM orders
4 WHERE cust_code='A04';
File          Block
-----
          8          488
```

**4** Check the number of extents used by the table `ORDERS`.

```
SQL> SELECT COUNT(*)
2 FROM dba_extents
3 WHERE segment_name='ORDERS'
4 AND owner='SYSTEM'
COUNT(*)
-----
          1
```

**5** Allocate an extent manually, with default size, for the table `ORDERS` and confirm that the extent has been added as specified.

```
SQL> ALTER TABLE system.orders
2 ALLOCATE EXTENT;
Table altered.
SQL> SELECT COUNT(*)
2 FROM dba_extents
3 WHERE segment_name='ORDERS'
4 AND owner='SYSTEM';
COUNT(*)
-----
          2
```

**6 (a)** Drop the table BIG\_EMP.

```
SQL> DROP TABLE system.big_emp;
Table dropped.
```

(b) Create another table, ORDERS2 as copy of the ORDERS table, but with MINEXTENTS=10 and PCTINCREASE=0. Verify that the table has been created with the specified number of extents.

```
SQL> CREATE TABLE system.orders2
 2 TABLESPACE data01
 3 STORAGE(MINEXTENTS 10 PCTINCREASE 0)
 4 AS
 5 SELECT * FROM system.orders;
Table created.
SQL> SELECT COUNT(*)
 2 FROM dba_extents
 3 WHERE segment_name='ORDERS2'
 4 AND owner='SYSTEM';
COUNT(*)
-----
        10
```

**7** Truncate table ORDERS without releasing space and check the number of extents to verify extents have not been deallocated.

```
SQL> TRUNCATE TABLE system.orders
 2 REUSE STORAGE;
Table truncated.
SQL> SELECT COUNT(*)
 2 FROM dba_extents
 3 WHERE segment_name='ORDERS'
 4 AND owner='SYSTEM';
COUNT(*)
-----
        2
```

**8** Alter the ORDERS2 table to reduce MINEXTENTS to 4. Has there been a reduction in the number of extents?

```
SQL> ALTER TABLE system.orders2
 2 STORAGE(MINEXTENTS 4);
Table altered.
SQL> SELECT COUNT(*)
 2 FROM dba_extents
 3 WHERE segment_name='ORDERS2'
 4 AND owner='SYSTEM';
COUNT(*)
-----
        10
```

**There is no change in the number of extents for the table.**

- 9 Truncate the ORDERS2 table, releasing space. How many extents does the table have now?

```
SQL> TRUNCATE TABLE system.orders2;
Table truncated.
SQL> SELECT COUNT(*)
       2 FROM dba_extents
       3 WHERE segment_name='ORDERS2'
       4 AND owner='SYSTEM';
COUNT(*)
-----
         4
```

- 10 (a) Run the script ins\_ord2.sql to insert some rows into the ORDERS2 table.

```
SQL> @$HOME/LABS/ins_ord2
1 row created.
...
Commit complete.
```

- (b) Release unused space from ORDERS2 and check the number of extents. Has space been released from the table? Why or why not?

```
SQL> ALTER TABLE system.orders2
       2 DEALLOCATE UNUSED;
Table altered.
SQL> SELECT COUNT(*)
       2 FROM dba_extents
       3 WHERE segment_name='ORDERS2'
       4 AND owner='SYSTEM';
COUNT(*)
-----
         4
```

**No space has been released because this command only releases space above the high-water mark; that is, in excess of MINEXTENTS.**

- 11 (a) For the order entry application, you now need to add a PRODUCTS table, which has the following columns:

Column	Data Type and Size
PROD_CODE	NUMBER(6)
DESCRIPTION	VARCHAR2(30)
PRICE	NUMBER(8,2)

Create this table in tablespace DATA02 using uniform extent sizes of 10 KB.

```
SQL> CREATE TABLE system.products
  2 (prod_code NUMBER(6),
  3 description VARCHAR2(30),
  4 price NUMBER(8,2))
  5 STORAGE ( INITIAL 10K NEXT 10K
  6           PCTINCREASE 0)
  7 TABLESPACE data02 ;
Table created.
```

- (b) Check the sizes of the extents for this table. What do you observe?

**Hint:** Check the DBA\_EXTENTS view to get the size.

```
SQL> COL segment_name FORMAT A12
SQL> SELECT segment_name, extent_id, blocks, bytes
  2 FROM dba_extents
  3 WHERE segment_name = 'PRODUCTS'
  4 AND owner='SYSTEM'
  5 /
```

SEGMENT_NAME	EXTENT_ID	BLOCKS	BYTES
PRODUCTS	0	25	102400

**The size of the extent is 100 KB and not the 10 KB specified while creating the table, because DATA02 tablespace has a MINIMUM EXTENT size of 100 KB.**



## Practice 12 Solutions

- 1 You are considering creating indexes on the NAME and REGION columns of the CUSTOMERS table. What types of index are appropriate for the two columns? Create the indexes, naming them CUST\_NAME\_IDX and CUST\_REGION\_IDX, respectively, and placing them in the appropriate tablespaces.

**Hint:** A B-tree index is suitable for a column with many distinct values, and a bitmap index is suitable for columns with only a few distinct values.

```
SQL> CREATE INDEX system.cust_name_idx
  2   ON system.customers(name)
  3   PCTFREE 30
  4   TABLESPACE indx01;
Index created.
SQL> CREATE BITMAP INDEX system.cust_region_idx
  2   ON system.customers(region)
  3   TABLESPACE indx01;
Index created.
```

- 2 Move the CUST\_REGION\_IDX index to another tablespace.

**Hint:** The index can be rebuilt specifying a different tablespace.

```
SQL> ALTER INDEX system.cust_region_idx REBUILD
  2   TABLESPACE indx;
Index altered.
```

- 3 Note the files and blocks used by the extents by CUST\_REGION\_IDX index.

File_id	Block_id	Blocks

**Hint:** Use the view DBA\_EXTENTS to get this information.

```
SQL> SELECT file_id, block_id, blocks
  2   FROM dba_extents
  3   WHERE segment_name='CUST_REGION_IDX'
  4   AND owner='SYSTEM';
FILE_ID  BLOCK_ID  BLOCKS
-----
      5         2         5
```

- 4 Re-create the CUST\_REGION\_IDX index without dropping and re-creating it, and retain it in the same tablespace as before. Does the new index use the same blocks that were used earlier?

**Hint:** Rebuild the index.

```
SQL> ALTER INDEX system.cust_region_idx REBUILD;
Index altered.
SQL>
SQL> SELECT file_id, block_id, blocks
       2 FROM dba_extents
       3 WHERE segment_name='CUST_REGION_IDX'
       4 AND owner='SYSTEM';
FILE_ID  BLOCK_ID  BLOCKS
-----
         5         7         5
```

The new index does not reuse the same space as seen from the location of the extent after rebuild. This is because Oracle server builds a temporary index, drops the old one, and renames the temporary index.

- 5 (a) Using *system* account, run the script `cr_numb.sql` to create and populate the NUMBERS table.

```
SQL> @$HOME/LABS/cr_numb
Table created.
PL/SQL procedure completed successfully.
```

(b) Query the table NUMBERS to find the number of distinct values in the two columns in the table.

```
SQL> SELECT COUNT(DISTINCT no),
       2          COUNT(DISTINCT odd_even)
       3 FROM system.numbers;
COUNT(DIST  COUNT(DIST
-----
    10000         2
```

(c) Using uniform extent sizes of 4 KB, create B-tree indexes NUMB\_OE\_IDX and NUMB\_NO\_IDX on the ODD\_EVEN and NO columns of the NUMBERS table, respectively, and check the total sizes of the indexes.

Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

**Hint:** Use PCTINCREASE=0 to create extents of equal size.

```
SQL> CREATE INDEX system.numb_oe_idx
  2  ON system.numbers(odd_even)
  3  TABLESPACE indx01
  4  STORAGE( INITIAL 4K NEXT 4k PCTINCREASE 0);
Index created.
SQL> CREATE INDEX system.numb_no_idx
  2  ON system.numbers(no)
  3  TABLESPACE indx01
  4  STORAGE( INITIAL 4K NEXT 4k PCTINCREASE 0);
Index created.
```

**Hint:** Check the total blocks allocated to the extents from DBA\_SEGMENTS.

```
SQL> SELECT segment_name, blocks
  2  FROM dba_segments
  3  WHERE segment_name LIKE 'NUMB%'
  4  AND segment_type='INDEX';
SEGMENT_NAME      BLOCKS
-----
NUMB_OE_IDX              40
NUMB_NO_IDX              45
```

(d) Once again, using uniform extent sizes of 4 K, create bitmap indexes NUMB\_OE\_IDX and NUMB\_NO\_IDX on the ODD\_EVEN and NO columns of the NUMBERS table, respectively, and check the total sizes of the indexes.

Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

What can you conclude about the relationship between cardinality and sizes of the two types of indexes?

**Hint:** The existing indexes need to be dropped before creating the new indexes.

```
SQL> DROP INDEX system.numb_oe_idx;
Index dropped.
SQL> DROP INDEX system.numb_no_idx;
Index dropped.
SQL> CREATE BITMAP INDEX system.numb_oe_idx
  2   ON system.numbers(odd_even)
  3   TABLESPACE indx01
  4   STORAGE( INITIAL 4K NEXT 4k PCTINCREASE 0);
Index created.
SQL> CREATE BITMAP INDEX system.numb_no_idx
  2   ON system.numbers(no)
  3   TABLESPACE indx01
  4   STORAGE( INITIAL 4K NEXT 4k PCTINCREASE 0);
Index created.
```

**Hint:** Now reexecute the query to check the sizes of the indexes.

```
SQL> SELECT segment_name, blocks
  2 FROM dba_segments
  3 WHERE segment_name LIKE 'NUMB%'
  4 AND segment_type='INDEX';
SEGMENT_NAME      BLOCKS
-----
NUMB_OE_IDX        2
NUMB_NO_IDX       71
```

It can be seen from the results that a bitmap index is compact for a low-cardinality column, while a B-tree index is compact for a high-cardinality column.

## Practice 13 Solutions

1 Examine the script `cr_cons.sql`. Run the script to create the constraints.

```
SQL> @$HOME/LABS/cr_cons
SQL> ALTER TABLE system.customers
  2 ADD (CONSTRAINT cust_pk PRIMARY KEY(cust_code)
  3       DEFERRABLE INITIALLY IMMEDIATE
  4       USING INDEX TABLESPACE indx01,
  5       CONSTRAINT cust_region_ck
  6       CHECK (region in ('East','West','North','South')));
Table altered.
SQL> ALTER TABLE system.orders
  2 ADD(CONSTRAINT ord_pk PRIMARY KEY(ord_id)
  3       USING INDEX TABLESPACE indx01,
  4       CONSTRAINT ord_cc_fk FOREIGN KEY(cust_code)
  5       REFERENCES customers(cust_code)
  6       DEFERRABLE INITIALLY IMMEDIATE,
  7       CONSTRAINT ord_dod_ck CHECK (date_of_dely >= ord_date));
Table altered.
SQL> ALTER TABLE system.products
  2 ADD CONSTRAINT prod_uk UNIQUE(prod_code)
  3       DEFERRABLE DISABLE;
Table altered.
```

2 Query the data dictionary to:

(a) Check for constraints, whether they are deferrable, and their status.

**Hint:** Use the `DBA_CONSTRAINTS` view to get this information.

```
SQL> SELECT constraint_name, table_name,
  2       constraint_type, deferrable, status
  3 FROM dba_constraints
  4 WHERE table_name IN
  5       ('PRODUCTS','ORDERS','CUSTOMERS')
  6 AND owner='SYSTEM';
```

CONSTRAINT_NAME	TABLE_NAME	C	DEFERRABLE	STATUS
CUST_PK	CUSTOMERS	P	DEFERRABLE	ENABLED
CUST_REGION_CK	CUSTOMERS	C	NOT DEFERRABLE	ENABLED
ORD_CC_FK	ORDERS	R	DEFERRABLE	ENABLED
ORD_DOD_CK	ORDERS	C	NOT DEFERRABLE	ENABLED
ORD_PK	ORDERS	P	NOT DEFERRABLE	ENABLED
PROD_UK	PRODUCTS	U	DEFERRABLE	DISABLED

6 rows selected.

(b) Check the names and types of indexes created to validate the constraints.

**Hint:** The indexes are only created for primary key and unique constraints and have the same name as the constraints.

```
SQL> SELECT index_name,table_name,uniqueness
  2 FROM dba_indexes
  3 WHERE index_name in
  4   (SELECT constraint_name
  5     FROM dba_constraints
  6     WHERE table_name IN ('PRODUCTS', 'ORDERS', 'CUSTOMERS')
  7     AND owner='SYSTEM'
  8     AND constraint_type in ('P','U'));
INDEX_NAME          TABLE_NAME          UNIQUENES
-----
CUST_PK             CUSTOMERS            NONUNIQUE
ORD_PK              ORDERS              UNIQUE
2 rows selected.
```

**c** Check which columns are used in the constraints created.

**Hint:** This information may be obtained from DBA\_CONS\_COLUMNS.

```
SQL> SELECT constraint_name,table_name, column_name
  2 FROM dba_cons_columns
  3 WHERE table_name IN ('PRODUCTS', 'ORDERS',
  4 'CUSTOMERS')
  4 AND owner='SYSTEM'
  5 ORDER BY table_name, constraint_name, position;
CONSTRAINT_NAME     TABLE_NAME          COLUMN_NAME
-----
CUST_PK             CUSTOMERS            CUST_CODE
CUST_REGION_CK      CUSTOMERS            REGION
ORD_CC_FK           ORDERS              CUST_CODE
ORD_DOD_CK          ORDERS              ORD_DATE
ORD_DOD_CK          ORDERS              DATE_OF_DELY
ORD_PK              ORDERS              ORD_ID
PROD_UK             PRODUCTS            PROD_CODE
7 rows selected.
```

**3** Insert two records with the following values into the PRODUCTS table:

PROD_CODE	DESCRIPTION	PRICE
100860	Ace Tennis Racket	36.20
100860	Ace Tennis Ball 3-Pack	2.40

```
SQL> INSERT INTO system.products
  2 VALUES(100860,'Ace Tennis Racket',36.20);
  3 /
1 row created.

SQL> INSERT INTO system.products
  2 VALUES(100860,'Ace Tennis Ball 3-Pack', 2.4);
1 row created.
```

- 4 Enable the unique constraint on the PRODUCTS table. Was it successful? Why or why not?

```
SQL> ALTER TABLE system.products
  2 ENABLE CONSTRAINT prod_uk;
ALTER TABLE system.products
*
ORA-02299: cannot enable (SYSTEM.PROD_UK) - duplicate keys found
```

The constraint cannot be enabled because there are rows that violate the constraint.

- 5 (a) Ensure that new rows added to the table do not violate the constraint on the PRODUCTS table.

**Hint:** This can be done by enabling the constraint NOVALIDATE.

```
SQL> ALTER TABLE products
  2 ENABLE NOVALIDATE CONSTRAINT prod_uk;
Table altered.
```

- (b) Query the data dictionary to verify the effect of the change.

```
SQL> SELECT constraint_name, table_name,
  2         constraint_type, validated, status
  3 FROM dba_constraints
  4 WHERE table_name = 'PRODUCTS'
  5 AND owner='SYSTEM';
```

CONSTRAINT_NAME	TABLE_NAME	C	VALIDATED	STATUS
PROD_UK	PRODUCTS	U	NOT VALIDATED	ENABLED

- (c) Test that the constraint disables inserts that violate the change by adding a row with the following values:

**PROD\_CODE: 100860**

**DESCRIPTION: Yellow Jersey Bicycle Helmet**

**PRICE: 30**

```
SQL> INSERT INTO products
  2 VALUES(100860,'Yellow Jersey Bicycle Helmet',
  3 30);
ORA-00001: unique constraint (SYSTEM.PROD_UK) violated
```

- 6** Take the necessary steps to identify existing constraint violations in the PRODUCTS table, modify product codes as needed, and guarantee that all existing as well as new data do not violate the constraint. (Assume that the table has several thousands of rows and it is too time-consuming to verify each row manually.)

**Hint:** Use the following steps:

Create the EXCEPTIONS table.

```
SQL> connect system/manager;
Connected.
SQL> @?/rdbms/admin/utlexcpt
Table created.
```

**Hint:** Run the command to enable the constraint and trap the exceptions.

```
SQL> ALTER TABLE system.products
  2 ENABLE CONSTRAINT prod_uk
  3 EXCEPTIONS INTO system.exceptions;
ALTER TABLE system.products
*
ORA-02299: cannot enable (SYSTEM.PROD_UK) - duplicate keys found
```

**Hint:** Use the ROWIDs in the EXCEPTIONS table to list the rows in the PRODUCTS table that violate the constraint. (Do not list LOB columns.)

```
SQL> SELECT rowid, prod_code, description
  2 FROM products
  3 WHERE rowid IN (SELECT row_id
  4                  FROM exceptions
  5                  WHERE table_name='PRODUCTS');
ROWID                PROD_CODE DESCRIPTION
-----
AAAAtRAAJAAAAADAAA   100860 Ace Tennis Racket
AAAAtRAAJAAAAADAAB   100860 Ace Tennis Ball 3-Pack
```



**Hint:** Rectify the errors.

```
SQL> UPDATE products
  2 SET prod_code='100861'
  3 WHERE rowid= (SELECT max(row_id)
  4                FROM exceptions
  5                WHERE table_name='PRODUCTS');
1 row Updated.
```

**Hint:** Enable the constraint.

```
SQL> TRUNCATE TABLE exceptions;
Table truncated.
SQL> ALTER TABLE products
  2 ENABLE CONSTRAINT prod_uk
  3 EXCEPTIONS INTO exceptions;
Table updated.
```

- 7** Run the script `ins_ocus1.sql` to insert rows into the table. Were the inserts successful? Roll back the changes.

```
SQL> @$HOME/LABS/ins_ocus1
SQL> SET ECHO on
Echo ON
SQL> INSERT INTO system.orders
  2 VALUES(800,'01-JAN-98','J01',NULL)
  3 /
ORA-02291: integrity constraint (SYSTEM.ORD_CC_FK) violated - par-
ent key not found
SQL>
SQL> INSERT INTO system.customers
  2 VALUES('J01','Sports Unlimited','West')
  3 /
1 row created.
```

**The insert to the ORDERS table fails because the customer code has not been inserted into the CUSTOMERS table.**

```
SQL> ROLLBACK;
Rollback complete.
```

- 8** Now examine the script `ins_ocus2.sql`. Notice that this script also performs the inserts in the same sequence. Run the script and check if it executes successfully.

```
SQL> @$HOME/LABS/ins_ocus2
SQL> ALTER SESSION SET CONSTRAINTS=deferred
    2 /
Session altered.
SQL> INSERT INTO system.orders
    2 VALUES(800,'01-JAN-98','J01',NULL)
    3 /
1 row created.
SQL> INSERT INTO system.customers
    2 VALUES('J01','Sports Unlimited','West')
    3 /
1 row created.
SQL> COMMIT;
Commit complete.
```

This script executes successfully because it defers the constraint checking until commit.

- 9** Truncate the CUSTOMERS table. Was it successful? Why or why not?

```
SQL> TRUNCATE TABLE system.customers;
TRUNCATE TABLE system.customers
      *
ORA-02266: unique/primary keys in table referenced by enabled
foreign keys
```

**The table cannot be truncated because there is a foreign key in ORDERS that references this table.**

## Practice 14 Solutions

Use the account *system* for all the questions in this practice.

- 1 (a) Run the script *ins\_item.sql* to insert data into the ITEMS table.

```
SQL> @$HOME/LABS/ins_item
1 row created.
...
```

- (b) Create a table ITEMS2 as a copy of the ITEMS table.

```
SQL> CREATE TABLE items2
2 TABLESPACE data01
3 AS
4 SELECT * FROM items;
Table created.
```

- (c) Note the last file/block number currently used by a row in the table

**Hint:** Query the ROWIDs using the functions in the DBMS\_ROWID package.

```
SQL> SELECT max(dbms_rowid.rowid_relative_fno(rowid)
2 || '.' || dbms_rowid.rowid_block_number(rowid))
3 AS "File.Block"
4 FROM system.items2;
File.Block
-----
8.494
```

- 2 (a) Delete all rows in ITEMS2 and insert a new row with the following values into the table:

ORD\_ID: 200

PROD\_CODE: 200000

QTY: 20

```
SQL> DELETE FROM system.items2;
271 rows deleted.
SQL> INSERT INTO system.items2
      2 VALUES(200,200000,20);
1 row created.
```

(b) Check and note the file/block number for the new row.

```
SQL> SELECT dbms_rowid.rowid_relative_fno(rowid) || '.'
      2      || dbms_rowid.rowid_block_number(rowid)
      3 AS "File.Block"
      4 FROM system.items2;
File.Block
-----
8.494
```

**3** (a) Use direct-load insert to copy data into the ITEMS2 table from the ITEMS table.

```
SQL> INSERT /*+APPEND */ INTO system.items2
      2 SELECT * FROM system.items;
271 rows created.
SQL> COMMIT;
Commit complete.
```

(b) Check the file/block numbers of the rows with ORD\_ID<>200. What can you observe about the location of these rows?

```
SQL> SELECT DISTINCT
      2      dbms_rowid.rowid_relative_fno(rowid) || '.'
      3      || dbms_rowid.rowid_block_number(rowid)
      4      AS "File.Block"
      5 FROM items2
      6 WHERE ord_id <> 200;
File.Block
-----
8.495
8.496
```

**The rows inserted using direct-load insert have not reused blocks that were used earlier, while a normal insert uses these blocks.**

- 4** Examine the scripts `ulcase1.sql`, `ulcase1.ct1`, `ulcase2.ct1`, and `ulcase2.dat`. These are some of the standard loader demonstration files supplied with the demonstration Oracle8i Enterprise Edition. As user `system`, perform the following steps to try two load runs and get acquainted with using SQL\*Loader:

(a) Run the script `ulcase1.sql` to create the DEPT and EMP tables.

```
SQL> @$HOME/LABS/ulcase1$
```

- (b) Run SQL\*Loader to load data into the DEPT table using the control file `ulcase1.ctl`. Examine the log file generated, and query the DEPT table to check the data loaded.

```
$ sqlldr system control=ulcase1.ctl
```

```
Password:
```

```
SQL*Loader: Release 8.1.5.0.0 - Production on Thu Jun 17  
09:47:08 1999
```

```
(c) Copyright 1998 Oracle Corporation. All rights reserved.
```

```
Commit point reached - logical record count 7
```

```
$ sqlplus system/manager
```

```
SQL> SELECT *
```

```
2 FROM system.dept;
```

DEPTNO	DNAME	LOC
12	RESEARCH	SARATOGA
10	ACCOUNTING	CLEVELAND
11	ART	SALEM
13	FINANCE	BOSTON
21	SALES	PHILA.
22	SALES	ROCHESTER
42	INT'L	SAN FRAN

```
7 rows selected.
```

(c) Run SQL\*Loader again to load data into the EMP table using the control file ulcase2.ct1. Notice that this run uses an input data file to load data. Examine the log file generated and query the EMP table to check that the data loaded.

```
$ sqlldr system control=ulcase2.ct1
Password:

SQL*Loader: Release 8.1.5.0.0 - Production on Thu Jun 17
09:47:08 1999

(c) Copyright 1998 Oracle Corporation. All rights reserved.

Commit point reached - logical record count 7

$ sqlplus system/manager
SQL> SELECT *
  2 FROM system.emp;
EMPNO ENAME      JOB          MGR HIREDATE          SAL      COMM      DEPTNO
-----
7782 CLARK      MANAGER      7839          2572.5          10
7839 KING        PRESIDENT          5500          10
7934 MILLER     CLERK        7782          920            10
7566 JONES       MANAGER      7839          3123.75         20
7499 ALLEN      SALESMAN     7698          1600           300         30
7654 MARTIN    SALESMAN     7698          1312.5         1400         30
7658 CHAN      ANALYST      7566          3450           20
7 rows selected.
```

5 (a) Check the number of extents and total number of blocks in the ITEMS2 table.

**Hint:** Query the DBA\_SEGMENTS view.

```
SQL> SELECT extents, blocks
  2 FROM dba_segments
  3 WHERE segment_name='ITEMS2'
  4 AND owner='SYSTEM';
EXTENTS      BLOCKS
-----
1            5
```

(b) Allocate an extent to the table manually, and note the number of extents and blocks now.\_\_\_\_\_. Because the extent has just been created, the new extent is empty.

```
SQL> ALTER TABLE system.items2
      2 ALLOCATE EXTENT;
Table altered.
SQL> SELECT EXTENTS, BLOCKS
      2 FROM dba_segments
      3 WHERE segment_name='ITEMS2'
      4 AND owner='SYSTEM';
EXTENTS    BLOCKS
-----
      2      10
```



## Practice 15 Solutions

- 1 You want to reorganize the ITEMS2 table using export and import. Check the number and size of the extents in the ITEMS2 table after import. What can you infer about the behavior of export and import on space allocation?

**Hint:** Perform the following steps:

Use table-level export to do this:

```
$ exp system/manager tables=items2

Export: Release 8.1.5.0.0 - Production on Thu Jun 17 10:57:50 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.
Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 - Pro-
duction
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production
Export done in US7ASCII character set and WE8ISO8859P1 NCHAR char-
acter set server uses WE8ISO8859P1 character set (possible charset
conversion)
About to export specified tables via Conventional Path ...
. . exporting table                ITEMS2                272 rows
exported
Export terminated successfully without warnings.
```

**Hint:** Drop the ITEMS2 table.

```
SQL> SELECT EXTENTS, BLOCKS
       2 FROM dba_segments
       3 WHERE segment_name='ITEMS2'
       4 AND owner='SYSTEM';

EXTENTS    BLOCKS
-----
         2         10

SQL> DROP TABLE system.items2;

Table dropped.
```

**Hint:** Import the ITEMS2 table.

```
$ imp system/manager tables=items2

Import: Release 8.1.5.0.0 - Production on Thu Jun 23 11:02:20
1999

(c) Copyright 1997 Oracle Corporation. All rights reserved.

Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 -
Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

Export file created by EXPORT:V08.01.05 via conventional path
..
. importing SYSTEM's objects into SYSTEM
. . importing table                "ITEMS2"          272 rows
imported
Import terminated successfully without warnings.
```

**Hint:** Check the number and size of the extents in the ITEMS2 table using DBA\_SEGMENTS.

```
SQL> SELECT extents, blocks
2 FROM dba_segments
3 WHERE segment_name='ITEMS2'
4 AND owner='SYSTEM';
EXTENTS    BLOCKS
-----
1          10
1 row selected.
```

By default, exporting a table and importing it will result in the table having one extent, which is as large as the original size of the segment.

**2** You need to move several indexes from one tablespace to another. This practice uses one index to show how this can be done.

(a) Create an index named ITEM\_OID\_IDX in the tablespace DATA01 on the ORD\_ID column of the ITEMS2 table.

```
SQL> CREATE INDEX system.item_oid_idx
      2  ON system.items2(ord_id)
      3  TABLESPACE data01;
Index created.
```

(b) Using export and import, move the index to the tablespace INDX01.

**Hint:** Use the following steps:

Export the table with indexes, but without rows.

```
$ exp system/manager tables=items2 rows=n

Export: Release 8.1.5.0.0 - Production on Thu Jun 17 12:51:02
1999

(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 -
Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production
Export done in US7ASCII character set and WE8ISO8859P1 NCHAR
character set
Note: table data (rows) will not be exported

About to export specified tables via Conventional Path ...
. . exporting table                                ITEMS2
Export terminated successfully without warnings.
```

**Hint:** Drop the index.

```
SQL> DROP INDEX system.item_oid_idx;
Index dropped.
```

**Hint:** Import from the output of the previous export and create a script to build the index.

```
$ imp system/manager tables=items2 indexfile=cr_itix.sql
Import: Release 8.1.5.0.0 - Production on Thu Jun 17 12:24:42
1999

(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 -
Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

Export file created by EXPORT:V08.01.05 via conventional path
Import terminated successfully without warnings.
```

**Hint:** Edit the index file created in the previous step to change the tablespace to INDX01 and run the script to re-create the index.

```
SQL> @cr_itix
SQL>
SQL> REM CREATE TABLE "SYSTEM"."ITEMS2" ("ORD_ID" NUMBER(3, 0),
"PROD_CODE"
SQL> REM NUMBER(6, 0), "QTY" NUMBER(4, 0)) PCTFREE 10 PCTUSED
40 INITRANS 1
SQL> REM MAXTRANS 255 LOGGING STORAGE(INITIAL 40960 NEXT 24576
MINEXTENTS 1
SQL> REM MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1 FREELIST
GROUPS 1
SQL> REM BUFFER_POOL DEFAULT) TABLESPACE "DATA01" ;
SQL> CONNECT SYSTEM;
Password:
Connected.
SQL> CREATE INDEX "SYSTEM"."ITEM_OID_IDX" ON "ITEMS2" ("ORD_ID"
) PCTFREE 10
2 INITRANS 2 MAXTRANS 255 STORAGE (INITIAL 10240 NEXT 10240
MINEXTENTS 1
3 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1
BUFFER_POOL
4 DEFAULT) TABLESPACE "INDX01" LOGGING ;
Index created.
```

## Practice 16 Solutions

- 1 Enable password management by using the `utlpwdmg.sql` script.

```
SQL> connect sys/oracle as sysdba;
Connected.
SQL> @?/rdbms/admin/utlpwdmg
Function created.
Profile altered.
```

- 2 Try to change the password of user SCOTT to SCOTT. What happens?

```
SQL> ALTER USER scott
      2 IDENTIFIED BY scott;
ALTER USER scott
*
ORA-28003: password verification for the specified password
failed
ORA-20001: Password same as user
```

**3** Make sure that the following applies to users assigned the DEFAULT profile:

- After two login attempts, the account should be locked.
- The password should expire after 30 days.
- The same password should not be reused for at least one minute.
- The account should have a grace period of five days to change an expired password.

Ensure that the requirements given have been implemented.

**Hints:**

- Use the ALTER PROFILE command to change the default profile limits.
- Query the data dictionary view DBA\_PROFILES to verify the result.

```
SQL> ALTER PROFILE default LIMIT
  2  FAILED_LOGIN_ATTEMPTS 2
  3  PASSWORD_LIFE_TIME 30
  4  PASSWORD_REUSE_TIME 1/1440
  5  PASSWORD_GRACE_TIME 5;
Profile altered.
SQL> SELECT resource_name, limit FROM dba_profiles
  2  WHERE profile='DEFAULT' AND
  3  resource_type='PASSWORD';
```

RESOURCE_NAME	LIMIT
FAILED_LOGIN_ATTEMPTS	2
PASSWORD_LIFE_TIME	30
PASSWORD_REUSE_TIME	.0006
PASSWORD_REUSE_MAX	UNLIMITED
PASSWORD_VERIFY_FUNCTION	VERIFY_FUNCTION
PASSWORD_LOCK_TIME	.0006
PASSWORD_GRACE_TIME	5

```
7 rows selected.
```

- 4** Log in to user `SYSTEM` supplying an invalid password. Try this twice, then log in again, this time supplying the correct password.

```
SQL> connect system/manger@db01
ERROR:
ORA-01017: invalid username/password; logon denied
Warning: You are no longer connected to ORACLE.
SQL> connect system/man1@db01
ERROR:
ORA-01017: invalid username/password; logon denied
SQL> connect system/man2@db01
ERROR:
ORA-28000: the account is lockedSQL> connect system/manger@db01
ERROR:
ORA-01017: invalid username/password; logon denied
Warning: You are no longer connected to ORACLE.
SQL> connect system/man1@db01
ERROR:
ORA-01017: invalid username/password; logon denied
SQL> connect system/man2@db01
ERROR:
ORA-28000: the account is locked
```

- 5** Ensure that the `SYSTEM` can reconnect.

**Hint:** Execute the `ALTER USER` command to unlock the `SYSTEM` account.

```
SQL> CONNECT sys/oracle;
SQL> ALTER USER system
  2 ACCOUNT UNLOCK;
User altered.
```



**6** Disable password checks for the DEFAULT profile.**Hint:** Execute the ALTER PROFILE command to disable the password checks.

```
SQL> ALTER PROFILE default LIMIT
  2 FAILED_LOGIN_ATTEMPTS UNLIMITED
  3 PASSWORD_LIFE_TIME UNLIMITED
  4 PASSWORD_REUSE_TIME UNLIMITED
  5 PASSWORD_REUSE_MAX UNLIMITED
  6 PASSWORD_VERIFY_FUNCTION NULL
  7 PASSWORD_LOCK_TIME UNLIMITED
  8 PASSWORD_GRACE_TIME UNLIMITED;
Profile altered.
```

## Practice 17 Solutions

- 1 Create user `Bob` with a password of `ALONG`. Make sure that any objects and temporary segments created by Bob are not created in the system tablespace. Also, ensure that Bob can log in and create objects up to one megabyte in size in `DATA01` and `INDX01` tablespaces. If you are not using Oracle Enterprise Manager, run the script `bob.sql`.

**Hint:** Ensure that the temporary tablespace is assigned.

```
SQL> CREATE USER bob
  2  IDENTIFIED BY along
  3  DEFAULT TABLESPACE data01
  4  TEMPORARY TABLESPACE temp
  5  QUOTA 1M ON data01
  6  QUOTA 1M ON indx01;
User created.
SQL> @$HOME/LABS/bob
GRANT create session TO bob;
Grant succeeded.
```

- 2 (a) Create a user `Kay` with a password of `MARY`. Make sure that any objects and sort segments created by Kay are not created in the system tablespace.

```
SQL> CREATE USER kay
  2  IDENTIFIED BY mary
  3  DEFAULT TABLESPACE data01
  4  TEMPORARY TABLESPACE temp;
User created.
```

- (b) Copy the `ORDERS2` table from the `SYSTEM` schema to Kay's account. Call the new table `ORDERS`.

**Hint:** Kay needs a quota on her default tablespace before objects can be created in her schema.

```
SQL> ALTER USER kay
2   QUOTA UNLIMITED ON data01;
User altered.
SQL> CREATE TABLE kay.orders
2 AS
3 SELECT * FROM system.orders2;
Table created.
```

- 3** Display the information on Bob and Kay from the data dictionary.

**Hint:** This can be obtained by querying DBA\_USERS.

```
SQL> SELECT username, default_tablespace,
2         temporary_tablespace
3   FROM dba_users
4  WHERE username IN ('BOB', 'KAY');
USERNAME DEFAULT_TABLESPACE  TEMPORARY_TABLESPACE
-----
KAY      DATA01              TEMP
BOB      DATA01              TEMP
```

- 4** From the data dictionary, display the information on the amount of space that Bob can use in tablespaces.

**Hint:** This can be obtained by querying DBA\_TS\_QUOTAS.

```
SQL> SELECT * FROM dba_ts_quotas WHERE username = 'BOB';
TABLESPACE_NAM USERNAME      BYTES MAX_BYTES    BLOCKS MAX_BLOCKS
-----
INDX01          BOB              0  1048576         0      256
DATA01          BOB              0  1048576         0      256
```

**5 (a)** As user BOB change his temporary tablespace. What happens? Why?

```
SQL> CONNECT bob/along;
Connected.
SQL> ALTER USER bob
      2 TEMPORARY TABLESPACE data01;
ALTER USER bob
*
ORA-01031: insufficient privileges
```

**(b)** As Bob, change his password to SAM.

```
SQL> CONNECT bob/along;
Connected.
SQL> ALTER USER bob
      2 IDENTIFIED BY sam;
User altered.
```

**6** As system, remove Bob's quota on his default tablespace.

```
SQL> CONNECT system/manager
Connected.
SQL> ALTER USER bob QUOTA 0 ON data01;
User altered.
```

**7** Remove Kay's account from the database.

**Hint:** Because Kay owns tables, you need to use the CASCADE option.

```
SQL> DROP USER kay CASCADE;
User dropped.
```

**8** Bob has forgotten his password. Assign him a password of OLINK and require that Bob change his password the next time he logs on.

```
SQL> ALTER USER bob
      2 IDENTIFIED BY olink
      3 PASSWORD EXPIRE;
User altered.
```

## Practice 18 Solutions

- 1 As system, create user `kay` and give her the capability to log on to the database and create objects in her schema.

**Hint:** Kay needs the CREATE SESSION and CREATE TABLE privileges.

```
SQL> CONNECT system/manager
Connected.
SQL> CREATE USER kay
  2 IDENTIFIED BY "abcd12"
  3 DEFAULT TABLESPACE data01
  4 TEMPORARY TABLESPACE temp
  5 QUOTA 1M ON data01;
User created.
SQL> GRANT create session, create table TO kay;
Grante succeeded.
```

- 2 (a) Connect as `kay` and use the script `ulcase1.sql` to create tables `EMP` and `DEPT`.

```
SQL> CONNECT kay/abcd12
Connected.
SQL> @$HOME/LABS/ulcase1
```

- (b) Connect as `system` and copy the data from `SYSTEM.EMP` to Kay's `EMP` table. Verify that records have been inserted.

```
SQL> CONNECT system/manager
Connected.
SQL> INSERT INTO kay.emp
  2 SELECT * FROM emp;
7 rows created.
SQL> SELECT * FROM kay.emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7782	CLARK	MANAGER	7839		2572.5		10
7839	KING	PRESIDENT			5500		10
7934	MILLER	CLERK	7782		920		10
7566	JONES	MANAGER	7839		3123.75		20
7499	ALLEN	SALESMAN	7698		1600	300	30
7654	MARTIN	SALESMAN	7698		1312.5	1400	30
7658	CHAN	ANALYST	7566		3450		20

```
7 rows selected
```

(c) As `system`, give Bob the ability to select from Kay's EMP table. What happens and why?

```
SQL> CONNECT system/manager
Connected.
SQL> GRANT select ON kay.emp
      2 TO bob;
GRANT select ON kay.emp
                *
ORA-01031: insufficient privileges
```

SYSTEM can query Kay's table using the SELECT ANY TABLE privilege, but only Kay or any other user who received the privilege from Kay with GRANT OPTION can grant this privilege to others.

- 3 Reconnect as `kay` and give Bob the ability to select from Kay's EMP table. Also, enable Bob to give the select capability to other users. Examine the data dictionary views that record these actions.

**Hint:** Query the view `DBA_TAB_PRIVS` to see the privileges.

```
SQL> CONNECT kay/abcd12
Connected.
SQL> GRANT select ON emp
  2 TO bob WITH GRANT OPTION;
Grant succeeded.
SQL>
SQL> CONNECT system/manager
Connected.
SQL> SELECT * FROM dba_tab_privs
  2 WHERE grantee='BOB';
```

GRANTEE	OWNER	TABLE_NAME	GRANTOR	PRIVILEGE	GRA
BOB	KAY	EMP	KAY	SELECT	YES

- 4 Create user `Todd` with the capability to log on to the database

```
SQL> CREATE USER todd IDENTIFIED BY "abcd1?";
User created.
SQL> GRANT create session TO todd;
Grant succeeded.
```

- 5 (a) As Bob, enable Todd to access Kay's EMP table. Give Bob the new password `sam`.

```
SQL> CONNECT bob/olink
ERROR:
ORA-28001: the password has expired
Changing password for bob
Old password: *****
New password: ***
Retype new password: ***
Password changed
Connected. SQL> GRANT select ON kay.emp TO todd;
Grant succeeded.
```

(b) As Kay, remove Bob's privilege to read Kay's EMP table.

```
SQL> CONNECT kay/abcd12
Connected.
SQL> REVOKE select ON emp FROM bob;
Revoke succeeded.
```

(c) As Todd, query Kay's EMP table. What happens and why?

```
SQL> CONNECT todd/abcd1?
Connected.
SQL> SELECT * FROM kay.emp;
SELECT * FROM kay.emp
          *
ORA-00942: table or view does not exist
```

Since revoking object privileges has a cascading effect, Todd, who received the privilege from Bob, lost the SELECT privilege on Kay's table when Bob's privilege was revoked in the previous question.

**6** (a) Enable Kay to create tables in any schema. As Kay, create the table DEPT in Bob's schema as a copy of KAY.DEPT. What happened and why?

```
SQL> CONNECT system/manager
Connected.
SQL> GRANT create any table TO kay;
Grant succeeded.
SQL> CONNECT kay/abcd12
Connected.
SQL> CREATE TABLE bob.dept
  2 AS
  3 SELECT * FROM dept;
SELECT * FROM dept
          *
ORA-01536: space quota exceeded for tablespace 'DATA01'
```

Bob's quota on his default tablespace was removed in Practice 17, question 6.



(b) As `system`, examine the data dictionary view `DBA_TABLES` to check the result.

```
SQL> CONNECT system/manager
Connected.
SQL> SELECT owner, table_name
   2 FROM dba_tables
   3 WHERE table_name IN ('EMP', 'DEPT');
OWNER  TABLE_NAME
-----
SCOTT  DEPT
SCOTT  EMP
SYSTEM EMP
SYSTEM DEPT
KAY    DEPT
KAY    EMP

6 rows selected.
```

- 7 Give Kay the ability to start up and shut down the database without the ability to create a new database.

**Hint:** Give Kay the `SYSOPER` privilege.

```
SQL> CONNECT sys/coracle AS SYSDBA
Connected.
SQL> GRANT sysoper TO kay;
Grant succeeded.
```

## Practice 19 Solutions

- 1 Examine the data dictionary view and list the system privileges of the RESOURCE role.

**Hint:** This information is available from DBA\_SYS\_PRIVS.

```
SQL> CONNECT system/manager
Connected.
SQL> SELECT *
      2 FROM dba_sys_privs
      3 WHERE grantee = 'RESOURCE';
GRANTEE PRIVILEGE ADM
-----
RESOURCE CREATE CLUSTER NO
RESOURCE CREATE INDEXTYPE NO
RESOURCE CREATE OPERATOR NO
RESOURCE CREATE PROCEDURE NO
RESOURCE CREATE SEQUENCE NO
RESOURCE CREATE TABLE NO
RESOURCE CREATE TRIGGER NO
RESOURCE CREATE TYPE NO
8 rows selected.
```

- 2 Create a role called DEV, that enables a user to create a table, create a view, and select from Kay's EMP table.

```
SQL> CREATE ROLE dev;
Role created.
SQL> GRANT create table, create view TO dev;
Grant succeeded.
SQL> CONNECT kay/abcd12
Connected.
SQL> GRANT select ON emp TO dev;
Grant succeeded.
SQL> CONNECT system/manager
Connected.
SQL> GRANT dev TO bob;
Grant succeeded.
```

- 3 (a) Assign the RESOURCE and DEV roles to Bob, but make only the RESOURCE role automatically enabled when he logs on.

**Hint:** Use the ALTER USER command to specify the default role.

```
SQL> GRANT dev, resource TO bob;
Grant succeeded.
SQL> ALTER USER bob
      2 DEFAULT ROLE resource;
User altered.
```

- (b) Give Bob the ability to read all the data dictionary information.

**Hint:** Assign the SELECT\_CATALOG\_ROLE to Bob.

```
SQL> GRANT select_catalog_role TO bob;
Grant succeeded.
```

- 4 Bob needs to check the rollback segments that are currently used by the instance. Connect as Bob and list the rollback segments used.

**Hint:** Bob needs the SELECT\_CATALOG\_ROLE to check the rollback segments, but since this is not one of his default roles, it must be enabled first.

```
SQL> CONNECT bob/sam
Connected.
SQL> SET ROLE select_catalog_role;
Role set.
SQL> SELECT segment_name
      2 FROM dba_rollback_segs
      3 WHERE status='ONLINE';
SEGMENT_NAME
-----
SYSTEM
R01
R02
R03
R04
```

- 5** As `system`, try to create a view `EMP_VIEW` on Kay's `EMP` table. What happens and why?

```
SQL> connect system/manager
Connected.
SQL> CREATE VIEW kay_emp AS SELECT * FROM kay.emp;
CREATE VIEW kay_emp AS SELECT * FROM kay.emp
                                *
ORA-00942: table or view does not exist
```

**SYSTEM** can read Kay's table using privileges acquired by the **DBA** role, but a privilege obtained through a role cannot be used to create views.

## Practice 20 Solutions

- 1 Check the database and the national character set.

**Hint:** Query the data dictionary view NLS\_DATABASE\_PARAMETERS to check the character sets.

```
SQL> SELECT parameter,value
       2 FROM nls_database_parameters
       3 WHERE parameter LIKE '%CHARACTERSET';
```

PARAMETER	VALUE
NLS_CHARACTERSET	WE8ISO8859P1
NLS_NCHAR_CHARACTERSET	WE8ISO8859P1

- 2 Which are valid values for the database character set?

**Hint:** Query the data dictionary view V\$NLS\_VALID\_VALUES to list the valid values.

```
SQL> SELECT value FROM v$nls_valid_values
       2 WHERE parameter='CHARACTERSET';
```

VALUE

US7ASCII

WE8DEC

WE8HP

...

ZHT32TRISFIXED

233 rows selected.

**3** Make sure that all dates in this session are displayed using a four-digit year.**Hint:** Set the parameter NLS\_DATE\_FORMAT.

```
SQL> ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
Session altered.
SQL> SELECT SYSDATE FROM dual;
SYSDATE
-----
18-06-1999
```

**4** Define the NLS\_LANG on server and client side to enable 8-bit character encoding scheme. Run the script nls.sql connected as user sys and display the rows. Then export the table. Import the table to user Bob using the character set US7ASCII.

Query the table; what happened and why? (optional)

```
$NLS_LANG=German_Germany.WE8ISO8859P1
$export NLS_LANG
SQL> CONNECT sytem/manager@db01
Connected.
SQL> @$HOME/LABS/nls
Table created.
1 row created.
1 row created.
SQL> SELECT * FROM nls;
C
-----
\
-
SQL> EXIT
exp system/manager tables=nls
Export: Release 8.1.5.0.0 - Production on Thu Jun 17 12:51:02
1999
..
Export done in WE8ISO8859P1 character set and WE8ISO8859P1 NCHAR
character set
Note: table data (rows) will not be exported

About to export specified tables via Conventional Path ...
. . exporting table                                NLS 2 rows exported.
Export terminated successfully without warnings.
```

```
$NLS_LANG=American_America.US7ASCII
$export NLS_LANG
imp system/manager fromuser=system touser=bob tables=nls
Import: Release 8.1.5.0.0 - Production on Thu Jun 17 12:24:42
1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8i Enterprise Edition Release 8.1.5.0.0 -
Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

Export file created by EXPORT:V08.01.05 via conventional path
import done in US7ASCII character set and WE8ISO8859P1 character
set
import server uses WE8ISO8859P1 character set (possible charset
conversion)
export client uses WE8ISO8859P1 character set (possible charset
conversion)
. . importing table "NLS" 2 rows imported
Import terminated successfully without warnings.
SQL> CONNECT bob/sam
Connected.
SQL> SELECT * FROM nls;
C
-----
U
?
```





---

D

---

**Practice Solutions  
for Oracle Enterprise  
Manager**

## **Practice 1 Solutions**

All the solutions for this practice are the same as in Appendix C, “Practice Solutions for SQL\*Plus.” Please refer to Appendix C for the answers.

## Practice 2 Solutions

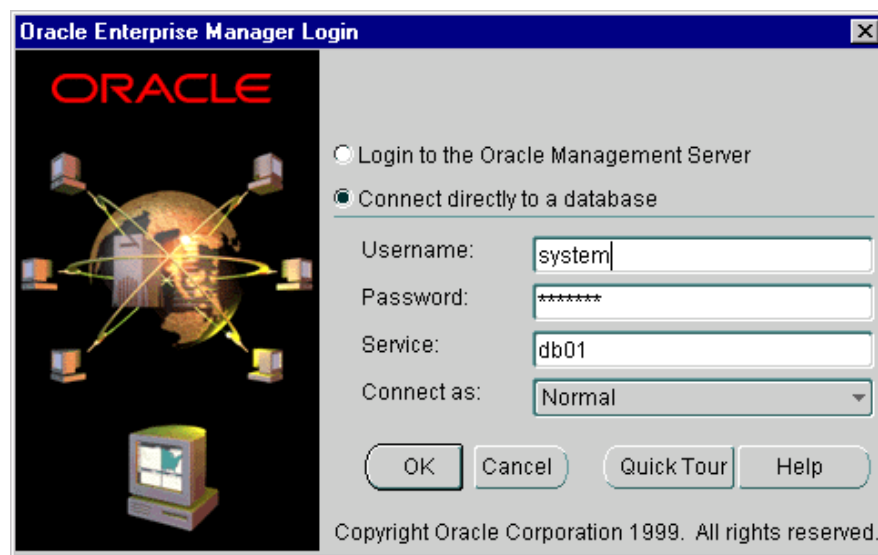
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet



#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

## Practice 3 Solutions

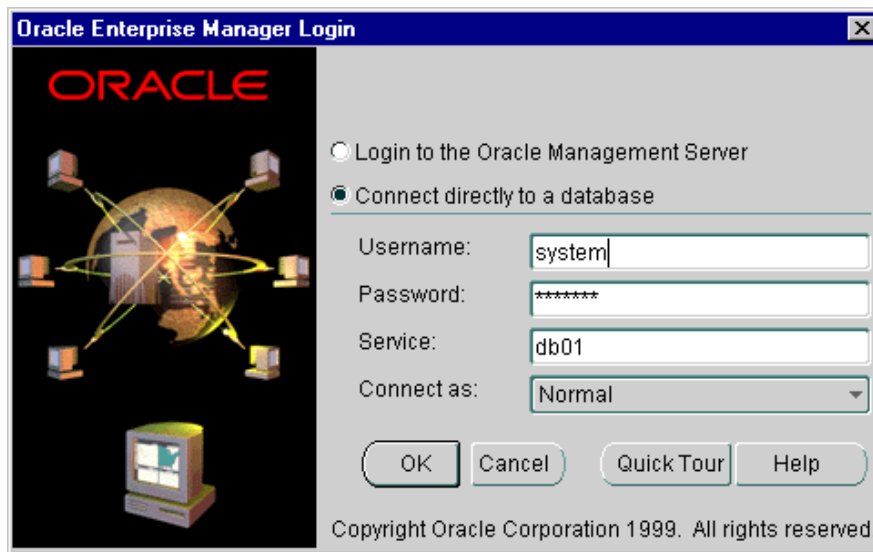
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers unless a specific Oracle Enterprise Manager solution is used.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet



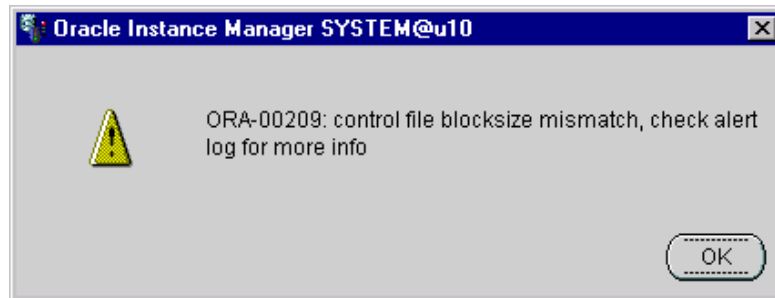
#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.
- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C.
- 4 The solution is the same as in Appendix C.
- 5 The solution is the same as in Appendix C.

**6** Try to change the database block size. What happened?

- a** (N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Instance Manager
- b** Connect using username *sys*, password *oracle*, service name *db01*, and connect as *sysdba*. Connect directly to the database. Do not connect to an OMS.
- c** Select Database—>Shutdown.
- d** Click Yes in the dialog box.
- e** Select Immediate in the Shutdown Options.
- f** Click OK.
- g** Edit *init<SID>.ora*, and add `DB_BLOCK_SIZE=8192`.
- h** Use Instance Manager.
- i** Select Database—>Startup.
- j** In the Startup dialog box, select Open and specify the name of the local parameter file.



- k** Shut down the instance.
- l** Edit the *init<SID>.ora* to comment out `DB_BLOCK_SIZE`.
- m** Start up and open the database. Select the Option button Open and click Apply.

**7** The solution is the same as in Appendix C.

- 8** The solution is the same as in Appendix C.
- 9** Start a SQL\*Plus session. Connect as user SCOTT and insert rows in the table EMP. Open a second session and try to shut down the database transactional. What happens?

```
SQL> connect scott/tiger
Connected.
SQL> INSERT INTO emp (empno, ename, deptno)
      2 VALUES (1,'Vijay',10);
1 row processed.
```

- a** Use Instance Manager
- b** Select Database—>Shutdown
- c** Click Yes in the dialog box
- d** Select Transactional in the Shutdown Options
- e** Click OK

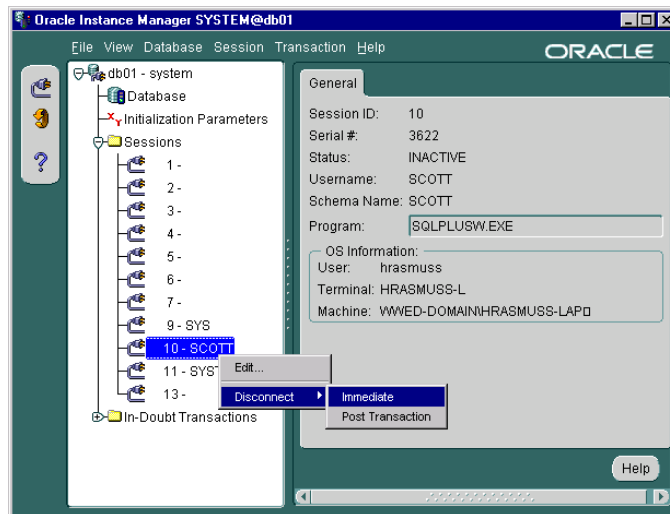
```
Rollback Scott's transaction.
```

**The Oracle server waits for SCOTT's transaction to end before shutting down. Wait for the instance to shut down at the second session. Then bring it back up.**

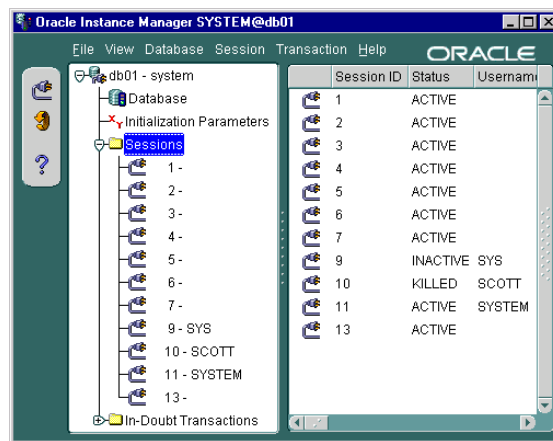
- 10** The solution is the same as in Appendix C.

- 11** Ensure that there are at least two sessions open, one session as user SCOTT and one as user SYS. Enable the restricted session, verify this, and ensure that only the database administrator SYS is connected.

- a Use Instance Manager.
- b Select Session—>Restrict from the menu and click OK in the dialog box.
- c Expand the Sessions folder.
- d Select user SCOTT and choose Disconnect—>Immediate from the right mouse menu.



- e Select the Sessions folder and examine the status of SCOTT's session.



## Practice 4 Solution

All the solutions for this practice are the same as in Appendix C. Please refer to Appendix C for the answers.

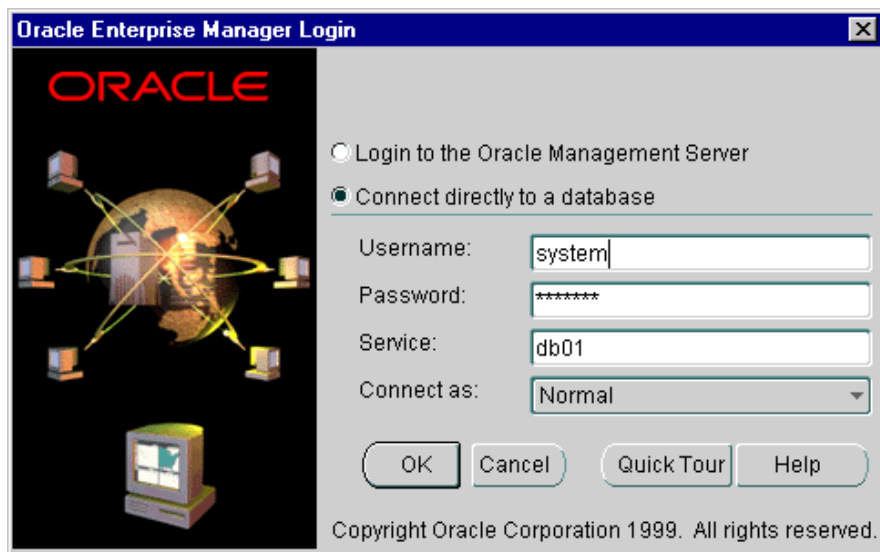
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet



#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.



## Practice 5 Solutions

All the solutions for this practice are the same as in Appendix C. Please refer to Appendix C for the answers.

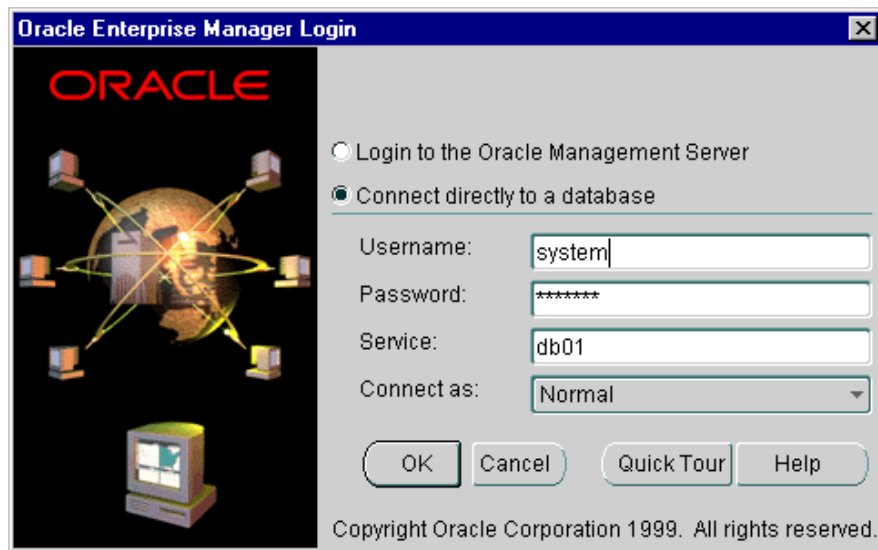
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet



#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

## Practice 6 Solutions

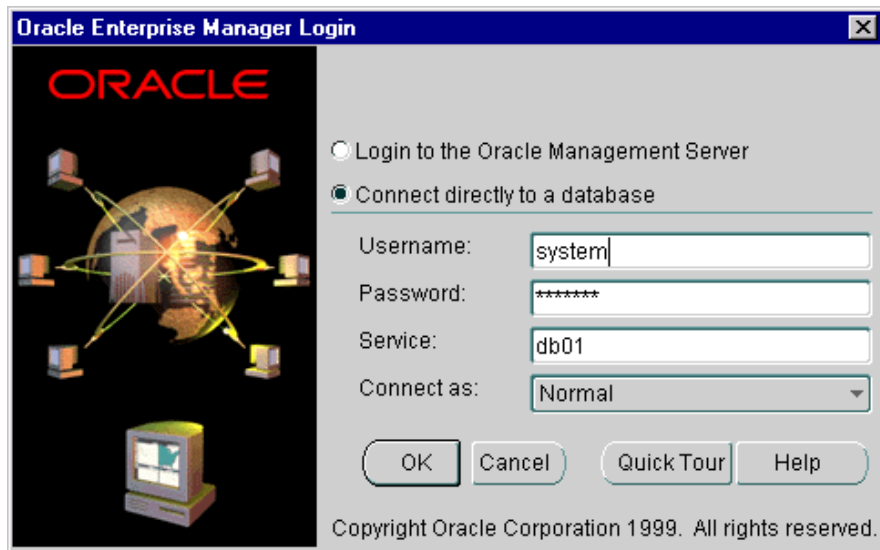
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers unless a specific Oracle Enterprise Manager solution is used.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet

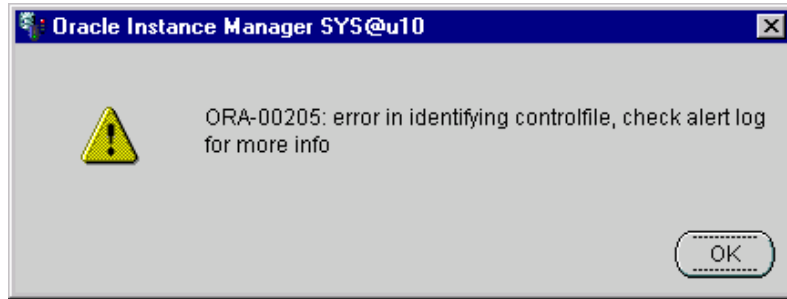


#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.
- 2 Try to start the database without any control files. (You can simulate this by changing the name of the control file in the parameter file.) What happens?

After renaming the file on the server, use Instance Manager to restart the database

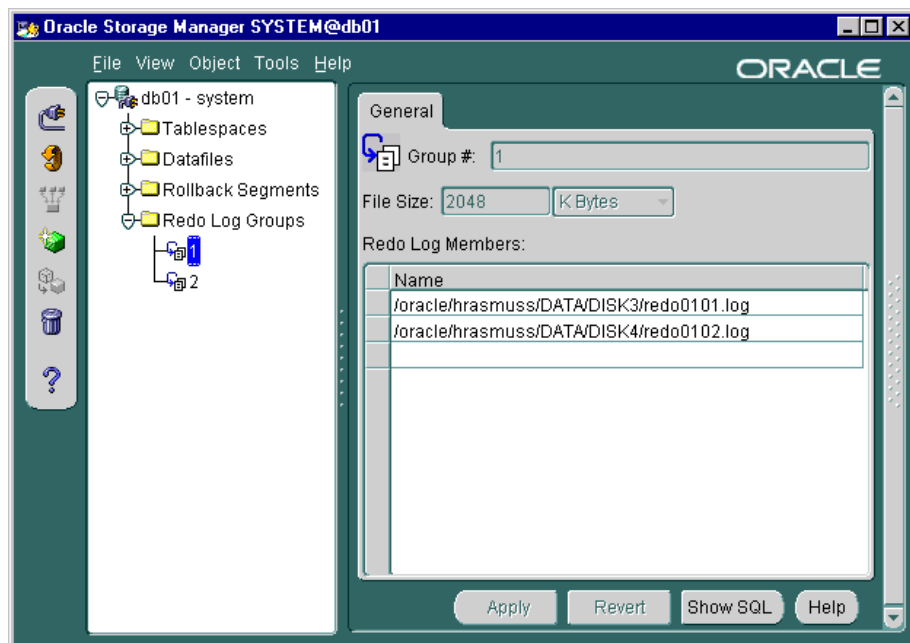


- 3 The solution is the same as in Appendix C.
- 4 The solution is the same as in Appendix C.

## Practice 7 Solutions

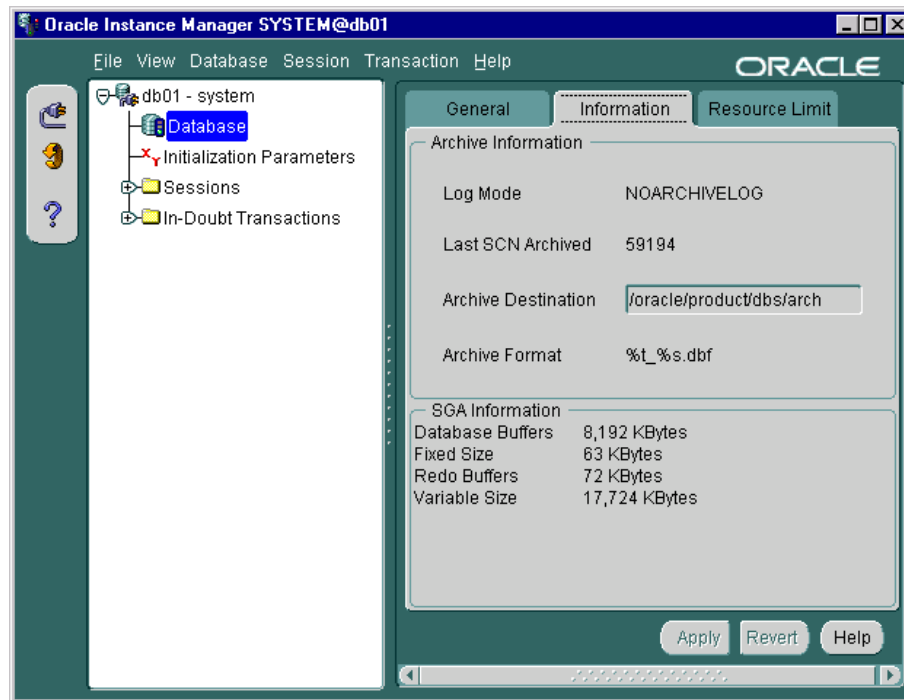
- 1 List the number and the location of the existing log files and display the number of redo log file groups and members your database has.

- a Select Programs —>Oracle - *EMV2 Home*—>DBA Management Pack —>Storage Manager.
- b Connect directly to a database in the login dialog box.
- c Expand Redo Log Groups folder to see the number of groups
- d Select a group to see the number of members in the group and their locations.



2 In which database mode is your database configured? Is automatic archiving enabled?

- a Use Oracle Enterprise Manager.
- b Select Programs —>Oracle - *EMV2 Home*—>DBA Management Pack—>Instance Manager.
- c Connect directly to a database in the login dialog box.
- d Select the Database in the navigator tree and click on the Information tab in the right window pane.

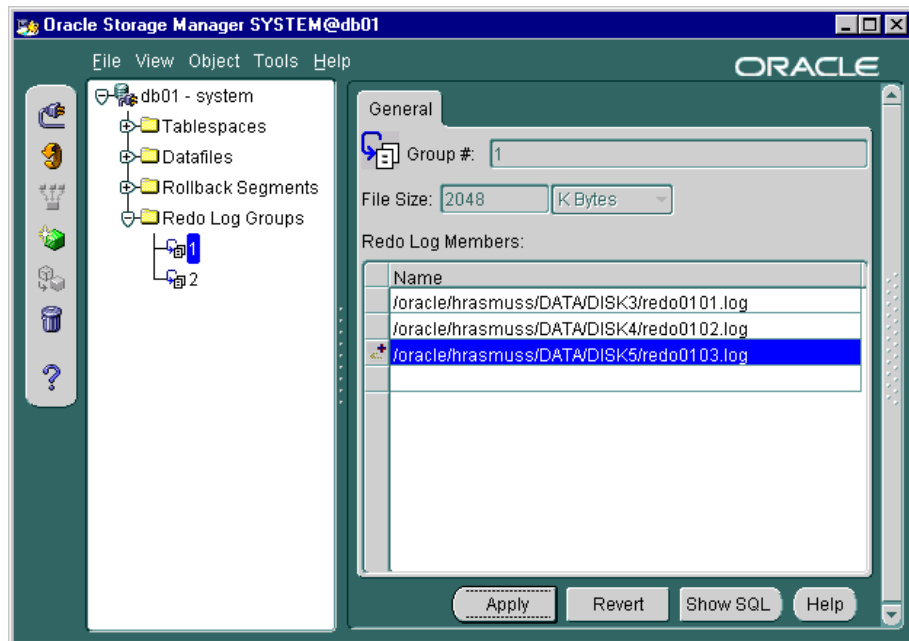


- e Select Initialization Parameters in the navigator tree and view the parameters in the right window pane. The parameter LOG\_ARCHIVE\_START should show on the Basic Tuning tab.

- 3** Add a redo log member to each group in your database located on DISK3, using the following naming conventions:

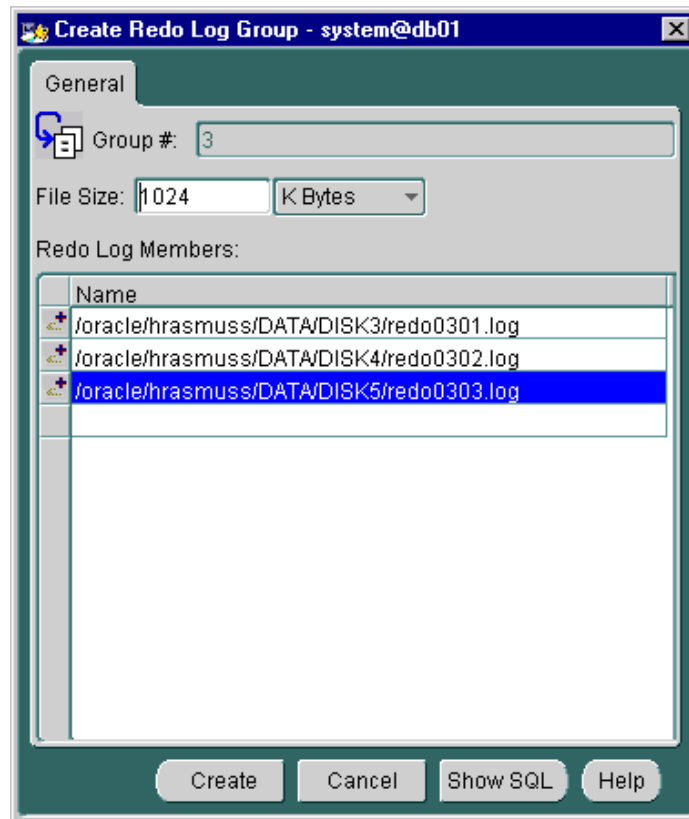
If group 1 has two existing files called `redo0101.log` and `redo0102.log`, then add a new member called `redo0103.log` on DISK3. Verify the result.

- a** Use Storage Manager.
- b** Expand the Redo Log Groups in the navigator tree.
- c** Select the redo log group and enter a new log file in the spreadsheet in the right window pane. Click Apply.



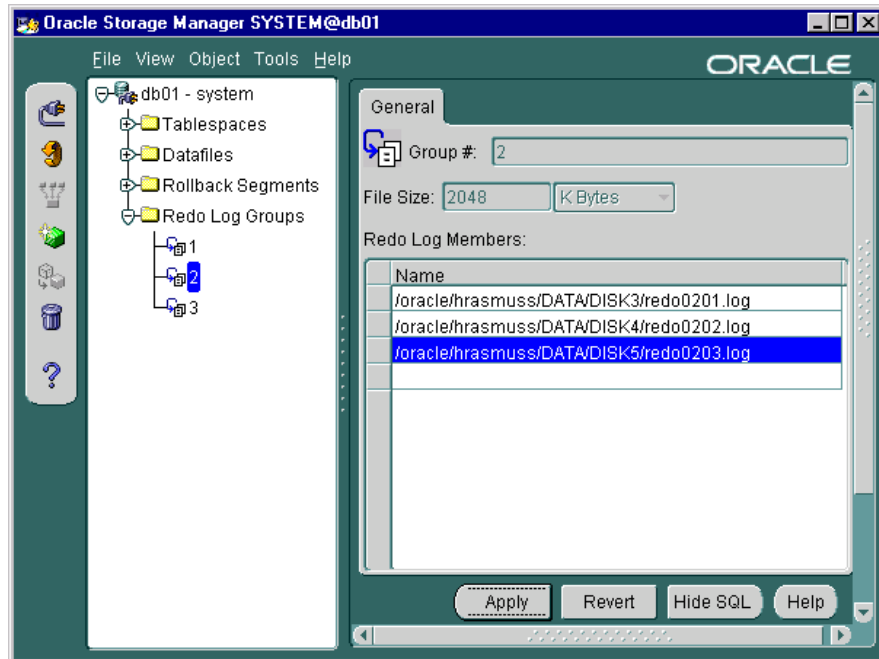
- 4 Create a new redo log group located in directories DISK3, DISK4, and DISK5, and verify the existence of the new group.

- a Use Storage Manager.
- b Select the Redo Log Groups in the navigator tree and choose Create from the right mouse menu.
- c Enter the name and location of the redo log members. Click Create.



- 5** Relocate the members *redo0103.log* and *redo0203.log* (see step 3) and locate them in the directory DISK5 from DISK3.

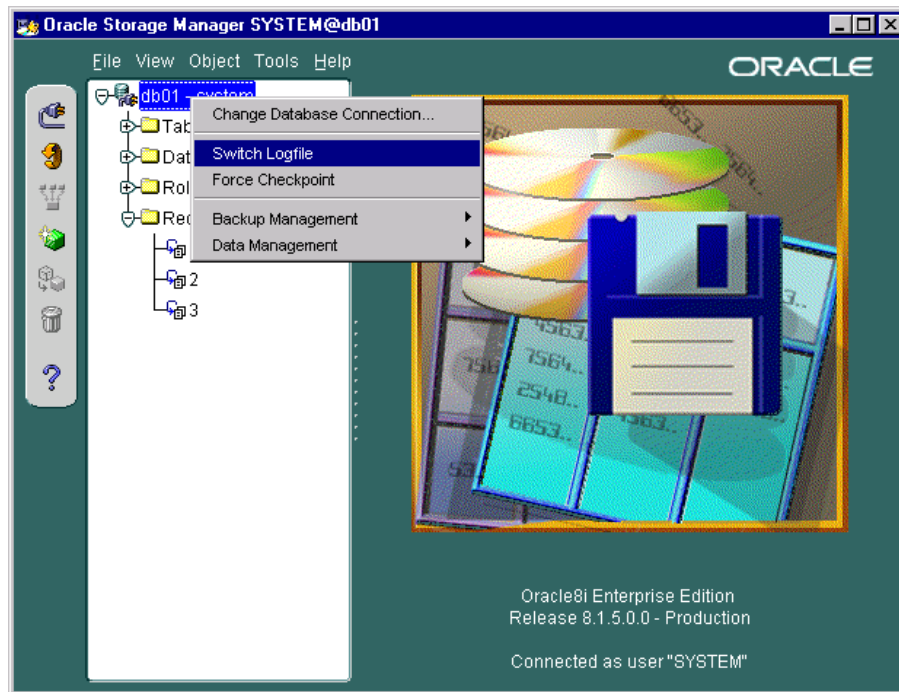
- a** Use Storage Manager.
- b** Select the Redo Log Groups in the navigator tree.
- c** Change the location of the redo log members. Click Apply. (Remember to copy the files to the new destination before renaming the files. Please refer to Appendix C for the answers.)





**6** Remove the log group created in step 4.

- d** Use Storage Manager.
- e** Select Redo Log Groups.
- f** If group 3 is active, select your working database and choose Switch Logfile from the right mouse menu.



- g** Click OK in the message box.
- h** Select group 3 in the navigator tree.
- i** Select Object—>Remove.
- j** Click Yes in the Storage Manager dialog box.
- k** Log in to the server and execute:  

```
$rm $HOME/DATA/DISK3/redo0301.log  
$rm $HOME/DATA/DISK4/redo0302.log  
$rm $HOME/DATA/DISK5/redo0303.log
```

**7** Resize all online redo log files to 1024 KB.

- a** Use Storage Manager to add two log groups with two members each.
- b** Switch log files until the groups 1 and 2 become inactive.
- c** Drop the log groups 1 and 2.
- d** Log in to the server and execute:  

```
$rm $HOME/DATA/DISK3/redo0101.log  
$rm $HOME/DATA/DISK3/redo0201.log  
$rm $HOME/DATA/DISK4/redo0102.log  
$rm $HOME/DATA/DISK4/redo0202.log  
$rm $HOME/DATA/DISK5/redo0103.log  
$rm $HOME/DATA/DISK5/redo0203.log
```
- e** Select each log group to verify that the files have been created properly.

---

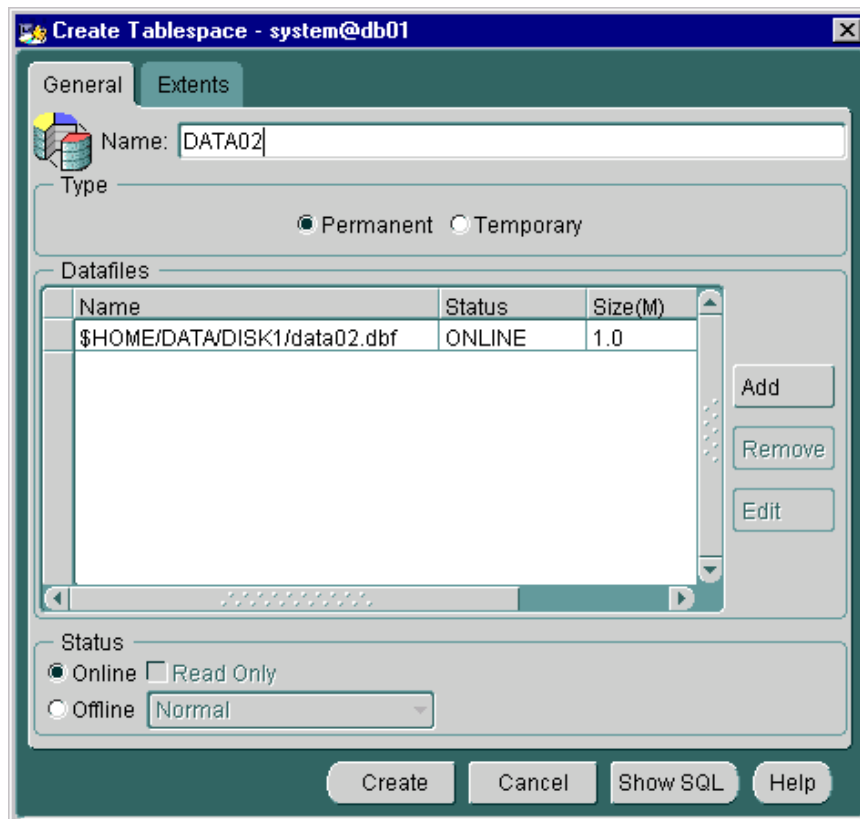
## Practice 8 Solutions

- 1 Create permanent tablespaces with the following names and storage:
  - a) DATA01 for tables with default storage
  - c) DATA02 for large objects with default storage (Ensure that every used extent size in the tablespace is multiple of 100 KB.)
  - d) INDX01 for indexes with the default storage (Enable automatic extension of 500 KB when more extents are required.)
  - e) RONLY for read-only tables with the default storage

Display the information from the data dictionary.

Tablespace Name	Subdirectory	Data File Location (Size)
DATA01	DISK4	data01.dbf (2M)
DATA02	DISK5	data02.dbf (1M)
INDX01	DISK3	indx01.dbf (1M)
RONLY	DISK1	ronly.dbf (1M)

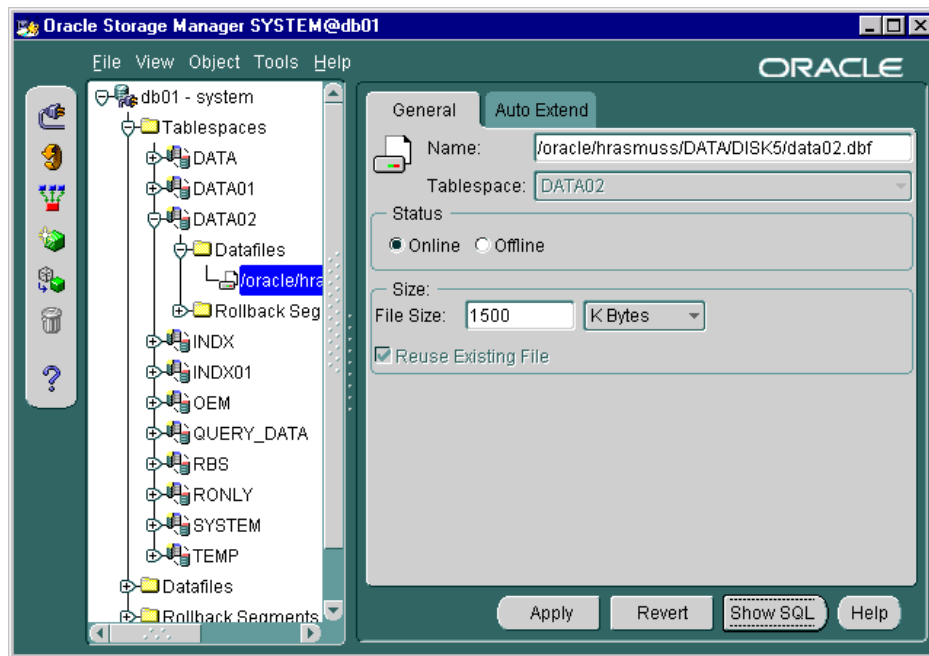
- a** Use Storage Manager
- b** Select Programs —>Oracle - *EMV2 Home*—>DBA Management Pack —>Storage Manager.
- c** Connect directly to a database in the login dialog box.
- d** Select the Tablespaces folder in the navigator tree and select Create from the right mouse button menu.
- e** Enter Name in the General tab of the Property Sheet and click Add.
- f** Enter the name and size of the data file in the Create Datafile dialog and click OK.
- g** Click Create.



- h** Repeat the above steps for all the tablespaces.

**2** Allocate 500 KB more to the tablespace DATA02 and verify the result.

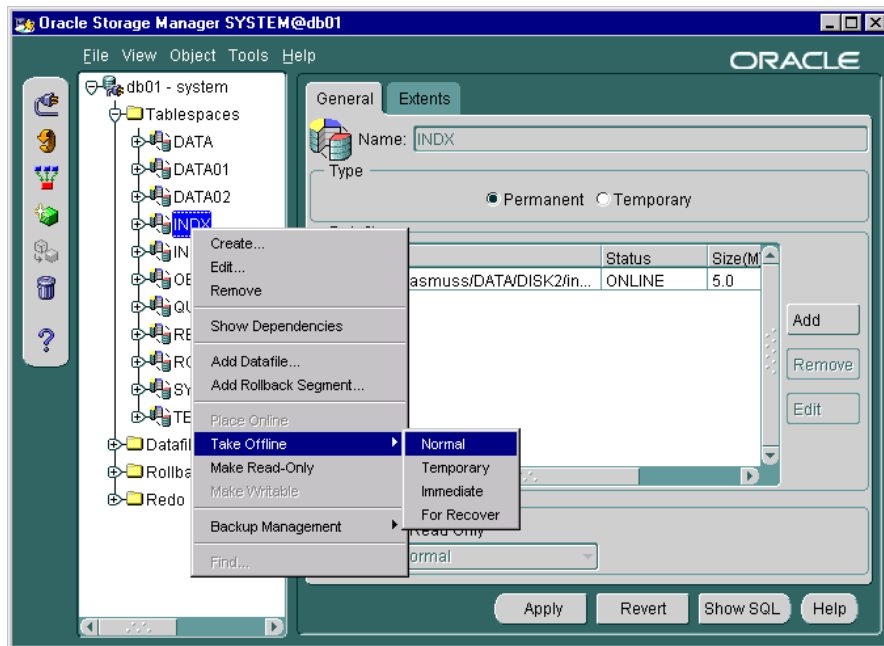
- a** Use Storage Manager.
- b** Expand the Tablespaces folder.
- c** Expand the data file DATA02.
- d** Expand the Datafiles folder.
- e** Select the data file.
- f** Specify 1500 KB for file size in the General page.



- g** Click Apply.
- h** Verify the size from the General page or by selecting Datafiles.

### 3 Relocate the INDX01 tablespace and move it to DISK6.

- a Use Storage Manager.
- b Expand Tablespaces folder.
- c Select the tablespace INDX01. Select Tablespace—>Take Offline—>Normal from the right mouse menu, and click Yes in the dialog box.

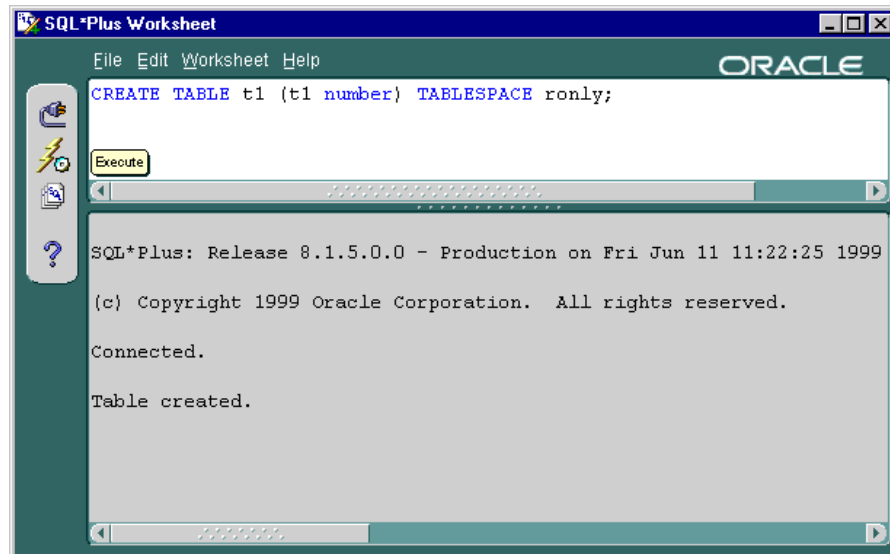


- d Confirm from the General page that the tablespace is offline.
- e Log in to the server and execute:  

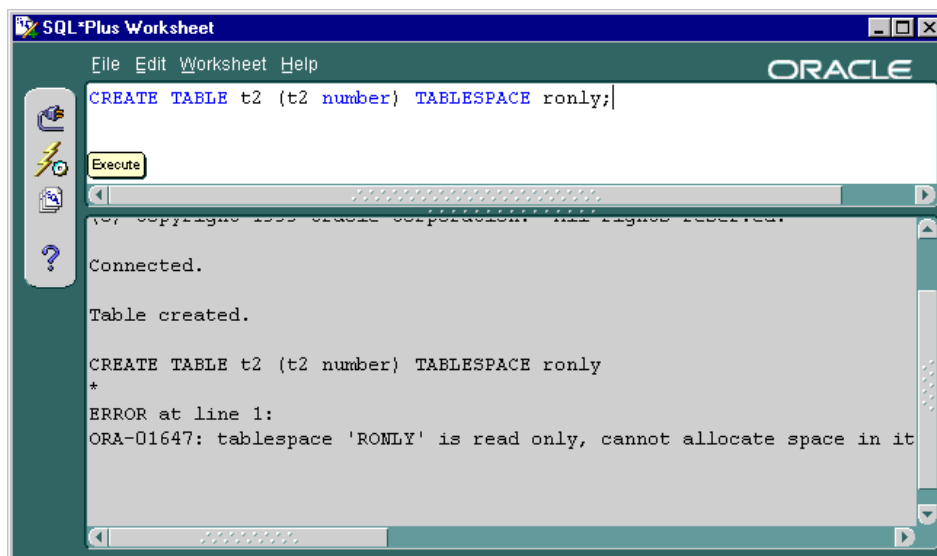
```
$mv $HOME/DATA/DISK3/indx01.dbf $HOME/DATA/DISK6/indx01.dbf
```
- f Expand Datafiles.
- g Select the file belonging to INDX01 tablespace.
- h Change the name in the General page.
- i Click Apply.
- j Expand Tablespaces.
- k Select INDX01 tablespace.
- l Select Object—>Place Online.
- m Click Yes in the Oracle Storage Manager dialog box.

- 4 Change the RONLY tablespace to read-only after creating a table in this tablespace. Attempt to create an additional table and drop the table. What happens and why?

- a Use SQL\*Plus Worksheet to run the following command:



- b Use Storage Manager.  
c Expand the Tablespaces folder and select the RONLY tablespace.  
d Select Tablespace—>Make Read-Only from the right mouse menu. Click Yes in the Oracle Storage Manager dialog box.  
e Verify from the General page that the tablespace is read-only.  
f Use SQL\*Plus Worksheet to run the following commands:



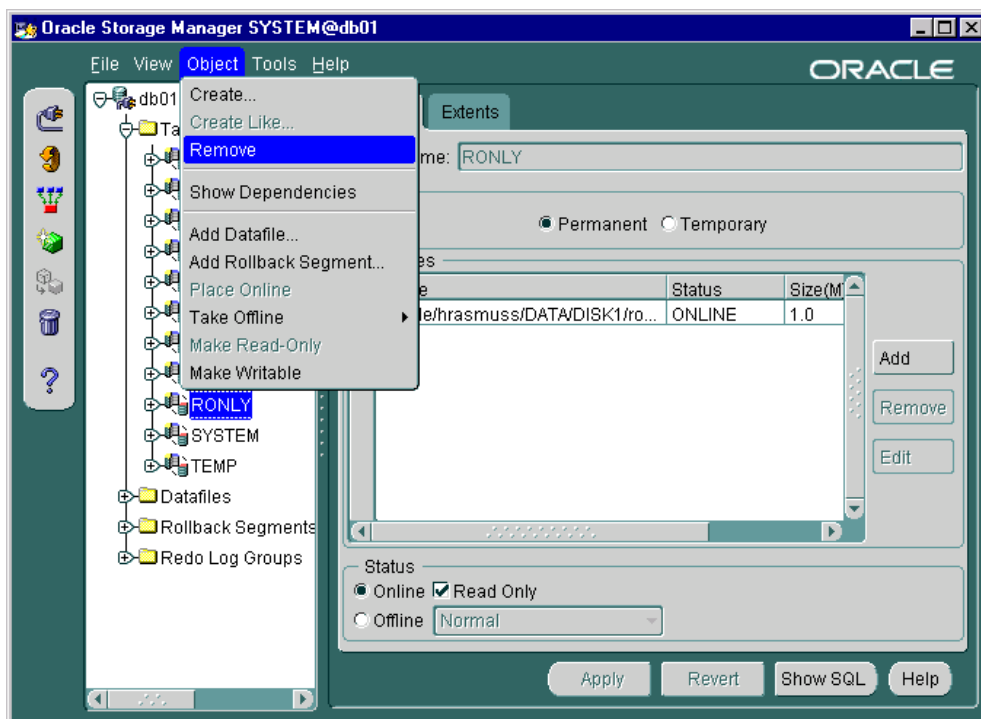
- g Use SQL\*Plus Worksheet to drop table t1;

**5** Drop the tablespace RONLY and verify it.

**Hint**

- Execute the DROP TABLESPACE... to remove the tablespace.
- Delete the operating system files.
- Query the dynamic performance view V\$TABLESPACE to verify the result.

- a Use Storage Manager.
- b Expand the Tablespaces folder.
- c Select the RONLY tablespace.
- d Select Object—>Remove.



- e Log in to the server and execute the command:

```
$rm $HOME/DATA/DISK1/ronly.dbf
```

- 6** The solution is the same as in Appendix C.
- 7** The solution is the same as in Appendix C.
- 8** The solution is the same as in Appendix C.



## Practice 9 Solutions

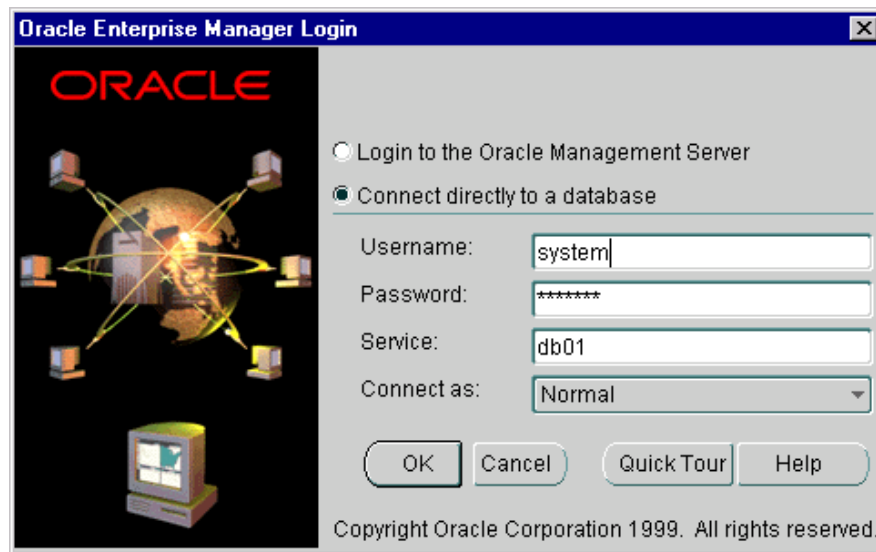
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers unless a specific Oracle Enterprise Manager solution is used.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet.



#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.
- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C.
- 4 The solution is the same as in Appendix C.
- 5 The solution is the same as in Appendix C.
- 6 The solution is the same as in Appendix C.
- 7 The solution is the same as in Appendix C.
- 8 The solution is the same as in Appendix C.

## Practice 10 Solutions

Before starting this practice, make sure you that run the script `$HOME/LABS/alt_sysrol.sql` as user `SYSTEM`.

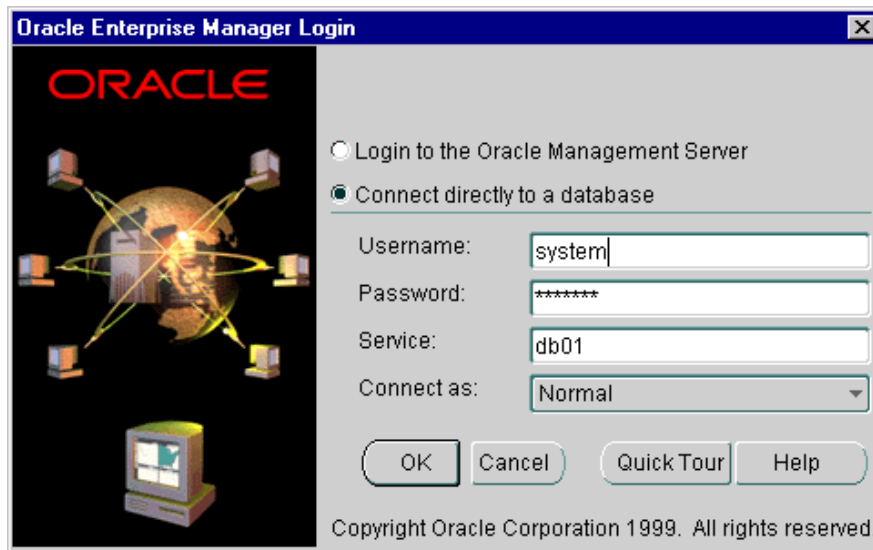
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers unless a specific Oracle Enterprise Manager solution is used.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



#### Step Two

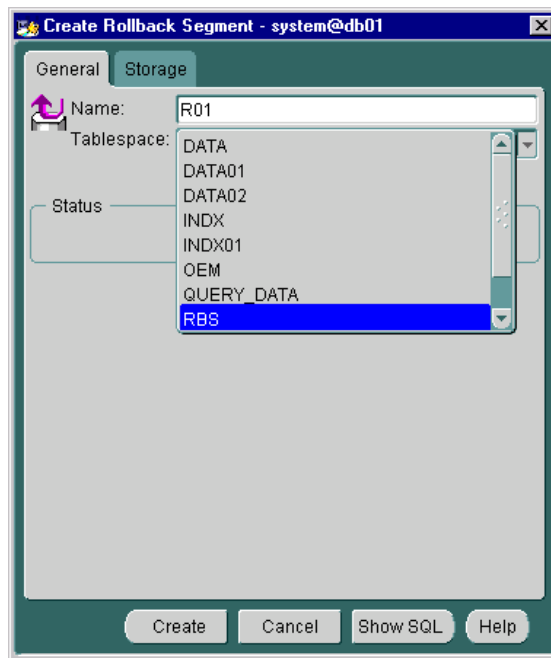
Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.

- 2 You are going to be running an online order entry application on your database. The orders will be entered using 15 client stations, which have a very high volume of activity in the mornings. Create an appropriate number of rollback segments in the database. (Assume default sizing for the purpose of this exercise.)

**Hint:** Using a ratio of one rollback segment for every four transactions, you will need to create four rollback segments.

- a Use Storage Manager
- b Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Storage Manager.
- c Select the Rollback Segments folder and select Create from the right mouse button menu.
- d Specify the name of the rollback segment and select RBS as the tablespace in the General page.



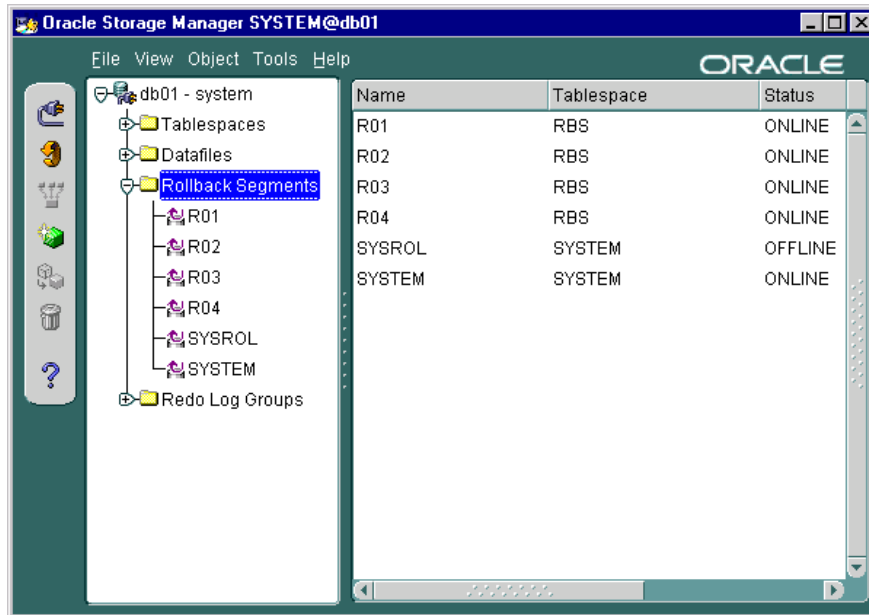
- e Click Create.
- f The second, third, and fourth rollback segments can be created by selecting the rollback segment R01 and choosing Create Like from the right mouse button menu.

**3** The solution is the same as in Appendix C.

**4** Verify which rollback segments in the system are available for use by transactions.

**Hint:** This information can be obtained from DBA\_ROLLBACK\_SEGS view.

- a** Use Storage Manager.
- b** Select the Rollback Segments folder.
- c** Check for all rollback segments with the status ONLINE.

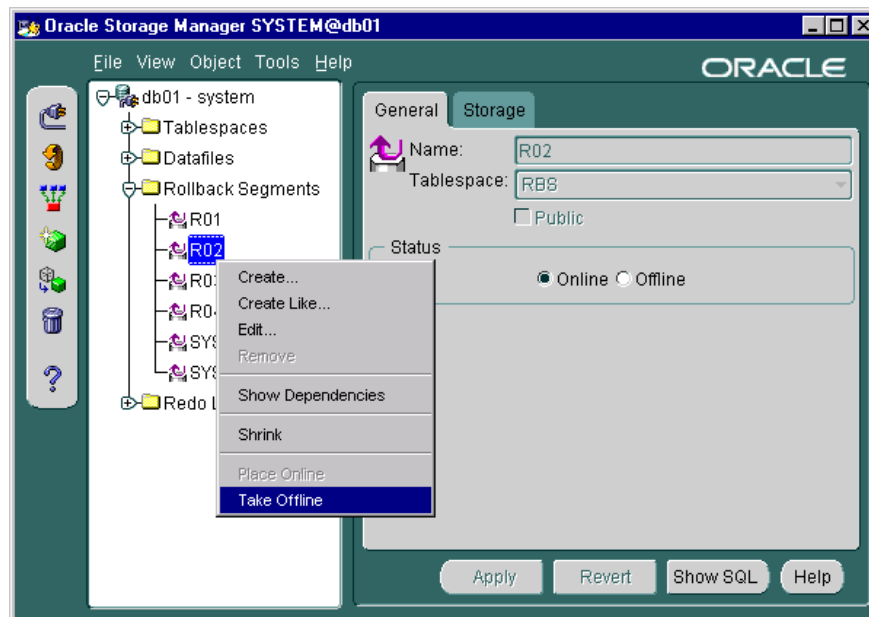


- 5 The solution is the same as in Appendix C.
- 6 Invoke SQL\*Plus Worksheet. Run the script *ins\_emp.sql*. Using a separate session, take the rollback segment tablespace offline.

**Hint:** The script starts another transaction that inserts a row into the EMP table. Because both transactions are using the rollback segments in this tablespace, you need to use the following steps:

First, take all rollback segments in the tablespace offline to prevent new transactions from using the rollback segments in the tablespace.

- a Use Storage Manager.
- b Expand Rollback Segments.
- c Select at least one non-SYSTEM rollback segment. Select Take Offline from the right mouse button menu.



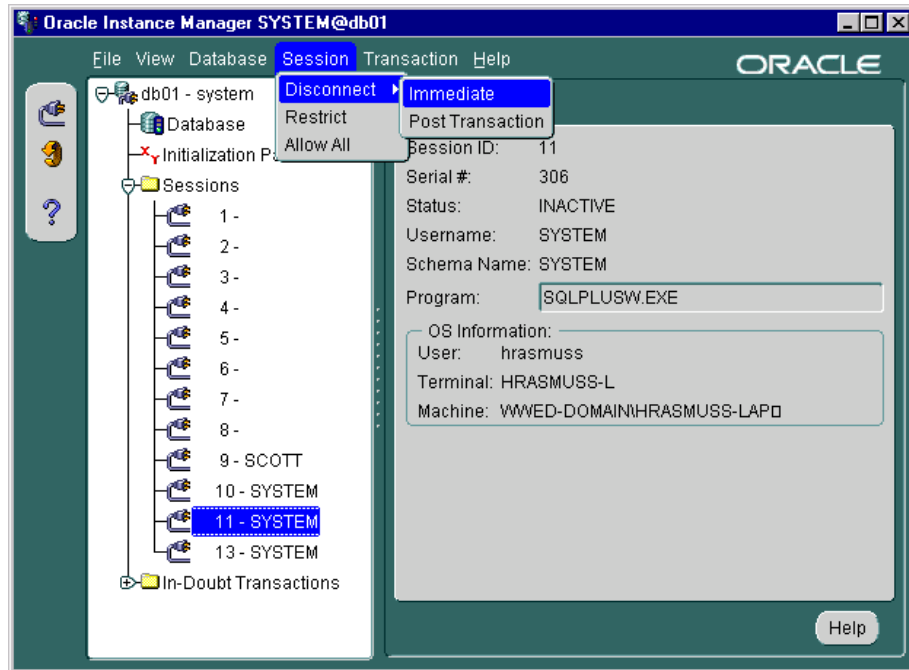
- d Respond to the Storage Manager dialog by clicking Yes.

**Hint:** Now, identify the sessions that are using the rollback segments in the RBS tablespace.

The solution is the same as in Appendix C.

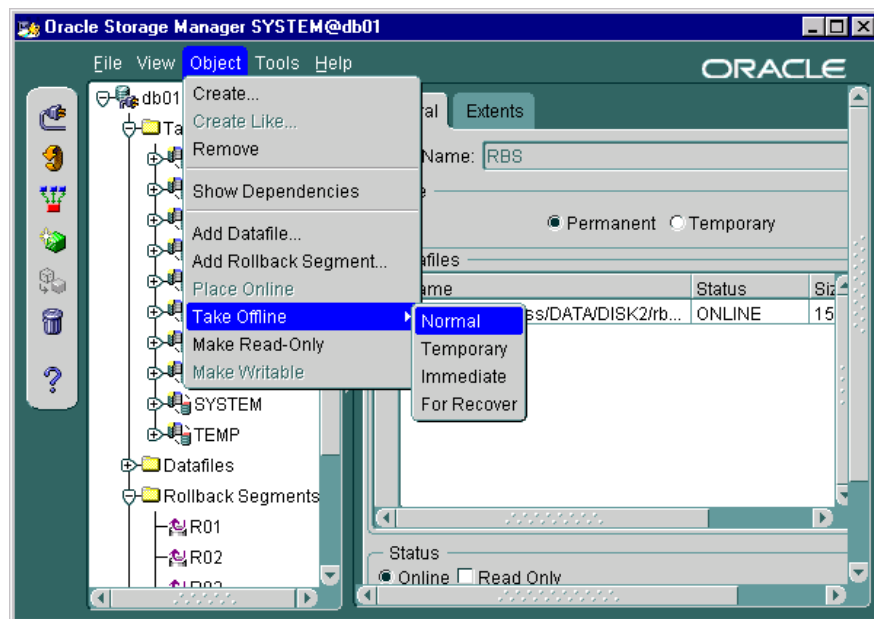
**Hint:** Kill the sessions using rollback segments in the tablespace.

- a Use Instance Manager.
- b Expand the Sessions folder. Select the session to be killed.
- c Select Session—>Disconnect—>Immediate.



**Hint:** Take the RBS tablespace offline.

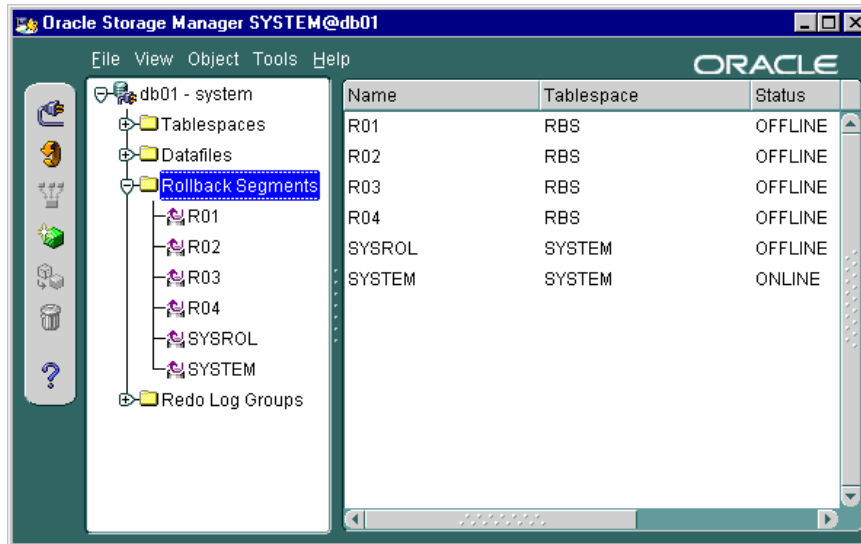
- a Use Storage Manager.
- b Expand the Tablespaces folder. Select RBS.
- c Select Tablespace—>Take Offline—>Normal from the right mouse button menu.



- d Click Yes in the Oracle Storage Manager dialog box.

- 7** (a) Shut down the instance, start up again, and query the rollback segment information in the data dictionary to check how many rollback segments are online.

- a** Use Instance Manager to shutdown and restart the instance.  
**b** Use Storage Manager to check the status of all rollback segments

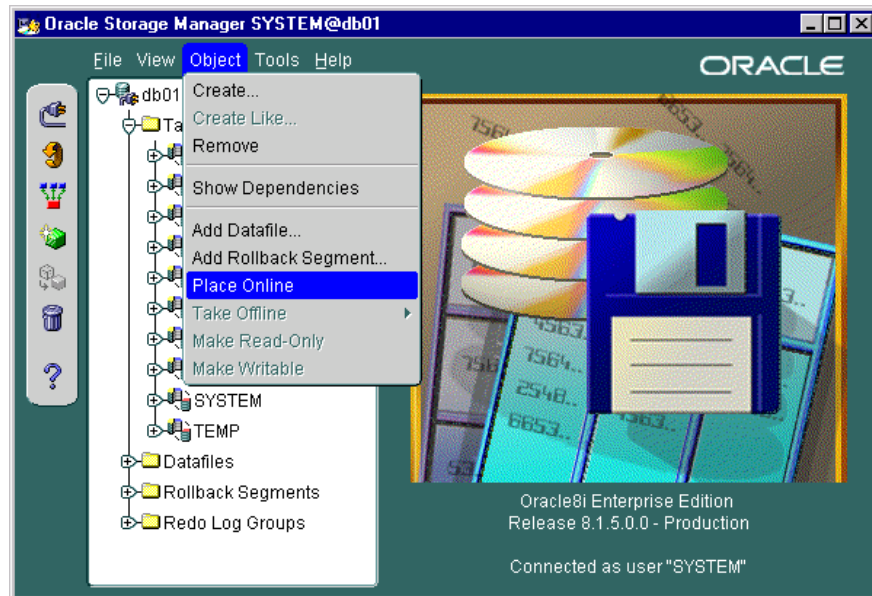


- (b) Ensure that all rollback segments will be brought online whenever the database is opened in the future and restart the instance. Make sure that all the rollback segments have the status ONLINE.



**Hint:** This can be done using the `ROLLBACK_SEGMENTS` initialization parameter. Also, the RBS tablespace must be brought online.

- a** Use Storage Manager to place the RBS tablespace online.



- b** Edit the *initdb01.ora* and specify:  
`rollback_segments = (r01,r02,r03,r04)`
- c** Use Instance Manager to shut down and reopen the database.
- d** Verify that the rollback segments are online, using Storage Manager.

- 8** Create a new rollback segment called DEMO\_RBS in the database with the following storage parameters, and ensure that it can be used. This is essential for the next question:

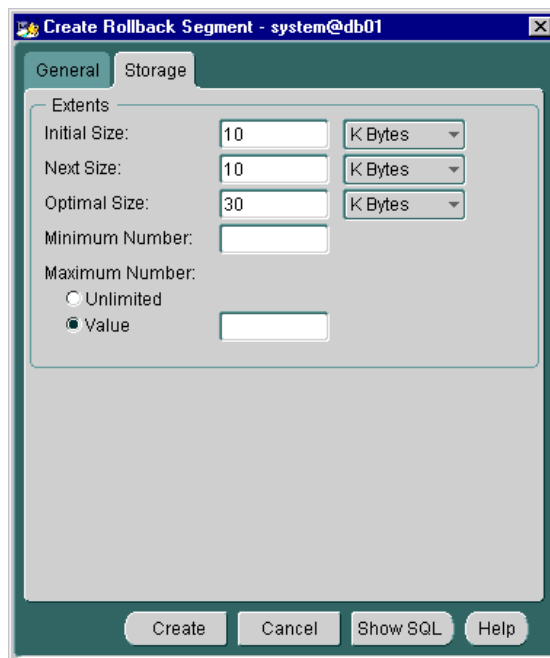
INITIAL = 10K

NEXT = 10K

OPTIMAL = 30K

**Hint:** To ensure that the rollback segment can be used, it must be brought online after creation.

- a** Use Storage Manager.
- b** Select the Rollback Segments folder. Select Create from the right mouse button menu.
- c** Specify DEMO\_RBS as the name of the rollback segment, choose RBS as the tablespace, and select Online in the General page.
- d** Specify the storage settings in the Extents page.



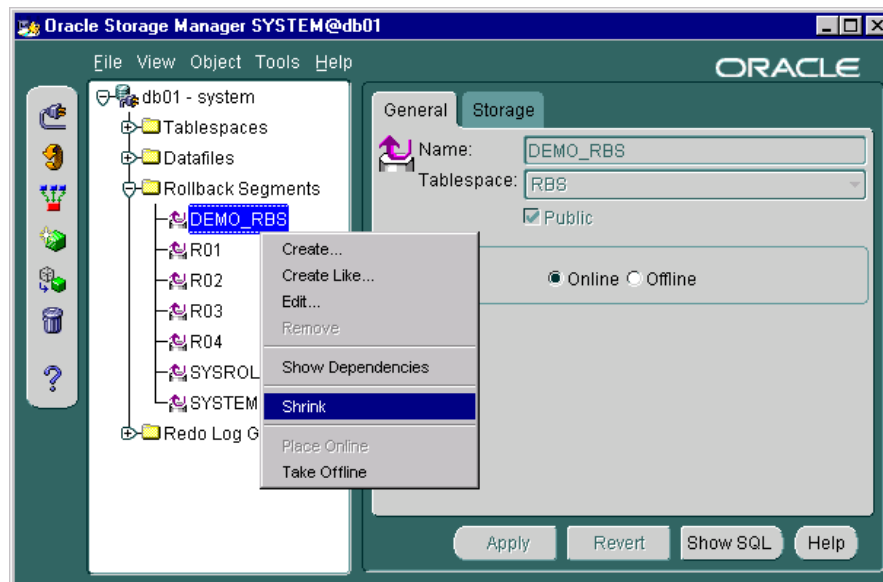
- e** Click Create.

- 9** The solution is the same as in Appendix C.

**10** Ensure that the rollback segment DEMO\_RBS is reduced to its optimal size and verify that it has reduced.

- a** Use SQL\*Plus Worksheet to run the following query:  

```
SQL> SELECT extents, rssize, optsize
2 FROM v$rollstat
3 WHERE usn = (SELECT usn
4 FROM v$rollname
5 WHERE name='DEMO_RBS');
```
- a** Use Storage Manager.
- b** Expand the Rollback Segments folders.
- c** Select DEMO\_RBS. Select Shrink from the right mouse button menu.



- d** Specify Optimal Size in the Shrink Rollback dialog and click OK.
- e** Use SQL\*Plus Worksheet to run the following query:  

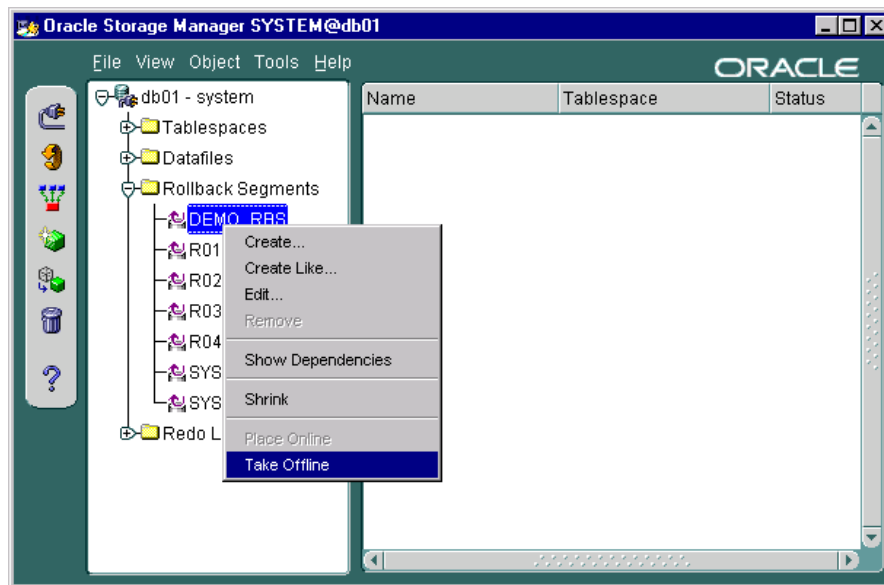
```
SQL> SELECT extents, rssize, optsize
2 FROM v$rollstat
3 WHERE usn = (SELECT usn
4 FROM v$rollname
5 WHERE name='DEMO_RBS');
```

**11** The solution is the same as in Appendix C.

**12** Drop the DEMO\_RBS rollback segment.

**Hint:** The rollback must be taken offline before it is dropped. You may need to ensure that no transactions are using the rollback segment.

- a** Roll back all active transactions.
- b** Use Storage Manager.
- c** Expand the Rollback Segments folder.
- d** Select DEMO\_RBS. Select Take Offline.



- e** Click Yes in the Storage Manager dialog box.
- f** Select Object—>Remove.
- g** Click Yes in the Storage Manager dialog box.

## Practice 11 Solutions

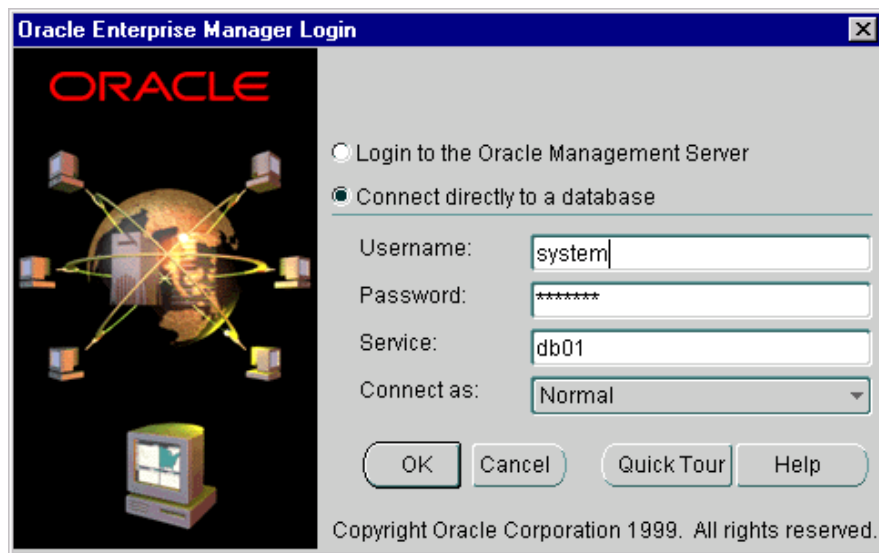
### Using SQL\*Plus Worksheet

The commands for using SQL\*Plus and SQL\*Plus Worksheet are identical. Please refer to Appendix C for the answers unless a specific Oracle Enterprise Manager solution is used.

You can start the SQL\*Plus Worksheet from the Windows NT menu by doing the following:

#### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



#### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 You need to create the following tables for an order entry system that you are implementing now. The tables and the columns are shown below:

Table	Column	Data Type and Size
CUSTOMERS	CUST_CODE	VARCHAR2(3)
	NAME	VARCHAR2(50)
	REGION	VARCHAR2(5)
ORDERS	ORD_ID	NUMBER(3)
	ORD_DATE	DATE
	CUST_CODE	VARCHAR2(3)
	DATE_OF_DELY	DATE

You have been informed that in the table ORDERS, rows will be inserted without a value for DATE\_OF\_DELY, and it will be updated when the order is fulfilled. Log in as *system/manager* and create the tables using the appropriate block space utilization and tablespace settings. Use tablespace DATA01. You can use the default storage settings. Use the Table Wizard when creating table CUSTOMERS. Do not use the Table Wizard when creating table ORDERS.

**Hint:** PCTFREE has to be set carefully for the ORDERS table because rows in this table are likely to grow after updates.

- a** Use Schema Manager
- b** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Schema Manager.
- c** Connect directly to a database in the login dialog box.
- d** Select Object —>Create from the menu.
- e** Select Table from the list. Click Create.
- f** Specify tablename, schema, and tablespace. Click Next.
- g** Specify column names, data types, and sizes. Click Insert to add a new column. Click Finish when done.

**Table Wizard Object Creation Wizard**

**Columns Definition**

Columns defined :

- CUST\_CODE
- NAME
- REGION

Properties of column : REGION

Column Name: REGION

Column Datatype: VARCHAR2

Size: 5

Scale:

Does this column have a default value? If so, please enter.

Insert Remove

Cancel Help Back Next Finish

- h** Click OK.
- i** Create the table ORDERS.
- j** Select Object—>Create from the menu.
- k** Select Table from the list. Clear the Use Wizard box. Click Create.
- l** Specify tablename, schema, tablespace, column names, data types, and sizes.
- m** Select the Storage tab. Specify 35 in the field% Free. Click Create. Click OK.

- 2** The solution is the same as in Appendix C.
- 3** The solution is the same as in Appendix C.
- 4** The solution is the same as in Appendix C.
- 5** The solution is the same as in Appendix C.
- 6** (a) Drop the table BIG\_EMP.

- a** Use Schema Manager
- b** Expand the Tables folder.
- c** Expand SYSTEM.
- d** Select table BIG\_EMP.
- e** Select Remove from the right mouse menu. Click Yes to remove the table.

- (b) The solution is the same as in Appendix C. Do not use Schema Manager, because it will not copy the data to the new table.
- 7** The solution is the same as in Appendix C.
- 8** The solution is the same as in Appendix C.
- 9** The solution is the same as in Appendix C.
- 10** The solution is the same as in Appendix C.
- 11** (a) For the order entry application, you now need to add a PRODUCTS table, which has the following columns:

Column	Data Type and Size
PROD_CODE	NUMBER(6)
DESCRIPTION	VARCHAR2(30)
PRICE	NUMBER(8,2)



Create this table in tablespace DATA02 using uniform extent sizes of 10 KB.

- a** Use Schema Manager.
- b** Expand the Tables folder.
- c** Select SYSTEM.
- d** Select Create from the right mouse button menu.
- e** Specify table name, tablespace name, column names, data types, and sizes in the General page.
- f** Specify extent sizes of 4 KB and PCTINCREASE=0 in the Storage page.

The screenshot shows the 'Create Table - system@db01' dialog box with the 'Storage' tab selected. The 'Explicit' radio button is chosen under 'Extents'. The 'Initial Size' is set to 10 K Bytes, 'Next Size' is 10 K Bytes, and 'Increase Size by' is 0 %. The 'Maximum Number' is set to 'Value' with an empty input field. The 'Space Usage' section has empty fields for '% Free' and '% Used'. The 'Number of Transactions' section has empty fields for 'Initial' and 'Maximum'. The 'Free Lists' section has empty fields for 'Free Lists' and 'Groups'. At the bottom are buttons for 'Create', 'Cancel', 'Show SQL', and 'Help'.

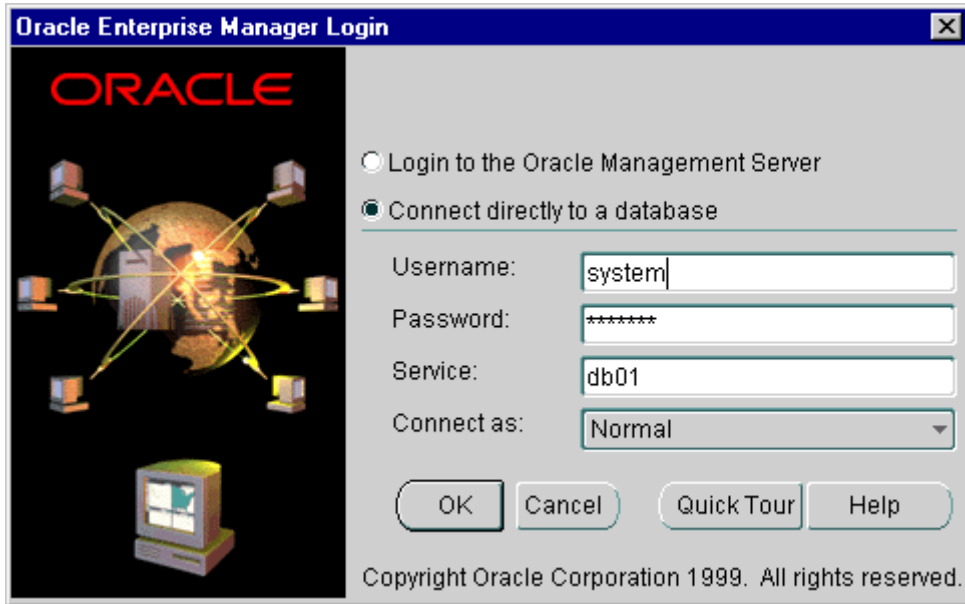
- g** Click Create.
- h** Click OK.

(b) The solution is the same as in Appendix C.

## Practice 12 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 You are considering creating indexes on the `NAME` and `REGION` columns of the `CUSTOMERS` table. What types of index are appropriate for the two columns? Create the indexes, naming them `CUST_NAME_IDX` and `CUST_REGION_IDX`, respectively, and placing them in the appropriate tablespaces.

**Hint:** A B-tree index is suitable for a column with many distinct values, and a bitmap index is suitable for columns with only a few distinct values.

- a Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack —>Schema Manager.
- b Connect directly to a database in the login dialog box.
- c Expand the Tables folder.
- d Expand SYSTEM. Select the CUSTOMERS table.
- e Select Create Index On from the right mouse button menu.
- f Specify index name, tablespace name, and column name in the General page.

**Create Index - system@db01**

General    Partitions    Storage    Options

Name: CUST\_NAME\_IDX  
 Schema: SYSTEM  
 Tablespace: INDX01

Index On: ☒ Table ☐ Cluster

Schema: SYSTEM  
 Table: CUSTOMERS

Table Columns	Order
CUST_CODE	
NAME	1
REGION	

Options  
☐ Unique ☐ Bitmap ☐ Unsorted ☐ Reverse ☐ No Logging

Create Cancel Show SQL Help

- g Specify a PCTFREE of 35 in the Storage page.
- h Click Create. Click OK.
- i Repeat steps f through j, specifying Bitmap on the General page for the bitmap index.

- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C
- 4 The solution is the same as in Appendix C.
- 5 (a) The solution is the same as in Appendix C.
- 6 (b) The solution is the same as in Appendix C.
- 7 (c) The solution is the same as in Appendix C.

Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

**Hint:** Use PCTINCREASE=0 to create extents of equal size.

Refer to Appendix C to query the segments.

- a Use Schema Manager.
- b Expand the Tables folder.
- c Expand SYSTEM.
- d Select NUMBERS.
- e Select Create Index On from the right mouse button menu.
- f Specify index name, tablespace name, and column name in the General page.
- g Specify extent sizes and PCTINCREASE of 0 in the Storage page.
- h Click Create.
- i Repeat steps 5 through 8 for the second index.

(d) Once again, using uniform extent sizes of 4 KB, create bitmap indexes NUMB\_OE\_IDX and NUMB\_NO\_IDX on the ODD\_EVEN and NO columns of the NUMBERS table, respectively, and check the total sizes of the indexes.

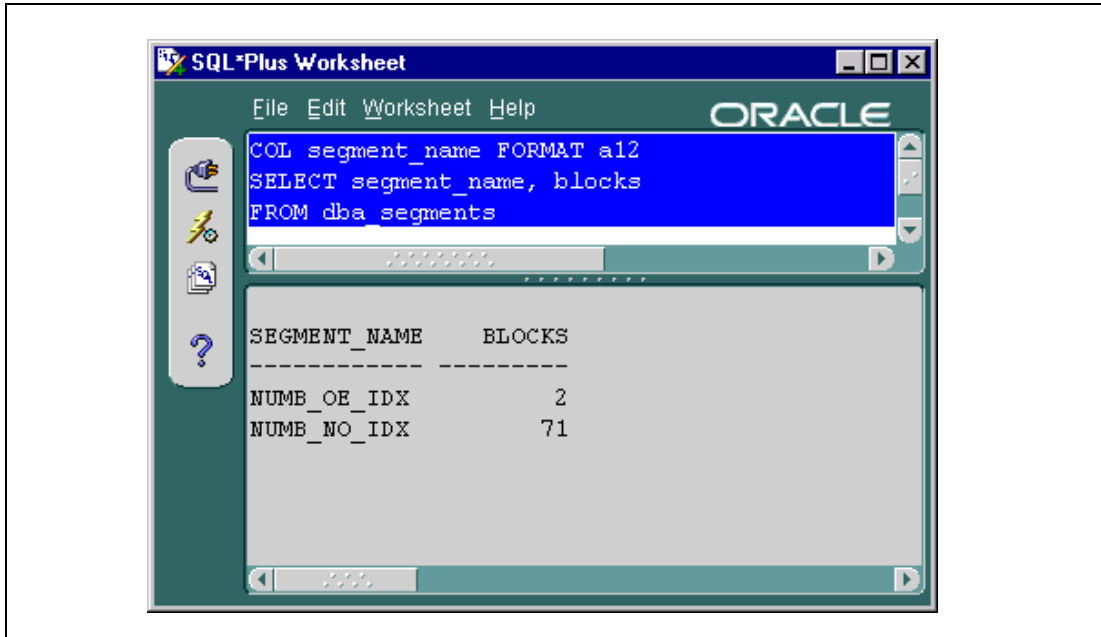
Index	Blocks
NUMB_OE_IDX	
NUMB_NO_IDX	

What can you conclude about the relationship between cardinality and sizes of the two types of indexes?

**Hint:** The existing indexes need to be dropped before creating the new indexes.

- a** Use Schema Manager.
- b** Expand the Indexes folder.
- c** Expand SYSTEM.
- d** Select the index to be dropped.
- e** Select Remove from the right mouse button menu. Click Yes in the Schema Manager Dialog.
- f** Expand the Tables folder.
- g** Expand SYSTEM.
- h** Select NUMBERS.
- i** Select Create Index On from the right mouse button menu.
- j** Specify index name, tablespace name, bitmap index type, and column name in the General page.
- k** Specify extent sizes and PCTINCREASE of 0 in the Storage page.
- l** Click Create.
- m** Repeat steps 2 through 13 for the second index.

**Hint:** Now reexecute the query to check the sizes of the indexes.

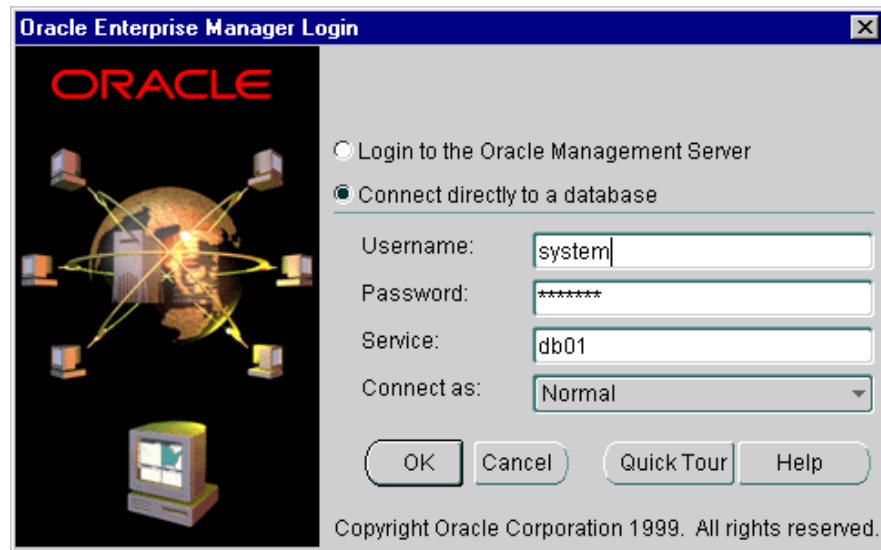


**It can be seen from the results that a bitmap index is compact for a low-cardinality column, while a B-tree index is compact for a high-cardinality column.**

## Practice 13 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



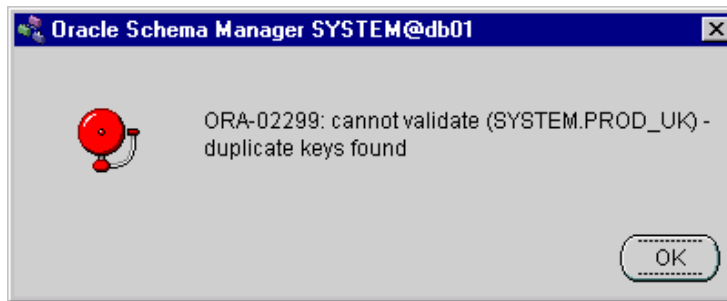
### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.
- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C.

- 4 Enable the unique constraint on the PRODUCTS table. Was it successful? Why or why not?

- a Use Schema Manager.
- b Expand the Tables folder.
- c Expand SYSTEM.
- d Select PRODUCTS.
- e Cross Disable against PROD\_UK in the Constraints page.
- f Click Apply.



**The constraint cannot be enabled because there are rows that violate the constraint.**

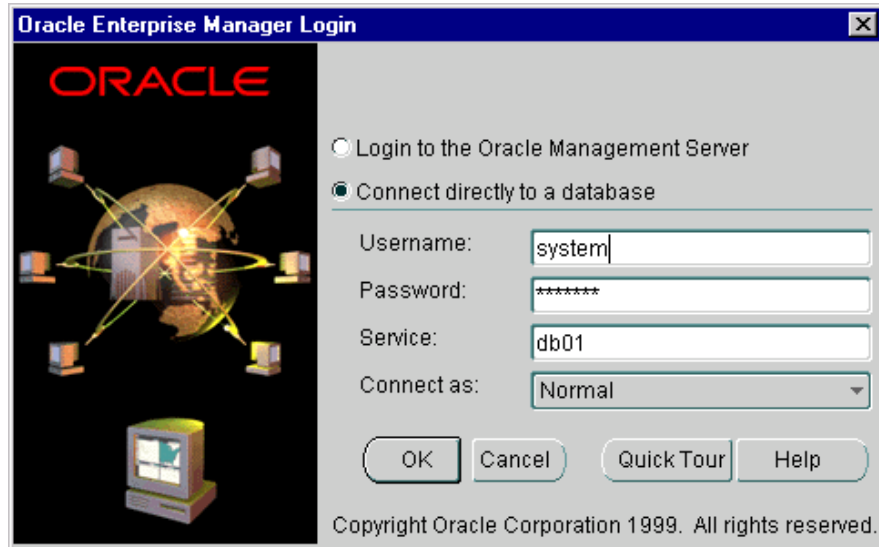
- 5 The solution is the same as in Appendix C.
- 6 The solution is the same as in Appendix C.
- 7 The solution is the same as in Appendix C.
- 8 The solution is the same as in Appendix C.
- 9 The solution is the same as in Appendix C.



## Practice 14 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



### Step Two

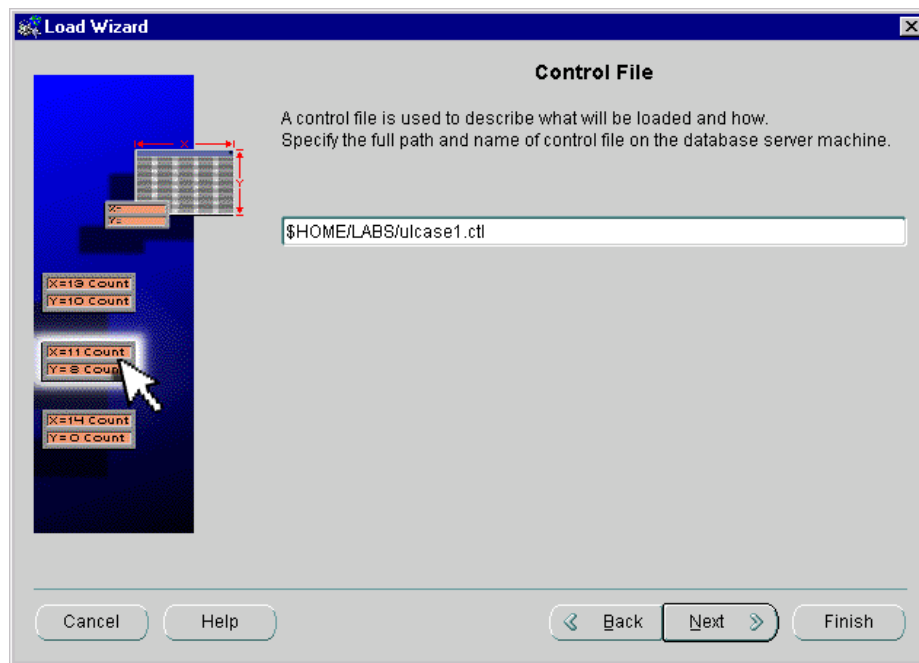
Make sure you connect directly to a database. Enter the username *system*, the password *manager*, and the service name for your working database, and click OK.

Use the account *system* for all the questions in this lab.

- 1 The solution is the same as in Appendix C.
- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C.
- 4 Examine the scripts *ulcase1.sql*, *ulcase1.ctl*, *ulcase2.ctl*, and *ulcase2.dat*. These are some of the standard loader demonstration files supplied with the Oracle8i Enterprise Edition. As user *system*, perform the following steps to try two load runs and get acquainted with using SQL\*Loader:
  - (a) The solution is the same as in Appendix C.

(b) Run SQL\*Loader to load data into the DEPT table using the control file *ulcase1.ctl*. Examine the log file generated, and query the DEPT table to check the data loaded

- a** Select Programs —> Oracle - *EMV2 Home* —> Oracle Enterprise Management —> Enterprise Manager Console.
- b** Expand your working database.
- c** Expand the Schema Objects folder.
- d** Expand the Tables folder.
- e** Expand SYSTEM.
- f** Select DEPT and choose Data Management —> Load from the right mouse button menu.
- g** Specify control file name as *ulcase1.ctl* in the Introductions page.



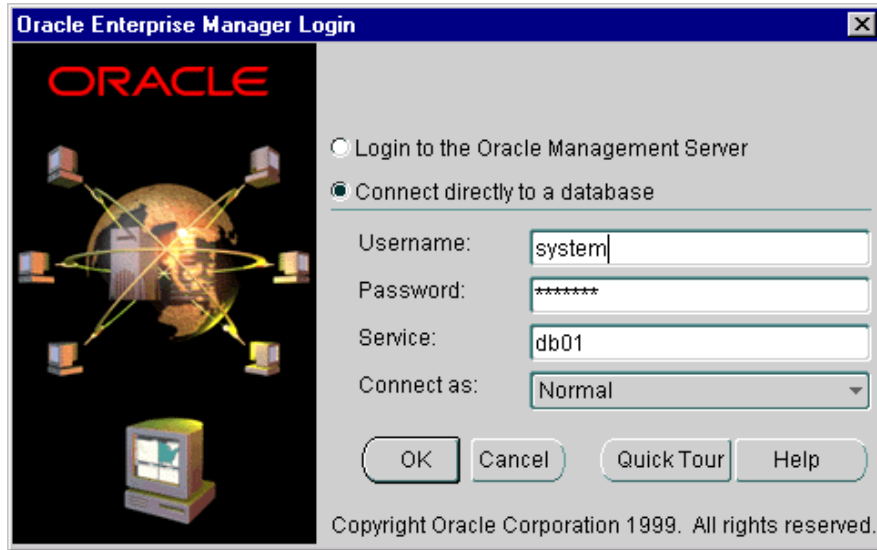
- h** Click Finish in this page and click OK in the Summary page.
- i** Use SQL\*Plus Worksheet to query the DEPT table. See Appendix C for the query.

- C** Run SQL\*Loader again to load data into the EMP table using the control file *ulcase2.ctl*. Notice that this run uses an input data file to load data. Examine the log file generated, and query the EMP table to check the data loaded. **Follow the steps shown in (b) but use EMP instead of DEPT and use *ulcase2.ctl* instead of *ulcase1.ctl*.**
- 5** The solution is the same as in Appendix C.

## Practice 15 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



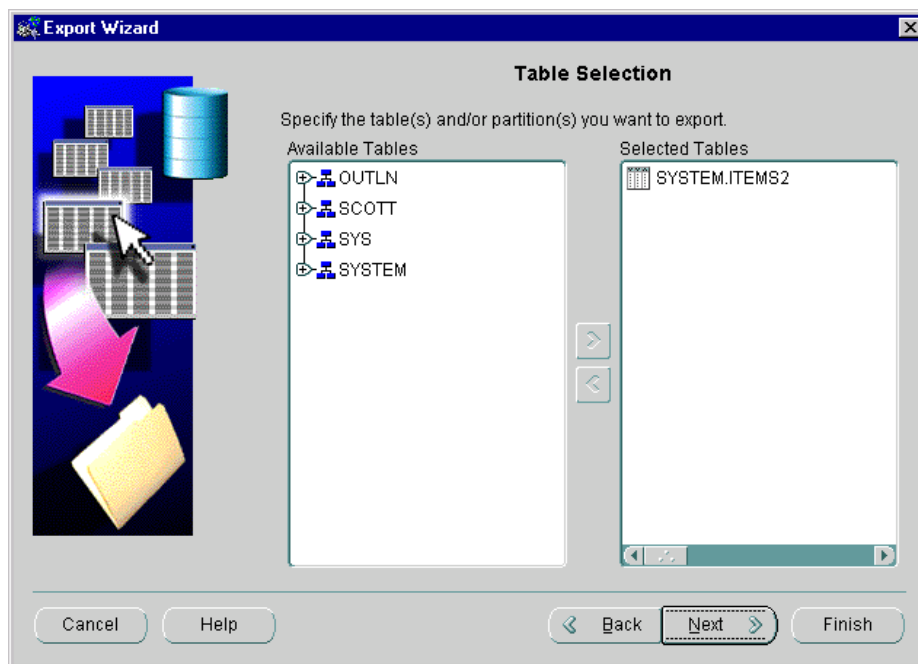
### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 You want to reorganize the `ITEMS2` table using export and import. Check the number and size of the extents in the `ITEMS2` table after import. What can you infer about the behavior of export and import on space allocation?

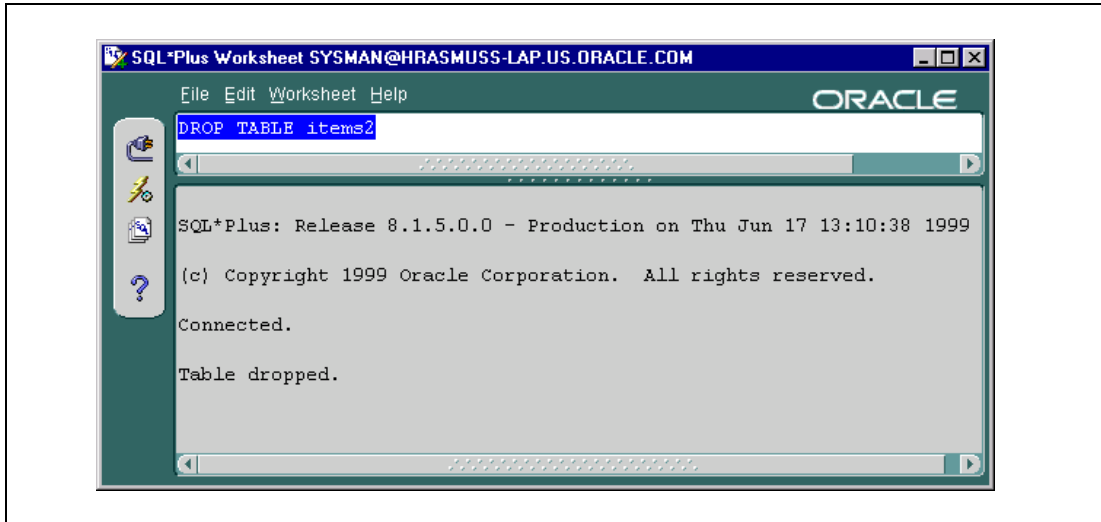
**Hint:** Use table level export to do this.

- a Select Programs —>Oracle - *EMV2 Home*—>Oracle Enterprise Management—>Enterprise Manager Console.
- b Expand your working database.
- c Expand the Schema Objects folder.
- d Expand the Tables folder.
- e Expand SYSTEM.
- f Select ITEMS2 and choose Data Management—>export from the right mouse button menu.
- g Accept the default Export File name of EXPDAT.DMP in the Export Filepage, and click Next.
- h Select Table in the Export Type page, and click Next.
- i Verify that only the ITEMS2 table is shown in the Selected Tables pane of the Table Selection page.



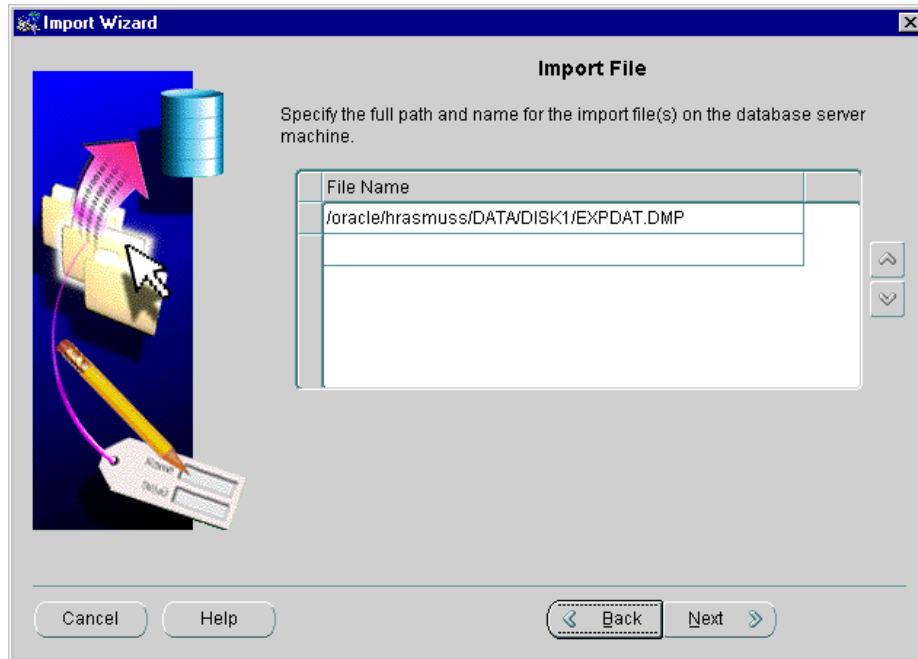
- j Click Finish in the Table Selection page and click OK in the Summary page.

**Hint:** Drop the ITEMS2 table.

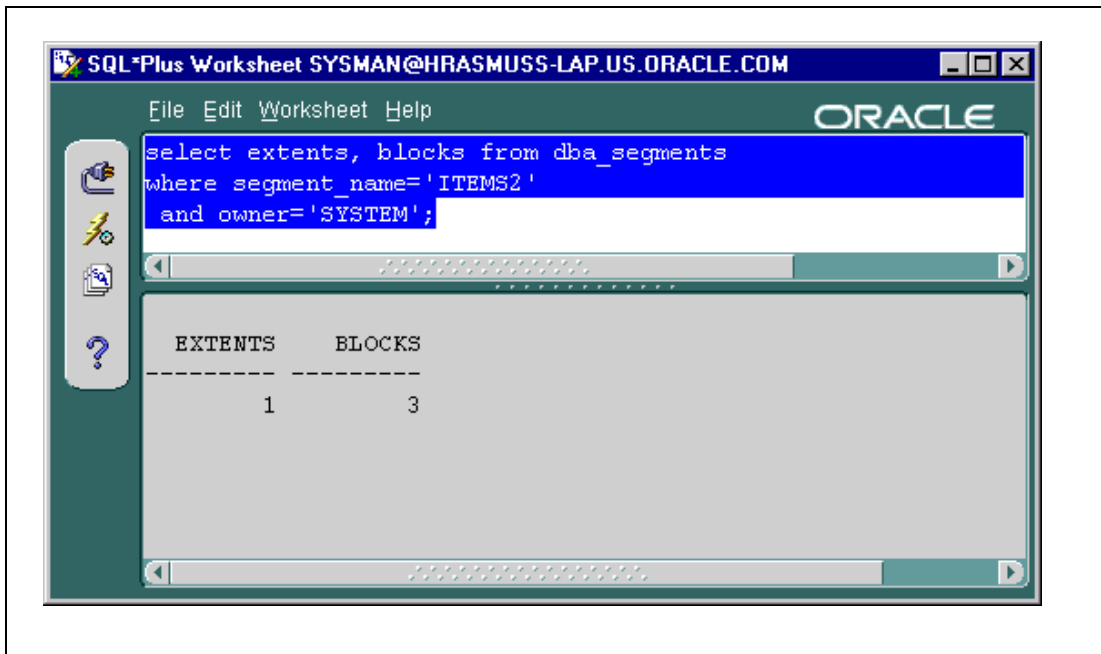


**Hint:** Import the ITEMS2 table.

- a** Select Programs —> Oracle - *EMV2 Home* —> Oracle Enterprise Management —> Enterprise Manager Console.
- b** Select your working database and choose Data Management —> Import from the right mouse button menu.
- c** Accept the default Export File name of EXPDAT.DMP in the Import Filepage, click Next, and wait for the job to finish



**Hint:** Check the number and size of the extents in the ITEMS2 table using DBA\_SEGMENTS.



**By default, exporting a table and importing it will result in the table having one extent, which is as large as the original size of the segment.**

**2** You need to move several indexes from one tablespace to another. This practice uses one index to show how this can be done.

(a) Create an index named ITEM\_OID\_IDX, in the tablespace DATA01, on the ORD\_ID column of the ITEMS2 table.

- a** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Schema Manager.
- b** Expand Tables.
- c** Expand SYSTEM.
- d** Select ITEMS2.
- e** Select Object—>Create Index On.
- f** Specify index name, tablespace name, and column name in the General page.
- g** Click Create.

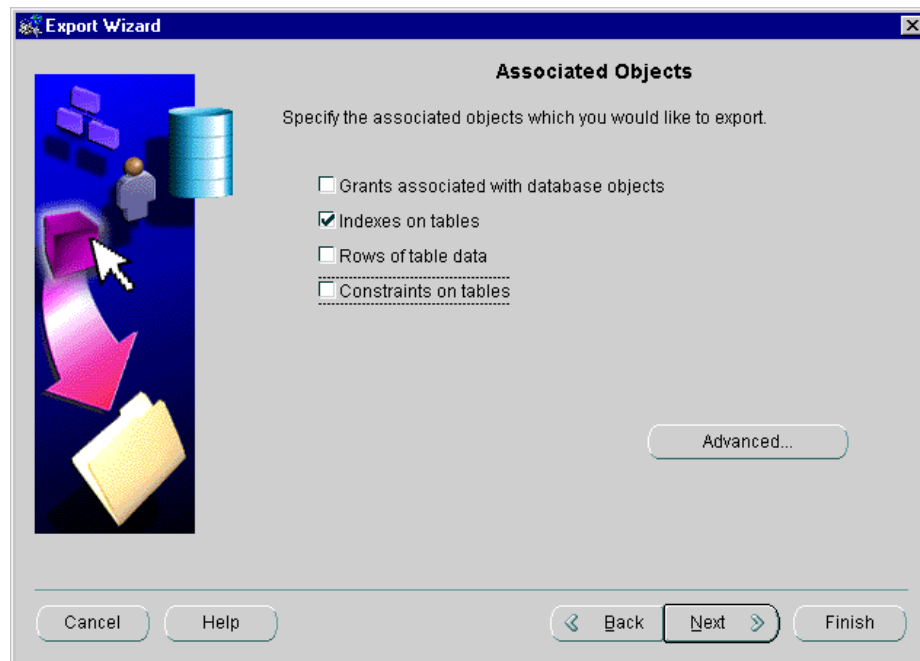
(b) Using export and import, move the index to the tablespace INDX01.

**Hint:** Use the following steps.



Export the table with indexes, but without rows.

- a** Select Programs —>Oracle - *EMV2 Home*—>Oracle Enterprise Management—>Enterprise Manager Console.
- b** Expand your working database.
- c** Expand the Schema Objects folder.
- d** Expand the Tables folder.
- e** Expand SYSTEM.
- f** Select ITEMS2 and choose Data Management—>Export from the right mouse button menu.
- g** Accept the default Export File name of EXPDAT.DMP in the Export Filepage, and click Next
- h** Select Table in the Export Type page, and click Next.
- i** Verify that only the ITEMS2 table is shown in the Selected Tables pane of the Table Selection page. Click Next.
- j** Select only Indexes on Tables in the Associated Objects page.



- k** Click Finish in the Associated Objects page and click OK in the Summary page.

**Hint:** Drop the index.

- a** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Schema Manager.
- b** Expand Tables.
- c** Expand SYSTEM.
- d** Select ITEMS2.
- e** Expand Indexes.
- f** Select ITEM\_OID\_IDX.
- g** Select Object—>Remove.
- h** Click Yes in the Schema Manager dialog box.

**Hint:** Import from the output of the previous export and create a script to build the index.

- a** Select Programs —>Oracle - *EMV2 Home*—>Oracle Enterprise Management—>Enterprise Manager Console.
- b** Select your working database and choose Data Management—>Import from the right mouse button menu.
- c** Accept the default Export File name of EXPDAT.DMP in the Import Filepage, and click Next.
- d** Click <– in the Selections page to specify that all items in the export file must be imported.
- e** Click Next in this page and the next two pages.
- f** In the Advanced Options page, specify Increment Type as none, click Write Index-Creation command to file, and specify a Index File name.
- g** Click Finish in this page and click OK the Summary page.

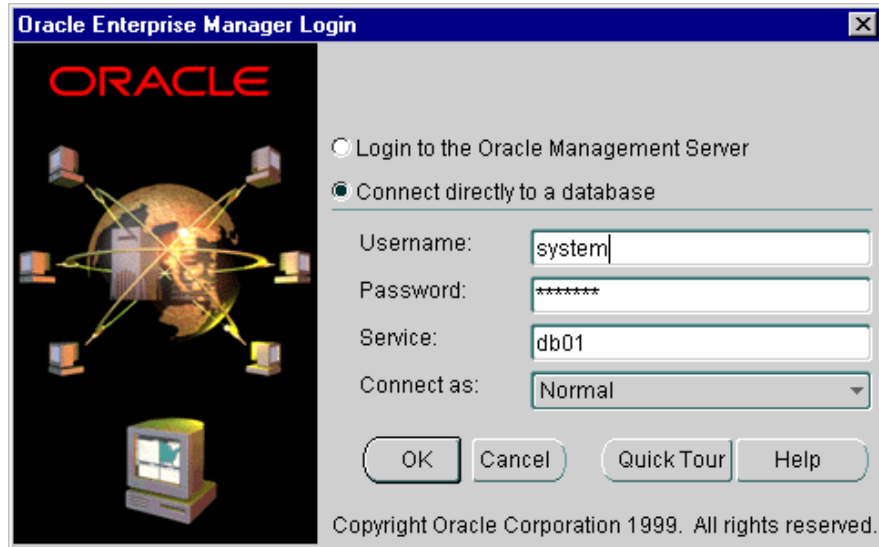
**Hint:** Edit the index file created in the previous step to change the tablespace to INDX0, and run the script to re-create the index.

The solution is the same as in Appendix C.

## Practice 16 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>SQL Plus Worksheet.



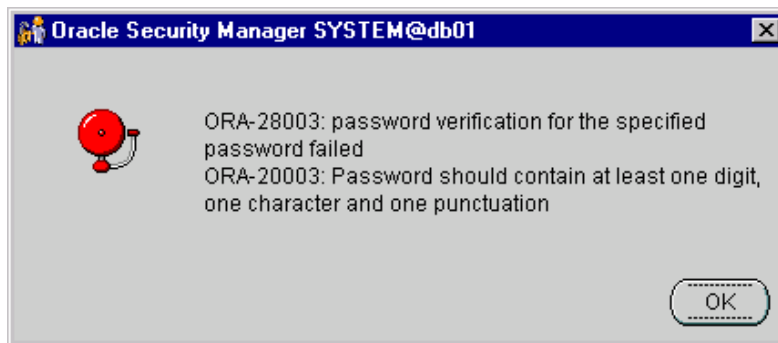
### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.

**2** Try to change the password of user SCOTT to SCOTT. What happens?

- a** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>Security Manager.
- b** Expand the Users folder.
- c** Select SCOTT.
- d** Specify the new password in General page.
- e** Click Apply



**3** Make sure that the following applies to users assigned the DEFAULT profile:

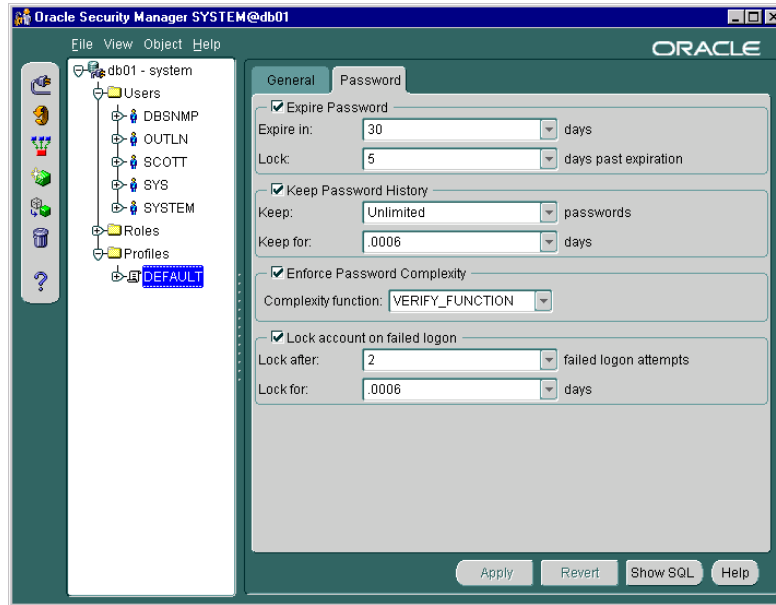
- After two login attempts, the account should be locked.
- The password should expire after 30 days.
- The same password should not be reused for at least one minute.
- The account should have a grace period of five days to change an expired password.

Ensure that the requirements given have been implemented

**Hint**

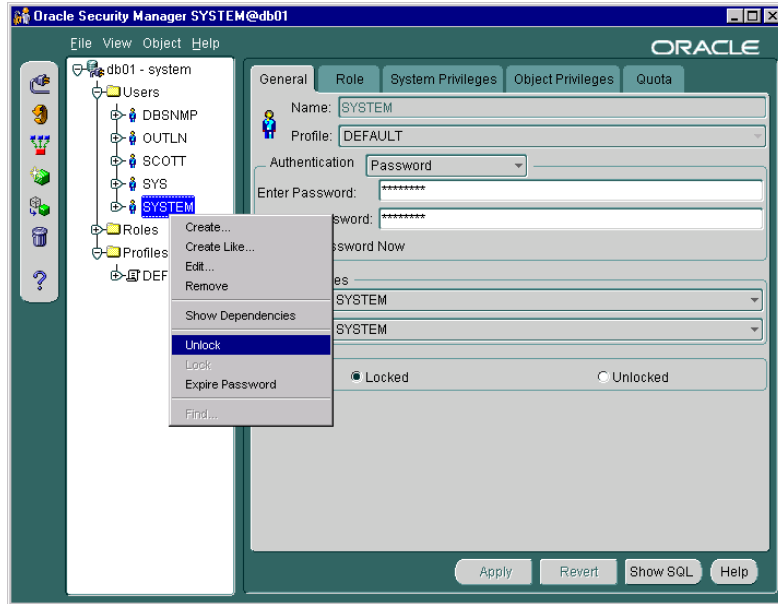
- Use the ALTER PROFILE command to change the default profile limits.
- Query the data dictionary view DBA\_PROFILES to verify the result.

- a Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Security Manager.
- b Expand the Profiles folder.
- c Select the DEFAULT profile.
- d Select the Password Tab.
- e Enter the new limits and click Apply



- 4 The solution is the same as in Appendix C.
- 5 Ensure that the SYSTEM can reconnect.  
**Hint:** Execute the ALTER USER command to unlock the SYSTEM account.

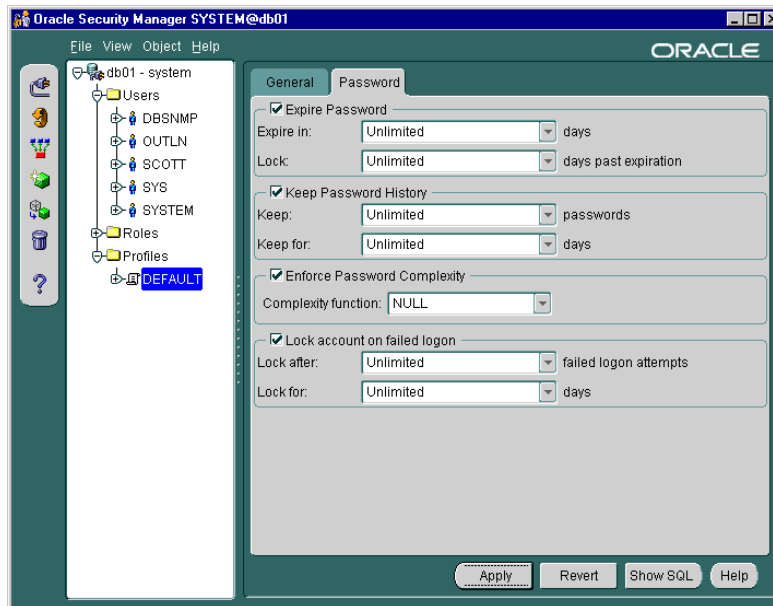
- a** Use Security Manager.
- b** Expand Users.
- c** Select SYSTEM.
- d** Select Unlock from the right mouse button menu.



- 6** Disable password checks for the DEFAULT profile.

**Hint:** Execute the ALTER PROFILE command to disable the password checks.

- a Use Security Manager.
- b Expand Profiles.
- c Select DEFAULT.
- d Specify NULL for Complexity function and UNLIMITED for all other values in the Password page.

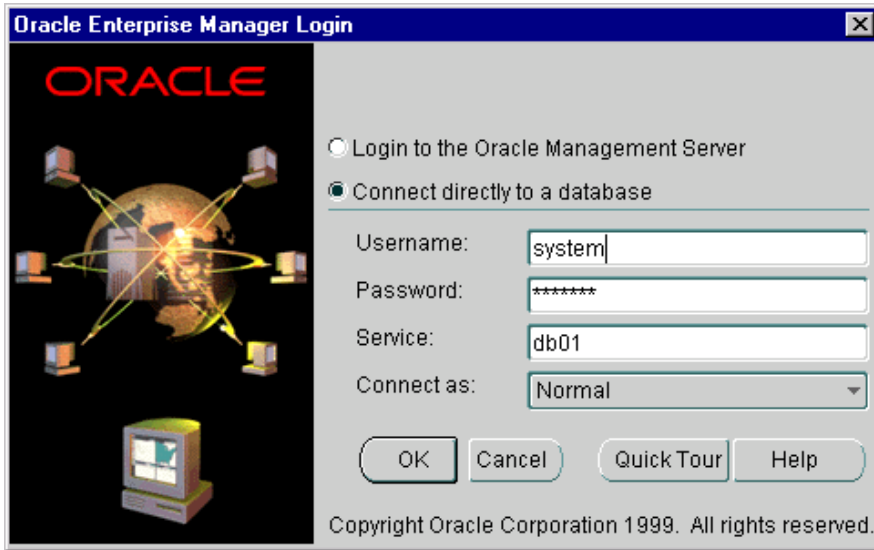


- e Click Apply.

## Practice 17 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet.



### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 Create user Bob with a password of ALONG. Make sure that any objects and temporary segments created by Bob are not created in the system tablespace. Also, ensure that Bob can log in and create objects up to one megabyte in size in the DATA01 and INDX01 tablespaces. If you are not using Oracle Enterprise Manager, run the script `bob.sql`.



**Hint:** Ensure that the temporary tablespace is assigned.

- a** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Security Manager.
- b** Select the Users folder and choose Create from the right mouse button menu.
- c** Specify username, password, default tablespace, and temporary tablespace in the General page.

The screenshot shows the 'Create User' dialog box with the following details:

- Title:** Create User - system@db01
- Tabs:** General, Role, System Privileges, Object Privileges, Quota
- Name:** BOB
- Profile:** DEFAULT
- Authentication:** Password
- Enter Password:** \*\*\*\*\*
- Confirm Password:** \*\*\*\*\*
- Expire Password Now:** ☐
- Tablespaces:**
  - Default:** DATA01
  - Temporary:** TEMP
- Status:** ☐ Locked, ☒ Unlocked
- Buttons:** Create, Cancel, Show SQL, Help

- d** Specify quotas on tablespaces in the Quotas page by selecting the tablespace in the scrolling list, clicking on the Value button, and defining the value.
- e** Click Create.

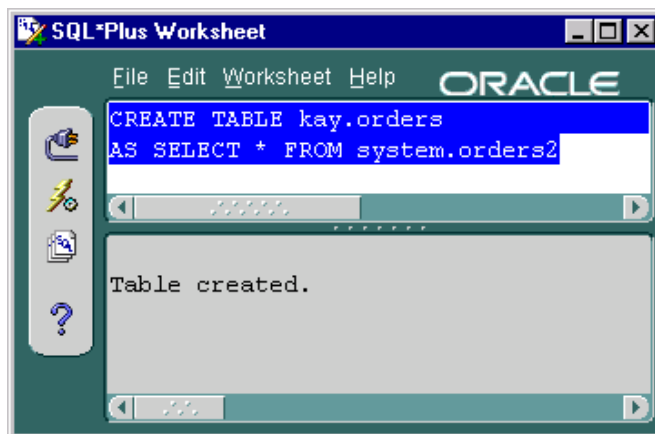
- 2** (a) Create a user Kay with a password of MARY. Make sure that any objects and sort segments created by Kay are not created in the system tablespace.

- a** Use Security Manager.
- b** Select User—>Create.
- c** Specify username, password, default tablespace, and temporary tablespace in the General page.
- d** Click Create.

(b) As user SYSTEM, copy the ORDERS2 table from the SYSTEM schema to Kay's account. Call the new table ORDERS.

**Hint:** Kay needs a quota on her default tablespace before objects can be created in her schema.

- a** Use Security Manager.
- b** Expand Users.
- c** Select KAY.
- d** Specify an UNLIMITED quota on DATA01 in Quotas page.
- e** Click Apply.



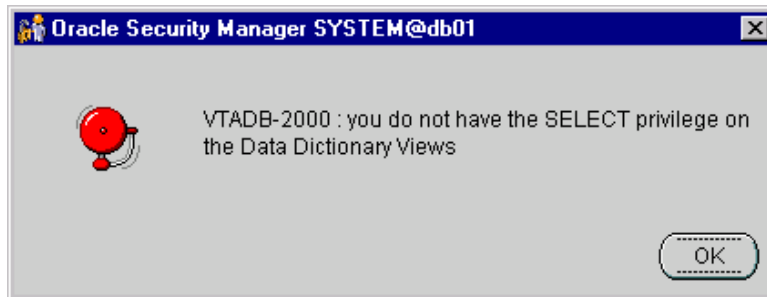
- 3** The solution is the same as in Appendix C.
- 4** The solution is the same as in Appendix C.

5 (a) As user Bob, change his temporary tablespace. What happens? Why?

- a Use Security Manager.
- b Select your working database and choose Change Database Connection from the right mouse button menu.



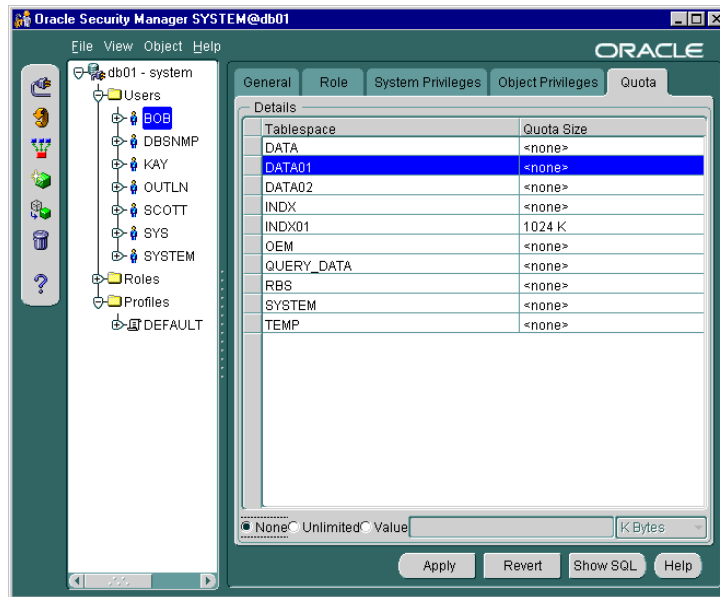
- c Enter the username bob, password along, and service name db01.



(b) The solution is the same as in Appendix C.

**6** As *system*, remove Bob's quota on his default tablespace.

- a** Use Security Manager.
- b** Expand the Users folder.
- c** Select BOB.
- d** Specify a 0 quota on DATA01 in Quotas page.

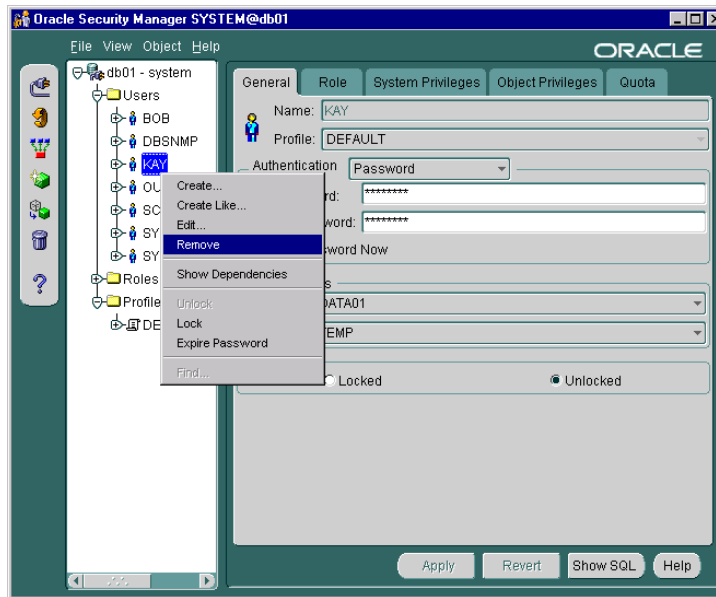


- e** Click Apply.

**7** Remove Kay's account from the database.

**Hint:** Because Kay owns tables, you need to use the CASCADE option.

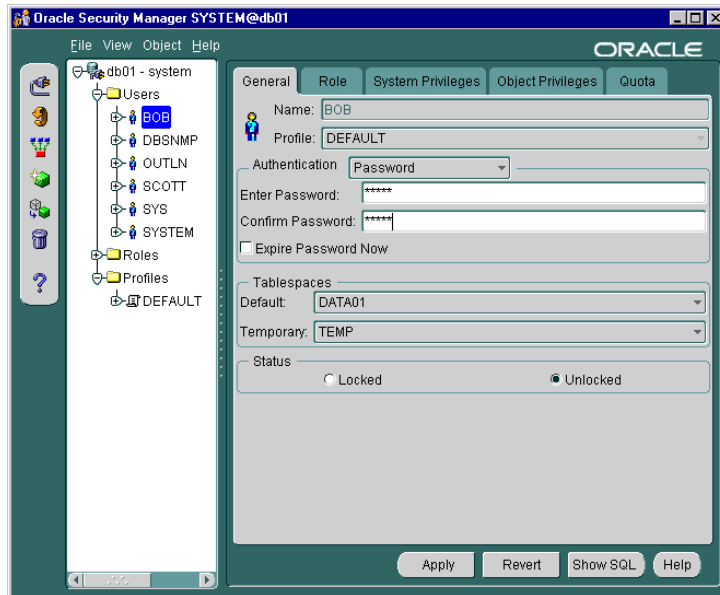
- a Use Security Manager.
- b Expand the Users folder.
- c Select KAY.
- d Select Remove from the right mouse button menu.



- e Click Yes in the Security Manager dialog box.

**8** Bob has forgotten his password. Assign him a password of OLINK and require that Bob change his password the next time he logs on.

- a** Use Security Manager.
- b** Expand the Users folder.
- c** Select BOB.
- d** Select Expire Password Now and enter the new password in the General page.

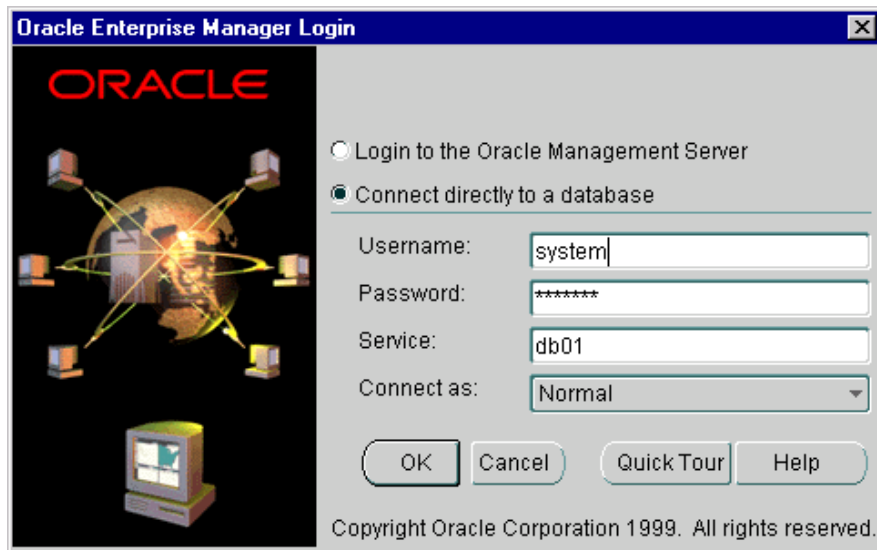


- e** Click Apply.

## Practice 18 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet.



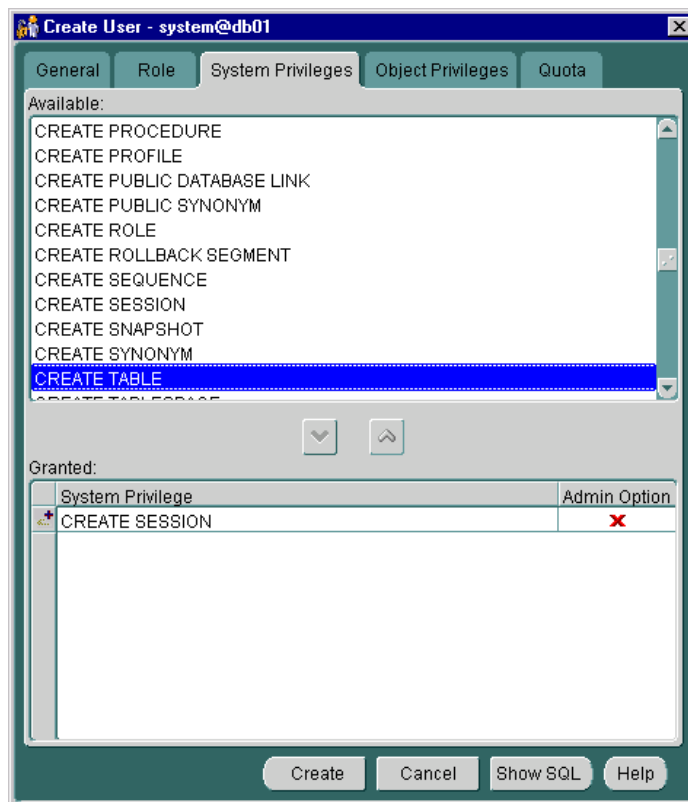
### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 As `system`, create user Kay and give her the capability to log on to the database and create objects in her schema.

**Hint:** Kay needs the CREATE SESSION and CREATE TABLE privileges.

- a** Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Security Manager.
- b** Select the Users folder and choose Create from the right mouse button menu.
- c** Specify username, password, default tablespace, and temporary tablespace in the General page.
- d** Specify a quota of 1MB on DATA01 in the Quotas page.
- e** Select System Privileges for Privilege Type, and grant CREATE TABLE privilege in the Roles/Privileges page.



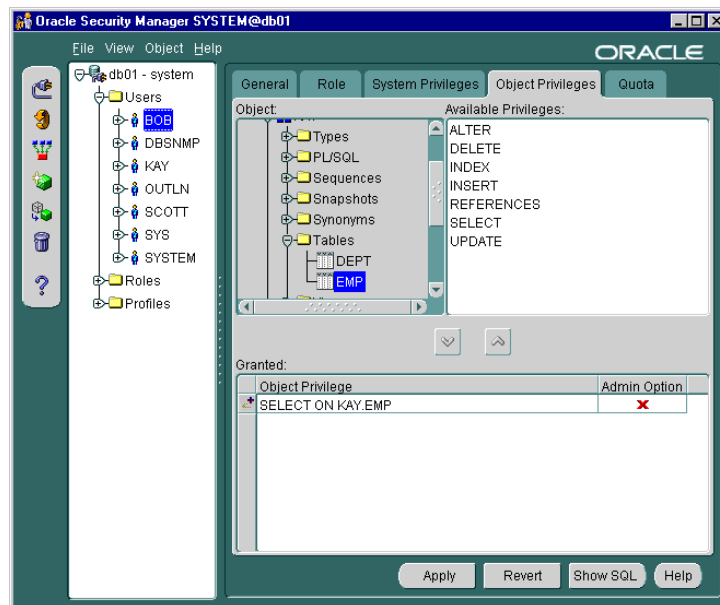
- f** Click Create. Click OK.

- 2** (a) The solution is the same as in Appendix C.
- (b) The solution is the same as in Appendix C.

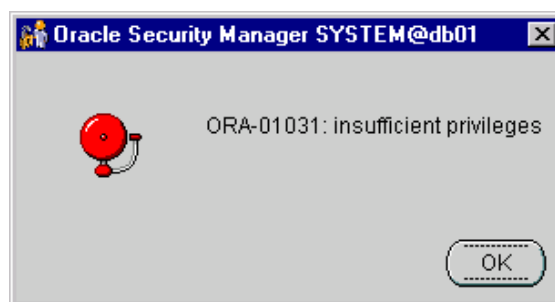


(c) As system, give Bob the ability to select from Kay's EMP table. What happens and why?

- a Use Security Manager.
- b Expand the Users folder.
- c Select user BOB.
- d In the Object Privileges page, expand KAY, expand TABLES, and select EMP.
- e Choose SELECT from Available Privileges and press the down arrow key.



- f Click Apply.



**SYSTEM can query Kay's table using the SELECT ANY TABLE privilege, but only Kay or any other user who received the privilege from Kay with GRANT OPTION can grant this privilege to others.**

**3** The solution is the same as in Appendix C.

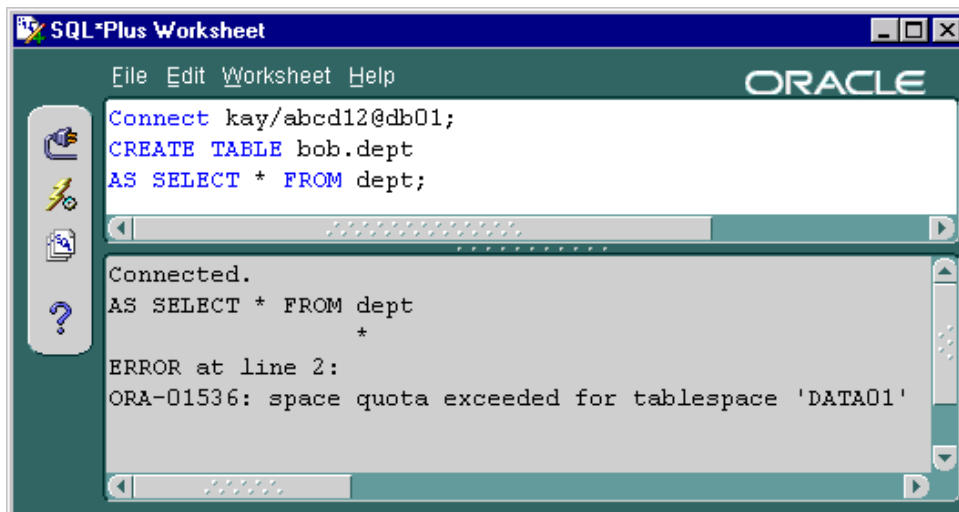
**4** Create user Todd with the capability to log on to the database

- a** Use Security Manager.
- b** Select the Users folder and choose Create from the right mouse button menu.
- c** Specify username, password, default tablespace, and temporary tablespace in the General page.
- d** Click Create.

**5** The solution is the same as in Appendix C.

**6** (a) Enable Kay to create tables in any schema. As Kay, create the table DEPT in Bob's schema as a copy of KAY.DEPT. What happened and why?

- a** Use Security Manager.
- b** Expand the Users folder.
- c** Select KAY.
- d** In the Roles/Privileges page, assign the CREATE ANY TABLE privilege.
- e** Click Apply.



**Bob's quota on his default tablespace was removed in Practice 17, question 6.**

(b) The solution is the same as in Appendix C.

- 7** Give Kay the ability to start up and shut down the database without the ability to create a new database, and verify the results.

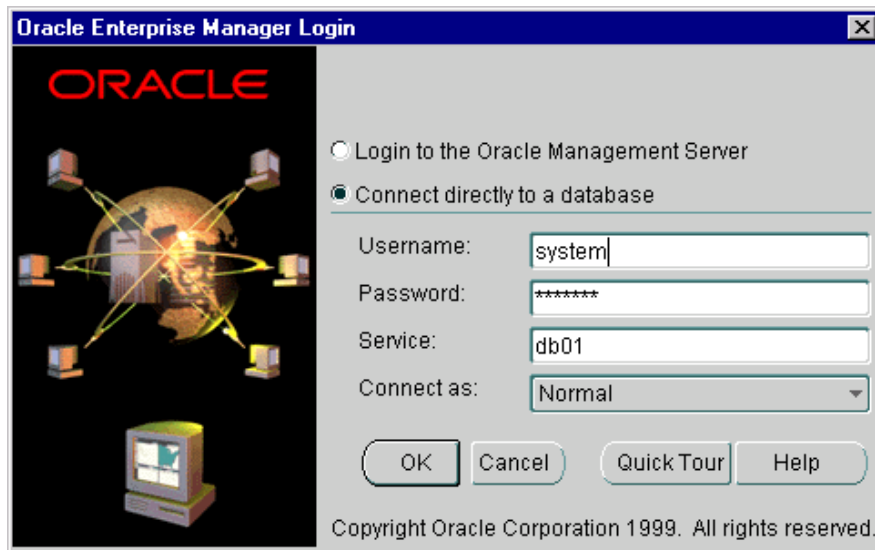
**Hint:** Give Kay the SYSOPER privilege.

- a** Use Security Manager (connect as SYSDBA).
- b** Expand the Users folder.
- c** Select KAY.
- d** In the Roles/Privileges page, assign the SYSOPER privilege.
- e** Click Apply.

## Practice 19 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet.



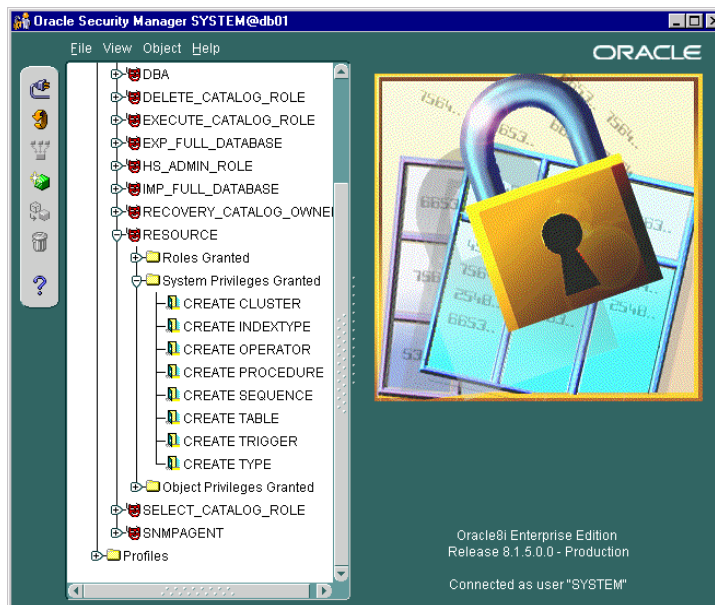
### Step Two

Make sure you connect directly to a database. Enter the username `system`, the password `manager`, and the service name for your working database, and click OK.

- 1 Examine the data dictionary view and list the system privileges of the RESOURCE role.

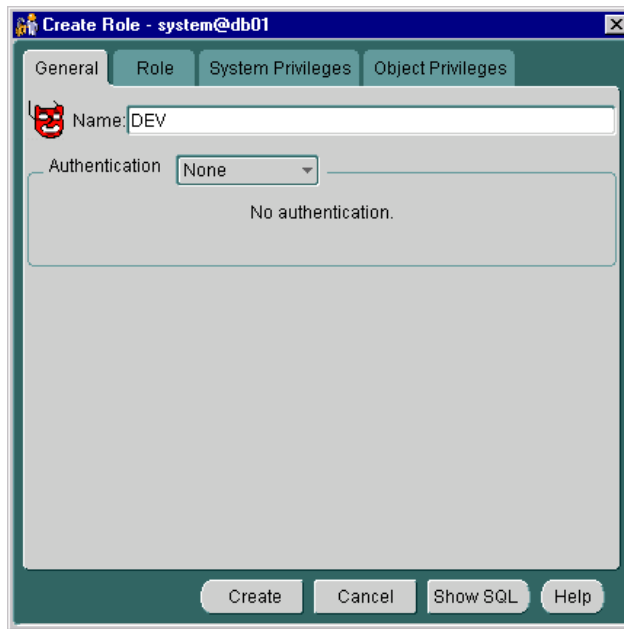
**Hint:** This information is available from DBA\_SYS\_PRIVS.

- a Select Programs—>Oracle - *EMV2 Home*—>DBA Management Pack—>Security Manager.
- b Expand the Roles folder.
- c Expand RESOURCE.
- d Expand System Privileges Granted.



- 2** Create a role called DEV that enables a user to create a table, create a view, and select from Kay's EMP table.

- a** Use Security Manager.
- b** Select the Roles folder and choose Create from the right mouse button menu.
- c** Specify name in General page.



- d** Assign CREATE TABLE, CREATE VIEW in the Roles/Privileges page.
- e** Click Create.
- f** Refer to Appendix C for using the SQL\*Plus Worksheet to finish this practice.

- 3** (a) Assign the RESOURCE and DEV roles to Bob, but make only the RESOURCE role automatically enabled when he logs on.

**Hint:** Use the ALTER USER command to specify the default role.

- a** Use Security Manager.
- b** Expand the Users folder.
- c** Select BOB.
- d** In the Roles/Privileges page, assign DEV and RESOURCE roles.
- e** Under Granted in the same page, deselect Default against DEV.
- f** Click Apply.

- (b) Give Bob the ability to read all the data dictionary information.

**Hint:** Assign the SELECT\_CATALOG\_ROLE to Bob.

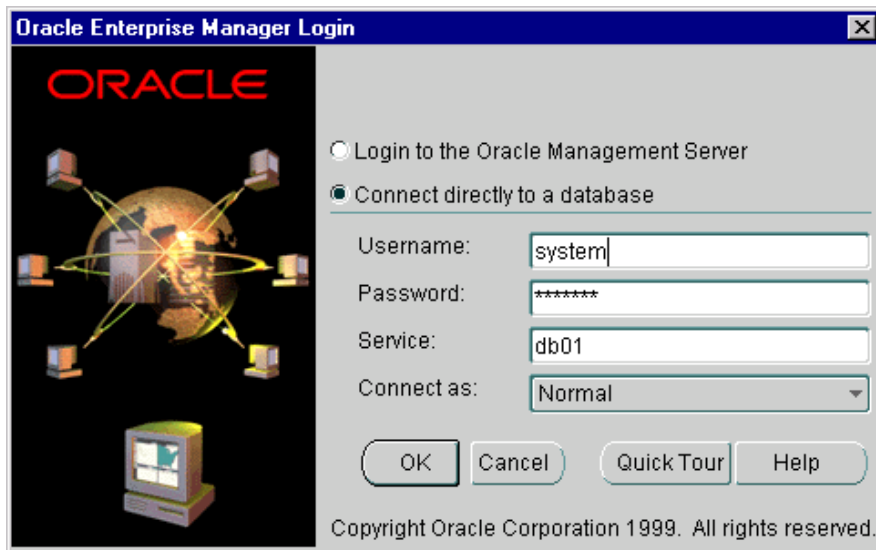
- a** Use Security Manager.
- b** Expand Users.
- c** Select BOB.
- d** In the Roles/Privileges page, assign SELECT\_CATALOG\_ROLE role, but do not make it a default role.
- e** Click Apply.

- 4** The solution is the same as in Appendix C.
- 5** The solution is the same as in Appendix C.

## Practice 20 Solutions

### Step One

(N) Start—>Programs—>Oracle - *EMV2 Home*—>DBA Management Pack  
—>SQL Plus Worksheet.



### Step Two

Make sure you connect directly to a database. Enter the username *system*, the password *manager*, and the service name for your working database, and click OK.

- 1 The solution is the same as in Appendix C.
- 2 The solution is the same as in Appendix C.
- 3 The solution is the same as in Appendix C.
- 4 The solution is the same as in Appendix C



---

E

---

## **Certification Test: Sample Questions**

## **Oracle Certified Professional (OCP) Program: Oracle Certified Database Administrator Track**

The OCP Program was developed by Oracle to recognize technical professionals who can demonstrate the depth of knowledge and hands-on skills required to fully support Oracle core products according to a standard of excellence established by Oracle.

### **A Performance-Based Credential**

To become an Oracle Certified Database Administrator, you must pass four separate tests. Each test will challenge you to apply specific knowledge you have gained through Oracle training, as well as experience you have developed on your job.

Working on a computer with no other resources, you will have up to 1.5 hours to answer 60–70 multiple-choice questions for each examination. Because there is no way to memorize what you need to know, only proven performers will be able to pass the test.

You are not required to take any Oracle Education courses before completing the certification tests for your job role track. However, this course is great preparation for test 2: Oracle Database Administration.

### **Are You Ready?**

Below are some test questions that are similar to the questions from:

- Test 2: Oracle Database Administration
- Test 3: Oracle Backup and Recovery

The answers are provided at the end of this appendix.

## Oracle Database Administration: Sample Test

- 1 The database has two redo log groups and each group has two members. Each member of the noncurrent online redo log group is corrupted. How could you reinitialize these files to begin using them again?
  - a Issue the ALTER DATABASE command to clear the corrupted files.
  - b Issue the ALTER DATABASE command to drop each member from the operating system and then re-create them.
  - c You must create a third group so that you can drop the corrupted group.
  - d The file must be deleted at the operating system level and re-created with the same name.
  
- 2 You need to allocate more space to the USERS tablespace, but the disk containing the data files is full. You want to take the tablespace offline so that you can move the associated files to another disk. The data files do not have errors and you need to be sure a checkpoint occurs when the tablespace is taken offline. Several users currently have transactions pending for the USERS tablespace. Which mode should you use to take the tablespace offline?
  - a IMMEDIATE
  - b NORMAL
  - c TEMPORARY
  
- 3 You have created a table and inserted data with no constraints enabled. You need to enable the table constraints to ensure that any new data inserted does not violate the constraints. You know that the table already contains data that violates the constraints, but you cannot update this data right now. Which state would be most appropriate for the integrity constraints?
  - a Disabled
  - b Enabled NOVALIDATE
  - c Enabled VALIDATE
  - d You cannot add a constraint to a table containing data that violates the constraint.
  - e A constraint cannot be enabled that will apply only to new transactions.

- 4** Currently, all database users are authenticated by the Oracle server. You want to ensure that all users use a password that contains three characters followed by three numbers. How would you set up the server to restrict the format of all users' passwords?
- a** You can create a profile that restricts the password format and assign it to each user.
  - b** You must alter the PASSWORD parameter in *init.ora* to enable the password control mechanism.
  - c** You must enable the password control mechanism when the user is created.
  - d** You cannot control the format of a user's password.
- 5** In your current session, the NLS\_LANG parameter is set to GERMAN\_GERMANY.WE8ISO8859P1. You are going to use SQL\*Loader to export and import several tables. You want to convert the data and import it into a database with a different character set. How could you accomplish this task?
- a** Set the NLS\_LANG parameter for the export.
  - b** Shut down the instance and restart it using the desired character set before exporting the data.
  - c** The data cannot be converted.
  - d** Set the NLS\_LANG parameter for the import.
- 6** You are determining the number of rollback segments needed for your OLTP environment. All of your transactions will be short and you have estimated that there will be at least 16 concurrent transactions during normal usage. What is the recommended number of rollback segments for this environment?
- a** 1
  - b** 2
  - c** 4
  - d** 8

## Oracle Backup and Recovery Sample Test

- 1 You are creating a new Oracle8 database and will be using Recovery Manager. Which memory structure should you explicitly create for use with Recovery Manager?
  - a Log buffer
  - b Shared pool
  - c Data buffer cache
  - d Large pool
- 2 The system administrator has just notified you that he has allocated two new disk drives to the PROD database. The database is currently being archived but you are concerned about the recent frequency of media failures. Now that you have additional disk resources, which *init.ora* parameter can you set to provide added protection against the loss of archived log files due to media failure?
  - a LOG\_ARCHIVE\_MIN\_SUCCEED\_DEST
  - b LOG\_ARCHIVE\_FORMAT
  - c LOG\_ARCHIVE\_DUPLEX\_DEST
  - d LOG\_ARCHIVE\_DEST
- 3 You are responsible for a hybrid database that contains large tablespaces, which are updated infrequently and used for reporting purposes only. This database also includes tablespaces that contain DML intensive tables. Why should you consider using Recovery Manager to maintain the backup and restore operations for this database?
  - a Incremental block-level backups can be performed, reducing backup and recovery time.
  - b By default, backups will not be performed for the large tablespaces, reducing backup time.
  - c Restoring operations will be faster because the large tablespaces will not have to be restored.
  - d Online table-level backups can be performed, reducing recovery time.

- 4** You are maintaining a recovery catalog for the PROD database and normally update the contents of the catalog each Friday afternoon; however, the development team created a new tablespace earlier in the week. In order to perform complete recovery using Recovery Manager, what must you do now to update the recovery catalog so that it reflects the changes made to the database?
- a** Update the recovery catalog by issuing the REGISTER DATABASE command.
  - b** Update the recovery catalog by issuing the RESYNC CATALOG command.
  - c** No action needs to be taken; the recovery catalog is automatically updated by Oracle8 when the database structure changes.
  - d** Update the recovery catalog by issuing the CATALOG DATAFILECOPY <name> TAG = <string> command for the new data file associated with the new tablespace.
  - e** Update the recovery catalog by issuing the RESET DATABASE command.
- 5** You have decided to use Recovery Manager to maintain backup and restore operations for the PROD database, which contains many large data files. What is one of the benefits of using Recovery Manager with this database?
- a** Mirror images of all data files will be automatically included in the backup set.
  - b** Backup sets will be written directly to tape.
  - c** Backup sets will be compressed by not including empty blocks in data files.
  - d** Backup sets will not include nonessential database files.
- 6** You are performing an online backup of the PROD database and a power outage occurs. Once the database is mounted, you query V\$BACKUP and determine that of the five data files being backed up, the second file was left in “hot backup” mode. Which action should you take to synchronize the data files so that you can open the database to users?
- a** Place the tablespace containing the data file offline by issuing the ALTER TABLESPACE <name> OFFLINE command.
  - b** Unfreeze the header for File #2 by issuing the ALTER DATABASE DATAFILE 3 END BACKUP command.
  - c** Shut down the database and restore from the previous full offline backup.
  - d** Shut down the database; restart it, then issue the RECOVER DATAFILE command to recover the second data file.
  - e** Determine the tablespace containing the second file and unfreeze the header by issuing the ALTER TABLESPACE <name> END BACKUP command.

- 7** The PROD database is in a “hung” state and you attempt to resolve the problem. Upon further investigation, you learn that the current redo log group has become corrupted. What can you do to make the database operational?
- a** Shut down the database, re-create the redo log group, then start up the database.
  - b** Clear the current log file by issuing the ALTER DATABASE CLEAR UNARCHIVED LOGFILE GROUP # command.
  - c** Shut down the database with the SHUTDOWN ABORT command and then restart the database.
  - d** Delete the current log file and issue the RECOVER DATABASE UNTIL CANCEL command, recovering the database to the last archived log file.

## **Answers**

### **Oracle Database Administration: Sample Test**

- 1 A**
- 2 B**
- 3 B**
- 4 A**
- 5 D**
- 6 C**

### **Oracle Backup and Recovery: Sample Test**

- 1 D**
- 2 C**
- 3 A**
- 4 B**
- 5 C**
- 6 B**
- 7 B**



## Registering for an OCP Test

OCP Program tests are offered at Authorized Prometric Testing Centers worldwide. Call the Sylvan Prometric Regional Service Center serving your country to find out which Authorized Prometric Testing Center is most convenient for you, and then register for a test in the appropriate certification track.

- Sylvan Sydney, Australia Regional Service Center +61.2.9414.3666
- Sylvan Paris, France Regional Service Center +33.1.4289.3122
- Sylvan Düsseldorf, Germany Regional Service Center +49.2159.9233.50
- Sylvan London, England Regional Service Center +44.181.607.9090
- Sylvan Latin America Regional Service Center +1.612.820.5200
- Sylvan North America Regional Service Center +1.800.891.3926

For more detailed information about specific OCP job role–related tracks, please contact your local Oracle Education representative.

- Americas Headquarters Phone +1.650.506.7000
- Asia/Pacific Rim/India Headquarters Phone +65.337.3797
- Europe/Middle East/Africa Headquarters Phone +31.30.669.9000
- Japan Headquarters Phone +81.3.5213.6666

